Cologne Chip

HFC-S mini

ISDN HDLC FIFO controller
with
S/T interface and integrated FIFOs

Data Sheet ✦ March 2006

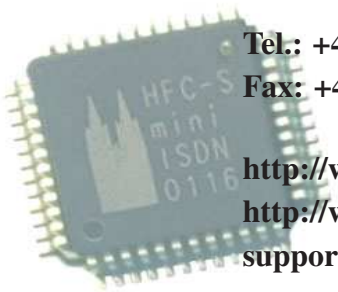
Revision History

Date	Remarks
March 2006	The consolidation begun with the previous data sheet revision attains its conclusion with this revision. Complete contents of the data sheet are now divided into topic-specific chapters. Plentifully information and examples were added, in order to facilitate the understanding. Nearly every chapter in the documentation was changed thereby – but the core statements to HFC-S mini are unchanged, of course. Some register names are changed (refer to remark on p. 15).
July 2003	The data sheet has completely been revised. Section “Processor interface modes” moved into section “Microprocessor interface”, “List of registers” moved to the document preamble, information has been added to FIFO initialization, transformer list, processor interface timings, PCM data rate restrictions, electrical characteristics, clock synchronization, cascade-connected HFC-S mini with only one quartz circuitry and the following registers: INC_RES_F, RAM_DATA, FIFO, F_USAGE, F_FILL, FIF_DATA_NOINC, F_THRES, INT_S1, INT_M1, INT_M2, CON_HDLC, STATUS, MST_MODE0, MST_MODE1, MST_MODE2, ST_RD_STA, ST_WR_STA, CLKDEL.
September 2001	Information added to: MST_MODE0 and CON_HDLC register description.

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



Contents

List of Figures

List of Tables

List of Registers

Important !

Due to a thorough revision of the HFC-S mini data sheet, some registers have to be renamed.

Now, as already known from other data sheets of Cologne Chip, all register names start with R_ (normal registers) or A_ (array registers). Furthermore, some names are changed to fit with common usage of Cologne Chip's abbreviations and to avoid misinterpreting (please see 'List of abbreviations' at the end of this document). Now all Bits and Bitmaps have a unique name.

For more clearness, every register table shows the formerly name as well. At the beginning of this document, the 'List of registers' is displayed in triplicate: sorted by name, sorted by address *and* sorted by formerly name. This ensures a fast location of the register tables even if you have the formerly name in mind.

An additional benefit of these steps is, that Cologne Chip now can present programming header files including all registers and bitmaps. Please ask our support team in case of interest.

Please note !

Register addresses are assigned independently for write and read access; i.e. in some 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.

Registers sorted by name

Write only registers:

Address	Name	Reset group	Page
0xF4	A_CH_MSK[FIFO]	S, H	95
0xFC	A_CHANNEL[FIFO]	S, H	99
0xFA	A_CON_HDLC[FIFO]	S, H	96
0x84	A_FIFO_DATA_NOINC		94
0xFB	A_HDLC_PAR[FIFO]	S, H	98
0x0E	A_INC_RES_FIFO[FIFO]	S, H	93
0x26	R_AUX1_RX_SL	S, H	169
0x22	R_AUX1_TX_SL	S, H	166
0x2E	R_AUX1_TX		172
0x27	R_AUX2_RX_SL	S, H	170
0x23	R_AUX2_TX_SL	S, H	167
0x2F	R_AUX2_TX		172
0x24	R_B1_RX_SL	S, H	168
0x20	R_B1_TX_SL	S, H	164
0x2C	R_B1_TX		171
0x25	R_B2_RX_SL	S, H	168
0x21	R_B2_TX_SL	S, H	165
0x2D	R_B2_TX		172
0x28	R_CI_TX	S, H	170
0x00	R_CIRM	H	43
0x0D	R_DF_MD	S, H	93
0x1A	R_FIFO_IRQMSK	S, H	185
0x0B	R_FIFO_REV	S, H	91
0x0C	R_FIFO_THRES	S, H	92
0x0F	R_FIFO	S, H	94
0x1B	R_MISC_IRQMSK	S, H	186
0x2A	R_MON1_TX	S, H	171
0x2B	R_MON2_TX	S, H	171
0x14	R_PCM_MD0	S, H	159
0x15	R_PCM_MD1	S, H	161
0x16	R_PCM_MD2	S, H	163
0x08	R_RAM_ADDR0	S, H	43
0x09	R_RAM_ADDR1	S, H	44
0x3C	R_ST_B1_TX		129
0x3D	R_ST_B2_TX		129
0x37	R_ST_CLK_DLY		128
0x31	R_ST_CTRL0	S, H	125
0x32	R_ST_CTRL1	S, H	126
0x33	R_ST_CTRL2	S, H	127
0x3E	R_ST_D_TX		129
0x34	R_ST_SQ_WR	S, H	127
0x30	R_ST_WR_STA	S, H	124
0x1C	R_TI	S, H	187

Read only registers:

Address	Name	Reset group	Page
0x0C	A_F1[FIFO]	S, H	100
0x0D	A_F2[FIFO]	S, H	100
0x1A	A_USAGE[FIFO]	S, H	101
0x04	A_Z1[FIFO]	S, H	99
0x06	A_Z2[FIFO]	S, H	100
0x2E	R_AUX1_RX		176
0x2F	R_AUX2_RX		176
0x2C	R_B1_RX		175
0x2D	R_B2_RX		175
0x16	R_CHIP_ID	S, H	45
0x28	R_CI_RX	S, H	173
0x19	R_F0_CNTH	S, H	173
0x18	R_F0_CNTL	S, H	173
0x10	R_FIFO_IRQ	S, H	188
0x1B	R_FILL	S, H	101
0x11	R_MISC_IRQ	S, H	189
0x2A	R_MON1_RX	S, H	174
0x2B	R_MON2_RX	S, H	175
0x29	R_PCM_GCI_STA	S, H	174
0x3C	R_ST_B1_RX		131
0x3D	R_ST_B2_RX		132
0x3E	R_ST_D_RX		132
0x3F	R_ST_E_RX		132
0x30	R_ST_RD_STA	S, H	130
0x34	R_ST_SQ_RD		131
0x1C	R_STATUS	S, H	190

Read / Write registers:

Address	Name	Reset group	Page
0x80	A_FIFO_DATA[FIFO]		102
0xC0	R_RAM_DATA		46

Registers sorted by address

Write only registers:

Address	Name	Reset group	Page
0x00	R_CIRM	H	43
0x08	R_RAM_ADDR0	S, H	43
0x09	R_RAM_ADDR1	S, H	44
0x0B	R_FIFO_REV	S, H	91
0x0C	R_FIFO_THRES	S, H	92
0x0D	R_DF_MD	S, H	93
0x0E	A_INC_RES_FIFO[FIFO]	S, H	93
0x0F	R_FIFO	S, H	94
0x14	R_PCM_MD0	S, H	159
0x15	R_PCM_MD1	S, H	161
0x16	R_PCM_MD2	S, H	163
0x1A	R_FIFO_IRQMSK	S, H	185
0x1B	R_MISC_IRQMSK	S, H	186
0x1C	R_TI	S, H	187
0x20	R_B1_TX_SL	S, H	164
0x21	R_B2_TX_SL	S, H	165
0x22	R_AUX1_TX_SL	S, H	166
0x23	R_AUX2_TX_SL	S, H	167
0x24	R_B1_RX_SL	S, H	168
0x25	R_B2_RX_SL	S, H	168
0x26	R_AUX1_RX_SL	S, H	169
0x27	R_AUX2_RX_SL	S, H	170
0x28	R_CI_TX	S, H	170
0x2A	R_MON1_TX	S, H	171
0x2B	R_MON2_TX	S, H	171
0x2C	R_B1_TX		171
0x2D	R_B2_TX		172
0x2E	R_AUX1_TX		172
0x2F	R_AUX2_TX		172
0x30	R_ST_WR_STA	S, H	124
0x31	R_ST_CTRL0	S, H	125
0x32	R_ST_CTRL1	S, H	126
0x33	R_ST_CTRL2	S, H	127
0x34	R_ST_SQ_WR	S, H	127
0x37	R_ST_CLK_DLY		128
0x3C	R_ST_B1_TX		129
0x3D	R_ST_B2_TX		129
0x3E	R_ST_D_TX		129
0x84	A_FIFO_DATA_NOINC		94
0xF4	A_CH_MSK[FIFO]	S, H	95
0xFA	A_CON_HDLC[FIFO]	S, H	96
0xFB	A_HDLC_PAR[FIFO]	S, H	98
0xFC	A_CHANNEL[FIFO]	S, H	99

Read only registers:

Address	Name	Reset group	Page
0x04	A_Z1[FIFO]	S, H	99
0x06	A_Z2[FIFO]	S, H	100
0x0C	A_F1[FIFO]	S, H	100
0x0D	A_F2[FIFO]	S, H	100
0x10	R_FIFO_IRQ	S, H	188
0x11	R_MISC_IRQ	S, H	189
0x16	R_CHIP_ID	S, H	45
0x18	R_F0_CNTL	S, H	173
0x19	R_F0_CNTH	S, H	173
0x1A	A_USAGE[FIFO]	S, H	101
0x1B	R_FILL	S, H	101
0x1C	R_STATUS	S, H	190
0x28	R_CI_RX	S, H	173
0x29	R_PCM_GCI_STA	S, H	174
0x2A	R_MON1_RX	S, H	174
0x2B	R_MON2_RX	S, H	175
0x2C	R_B1_RX		175
0x2D	R_B2_RX		175
0x2E	R_AUX1_RX		176
0x2F	R_AUX2_RX		176
0x30	R_ST_RD_STA	S, H	130
0x34	R_ST_SQ_RD		131
0x3C	R_ST_B1_RX		131
0x3D	R_ST_B2_RX		132
0x3E	R_ST_D_RX		132
0x3F	R_ST_E_RX		132

Read / Write registers:

Address	Name	Reset group	Page
0x80	A_FIFO_DATA[FIFO]		102
0xC0	R_RAM_DATA		46

Registers sorted by formerly name

Write only registers:

Address	Formerly name	New Name	Page
0x2E	AUX1_D	R_AUX1_TX	172
0x26	AUX1_RSL	R_AUX1_RX_SL	169
0x22	AUX1_SSL	R_AUX1_TX_SL	166
0x2F	AUX2_D	R_AUX2_TX	172
0x27	AUX2_RSL	R_AUX2_RX_SL	170
0x23	AUX2_SSL	R_AUX2_TX_SL	167
0x2C	B1_D	R_B1_TX	171
0x24	B1_RSL	R_B1_RX_SL	168
0x3C	B1_SEND	R_ST_B1_TX	129
0x20	B1_SSL	R_B1_TX_SL	164
0x2D	B2_D	R_B2_TX	172
0x25	B2_RSL	R_B2_RX_SL	168
0x3D	B2_SEND	R_ST_B2_TX	129
0x21	B2_SSL	R_B2_TX_SL	165
0x28	C/I	R_CI_TX	170
0xFC	CHANNEL	A_CHANNEL	99
0xF4	CH_MSK	A_CH_MSK	95
0x00	CIRM	R_CIRM	43
0x37	CLKDEL	R_ST_CLK_DLY	128
0xFA	CON_HDLC	A_CON_HDLC	96
0x3E	D_SEND	R_ST_D_TX	129
0x0F	FIFO	R_FIFO	94
0x84	FIF_DATA	A_FIFO_DATA_NOINC	94
0x0B	F_CROSS	R_FIFO_REV	91
0x0D	F_MODE	R_DF_MD	93
0x0C	F_THRES	R_FIFO_THRES	92
0xFB	HDLC_PAR	A_HDLC_PAR	98
0x0E	INC_RES_F	A_INC_RES_FIFO	93
0x1A	INT_M1	R_FIFO_IRQMSK	185
0x1B	INT_M2	R_MISC_IRQMSK	186
0x2A	MON1_D	R_MON1_TX	171
0x2B	MON2_D	R_MON2_TX	171
0x14	MST_MODE0	R_PCM_MD0	159
0x15	MST_MODE1	R_PCM_MD1	161
0x16	MST_MODE2	R_PCM_MD2	163
0x09	RAM_ADR_H	R_RAM_ADDR1	44
0x08	RAM_ADR_L	R_RAM_ADDR0	43
0x32	SCTRL_E	R_ST_CTRL1	126
0x33	SCTRL_R	R_ST_CTRL2	127
0x31	SCTRL	R_ST_CTRL0	125
0x34	SQ_SEND	R_ST_SQ_WR	127
0x30	STATES	R_ST_WR_STA	124
0x1C	TIME_SEL	R_TI	187

Read only registers:

Address	Formerly name	New Name	Page
0x2E	AUX1_D	R_AUX1_RX	176
0x2F	AUX2_D	R_AUX2_RX	176
0x2C	B1_D	R_B1_RX	175
0x3C	B1_REC	R_ST_B1_RX	131
0x2D	B2_D	R_B2_RX	175
0x3D	B2_REC	R_ST_B2_RX	132
0x28	C/I	R_CI_RX	173
0x16	CHIP_ID	R_CHIP_ID	45
0x3E	D_REC	R_ST_D_RX	132
0x3F	E_REC	R_ST_E_RX	132
0x19	F0_CNT_H	R_F0_CNTH	173
0x18	F0_CNT_L	R_F0_CNTL	173
0x0C	FIF_F1	A_F1	100
0x0D	FIF_F2	A_F2	100
0x04	FIF_Z1	A_Z1	99
0x06	FIF_Z2	A_Z2	100
0x1B	F_FILL	R_FILL	101
0x1A	F_USAGE	A_USAGE	101
0x10	INT_S1	R_FIFO_IRQ	188
0x11	INT_S2	R_MISC_IRQ	189
0x2A	MON1_D	R_MON1_RX	174
0x2B	MON2_D	R_MON2_RX	175
0x34	SQ_REC	R_ST_SQ_RD	131
0x30	STATES	R_ST_RD_STA	130
0x1C	STATUS	R_STATUS	190
0x29	TRxR	R_PCM_GCI_STA	174

Read / Write registers:

Address	Formerly name	New Name	Page
0x80	FIF_DATA	A_FIFO_DATA	102
0xC0	RAM_DATA	R_RAM_DATA	46

About this data sheet and Cologne Chip technical support

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

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

Organization of this data sheet

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

Links between chapters are mentioned in the text.

Development tools

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

- An evaluation board of HFC-S mini is available. This can be used in a common PC environment (using PCI bus) or can directly be attached to the target microcontroller platform.
- A demo layer 1 driver as source code is 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.
- Broad commercial driver support from experienced ISDN design houses as well as various open source driver implementations (e.g. for Linux) are available.

Visit our web site

Our web site (`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 the HFC-S mini, especially a general block diagram and a feature list. Pinout diagrams for the processor interfaces and a detailed list of all pins complete this chapter.

Chapter "Microprocessor bus interface": (2) The HFC-S mini supports a microprocessor interface which is explained in this chapter. This includes signal and timing characteristics as well as register access and typical connection circuitries. *(Prerequisite knowledge of the chosen interface mode is strongly recommended.)*

Chapter "HFC-S mini 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 the HFC-S mini.)*

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 "S/T interface": (5) The HFC-S mini has one S/T line interface. 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 the HFC-S mini. *(Read only when used, but don't skip the overview in this chapter even if the PCM interface is not used!)*

Chapter "Clock, reset, interrupt, timer and watchdog" (7) explains clock generation and distribution, reset function and interrupt capabilities. *(Must be read.)*

Chapter "Electrical characteristics": (8) Some information about the electrical characteristics of the HFC-S mini are given in this chapter. *(Information for hardware design.)*

Chapter "HFC-S mini package dimensions" (9) shows the HFC-S mini package dimensions. *(Information for hardware design.)*

General remarks to notations

1. The decimal point is written as a point (e.g. 1.23). Thousands separators are written with thin space.
2. Numerical values have different notations for various number systems; e.g. the hexadecimal value 0xC9 is '11001001' in binary and 201 in decimal notation.
3. The prefix 'kilo' is written k for the meaning of 1000 and it is written K for the meaning of 1024.
4. The first letter of register names indicates the type: 'R_ ...' is a register or multi-register, while 'A_ ...' is an array register.

Chapter 1

General description

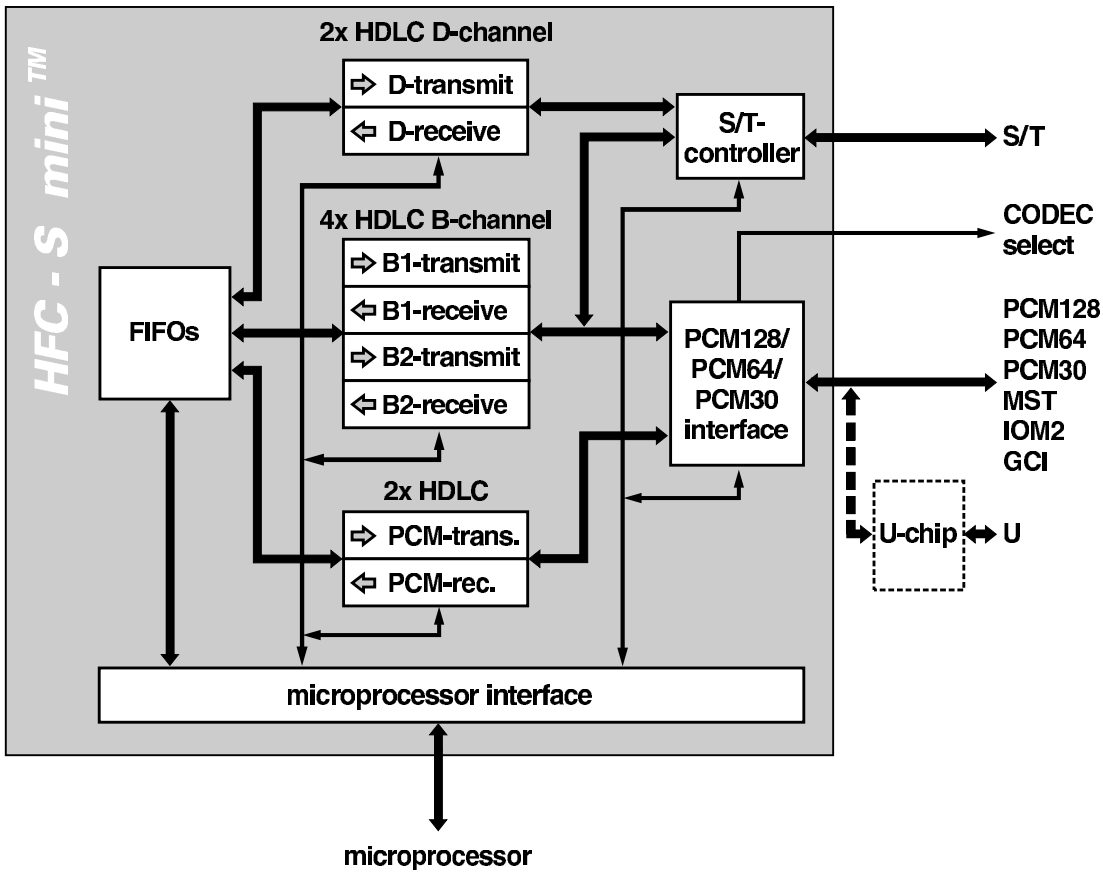


Figure 1.1: *HFC-S mini block diagram*

1.1 System overview

The HFC-S mini is a single-chip ISDN S/T HDLC Basic Rate Interface controller for embedded applications. The S/T interface, HDLC controllers, FIFOs and a microprocessor interface are integrated in the HFC-S mini. A PCM128 / PCM64 / PCM30 interface is also implemented which can be connected to many telecom serial busses. CODECs are usually connected to this interface. All ISDN channels (2 B + 1 D) and the PCM interface are served fully duplex by the 8 integrated FIFOs. HDLC controllers are implemented in hardware so there is no need to implement HDLC on the host processor.

The HFC-S mini can be used for all kinds of ISDN equipment with ISDN Basic Rate Interface, such as

- ISDN terminal adapters (for data communications, e.g. Internet access)
- ISDN terminal adapters (with POTS interfaces)
- ISDN PABX
- ISDN SOHO PABX (switching done by HFC-S mini)
- ISDN telephones
- VoIP gateways / VoIP routers
- Integrated Access Devices (IADs)
- POS terminals
- ISDN video conferencing equipment
- ISDN dialers and LCR (Least Cost Routers)
- ISDN LAN routers
- ISDN protocol analyzers
- ISDN call recording
- ISDN smart NTs

1.2 Features

- 1 integrated S/T interface
- ITU-T I.430 and TBR 3 conform S/T ISDN support in TE [3] and NT mode [3, 1] for 3.3 V and 5 V power supply
- HDLC controller with support for 2 B-channels and 1 D-channel
- Transparent mode independently selectable for all FIFOs
- 4 integrated FIFOs for transmit and receive data each, 128 bytes per FIFO
- Maximum 7 HDLC frames per FIFO
- 56 kbit/s restricted mode for U.S. ISDN lines selectable by software
- Programmable data flow to connect FIFOs, line interface channels and PCM time slots with each other
- PCM128 / PCM64 / PCM30 interface configurable to MST (MVIP) [1] or Siemens IOM™-2 and Motorola GCI for interchip connection to U-chip or external CODECs
- Programmable PCM time slot assigner for 4 channels in transmit and receive direction each (switch matrix for PCM)
- H.100 data rate supported
- Microprocessor interface compatible to Motorola bus and Siemens / Intel bus
- Interrupt controller
- Timer with interrupt capability
- CMOS technology with single 3.3 V or 5 V power supply
- PQFP 48 package
- RoHS compliant (week code dated later than 32-2004)

[1] Mitel Serial Telecom bus

1.3 Pin description

1.3.1 Pinout diagram

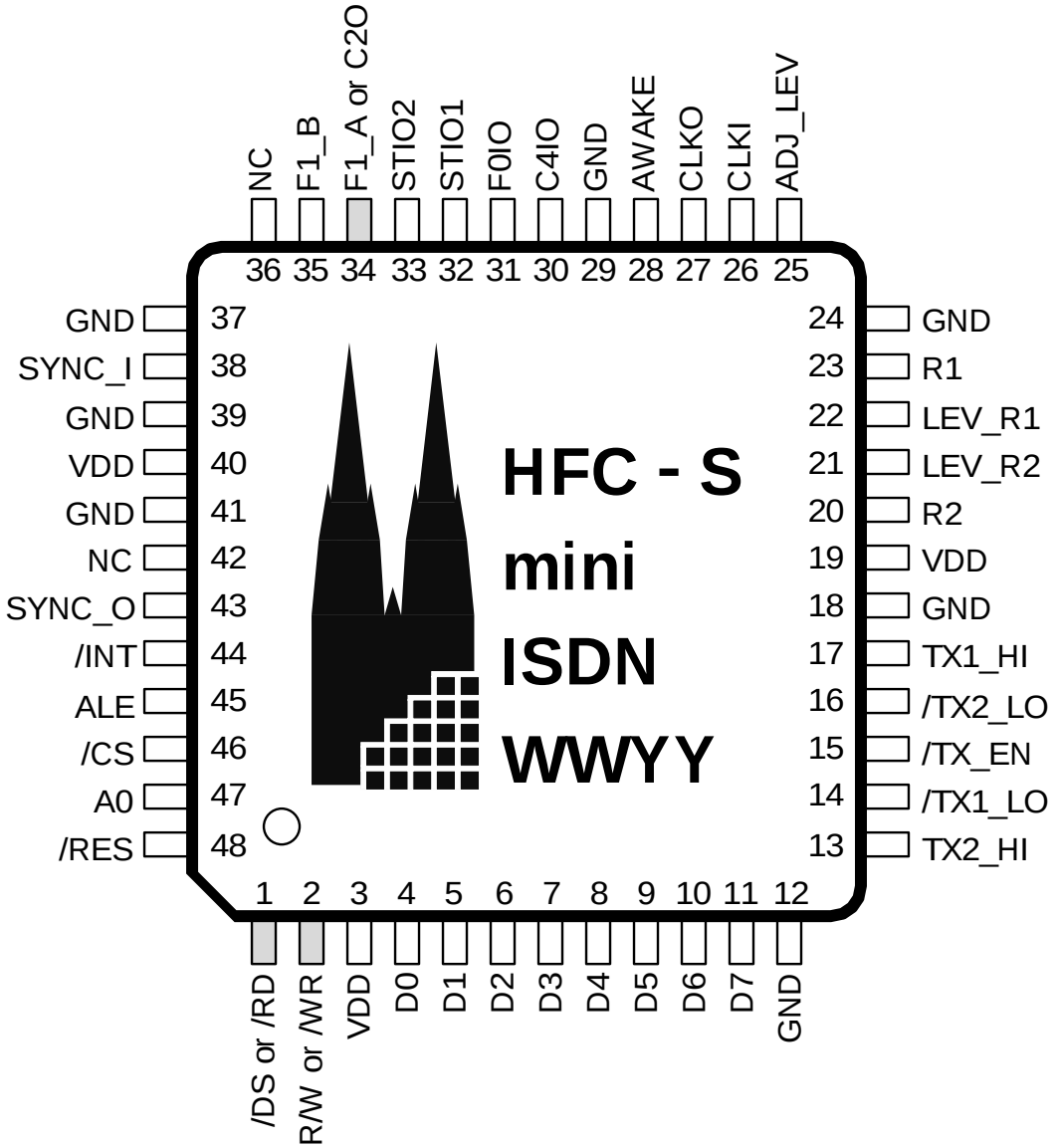


Figure 1.2: *Pinout of HFC-S mini*

1.3.2 Pin list

> **Please note !**
>
> Pin 36 (/WAIT) is no longer available and must not be connected.

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
1	1st function 2nd function	/DS /RD	I I	I/O data strobe (in processor mode 2) Read enable (in processor modes 3 and 4)		
2	1st function 2nd function	R/W /WR	I I	Read/Write select (in processor mode 2) Write enable (in processor modes 3 and 4)		
3		VDD		3.3 Vor 5 Vpower supply		
4		D0	IOr	Data bus bit 0		
5		D1	IOr	Data bus bit 1		
6		D2	IOr	Data bus bit 2		
7		D3	IOr	Data bus bit 3		
8		D4	IOr	Data bus bit 4		
9		D5	IOr	Data bus bit 5		
10		D6	IOr	Data bus bit 6		
11		D7	IOr	Data bus bit 7		
12		GND		Ground		
13		TX2_HI	O	Transmitter output 2		
14		/TX1_LO	O	Ground driver for transmitter 1		
15		/TX_EN	O	Transmitter enable		
16		/TX2_LO	O	Ground driver for transmitter 2		
17		TX1_HI	O	Transmitter output 1		
18		GND		Ground		
19		VDD		3.3 Vor 5 Vpower supply		
20		R2	I	Receiver input 2		
21		LEV_R2	I	Level detect for R2		
22		LEV_R1	I	Level detect for R1		
23		R1	I	Receiver input 1		
24		GND		Ground		
25		ADJ_LEV	Ood	Level generator		
26		CLKI	I	Oscillator input signal		

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
27		CLKO	O	Oscillator output signal		
28		AWAKE	I	Wakeup input		
29		GND		Ground		
30		C4IO	IOpu	PCM double bit clock I/O		
31		F0IO	IOpu	PCM frame clock I/O (8 kHz)		
32		STIO1	IOpu	PCM data bus 1, I or O per time slot		
33		STIO2	IOpu	PCM data bus 2, I or O per time slot		
34	1st function 2nd function	F1_A C2O	O O	External CODEC A enable PCM bit clock output		
35		F1_B	O	External CODEC B enable		
36		NC		Must not be connected		
37		GND		Ground		
38		SYNC_I	I	Synchronization input (8 kHz)		
39		GND		Ground		
40		VDD		3.3 Vor 5 Vpower supply		
41		GND		Ground		
42		NC		Must not be connected		
43		SYNC_O	O	Synchronization output (8 kHz)		
44		/INT	Ood	Interrupt request		
45		ALE	Ipu	Address latch enable		
46		/CS	Ipu	Chip select		
47		A0	I	Address bit 0		
48		/RES	Istpu	Reset		

Legend:

I	Input pin
O	Output pin
IO	Bidirectional pin
Ipu	Input pin with internal pull-up resistor of app. 100 kΩ to VDD
IOpu	Bidirectional pin with internal pull-up resistor of app. 100 kΩ to VDD
Istpu	Input pin with Schmitt trigger characteristic and internal pull-up resistor of app. 100 kΩ to VDD
IOr	Tristated during reset
Ood	Output pin with open drain
NC	Not connected

Chapter 2

Microprocessor bus interface

Table 2.1: *Overview of the microprocessor interface pins*

Number	Name	Description	Number	Name	Description
1	/DS	I/O data strobe (in processor mode 2)	4	D0	Data bus bit 0
1	/RD	Read enable (in processor modes 3 and 4)	5	D1	Data bus bit 1
2	R/W	Read/Write select (in processor mode 2)	6	D2	Data bus bit 2
2	/WR	Write enable (in processor modes 3 and 4)	7	D3	Data bus bit 3
44	/INT	Interrupt request	8	D4	Data bus bit 4
45	ALE	Address latch enable	9	D5	Data bus bit 5
46	/CS	Chip select	10	D6	Data bus bit 6
47	A0	Address bit 0	11	D7	Data bus bit 7
48	/RES	Reset			

Table 2.2: *Overview of the HFC-S mini bus interface registers*

Write only registers:

Address	Name	Page
0x00	R_CIRM	43
0x08	R_RAM_ADDR0	43
0x09	R_RAM_ADDR1	44

Read only register:

Address	Name	Page
0x16	R_CHIP_ID	45

Read / write register:

Address	Name	Page
0xC0	R_RAM_DATA	46

2.1 Processor interface modes

The HFC-S mini has an integrated 8 bit microprocessor interface. It is compatible with Motorola bus and Intel bus. The different microprocessor interface modes are selected during reset by ALE .

In mode 2 (Motorola bus mode) and mode 3 (de-multiplexed Intel bus mode) pin A0 is the address input. The data bus is D7 .. D0 .

In mode 4 (multiplexed Intel bus mode) D[7:0] is the multiplexed address/data bus. A0 must be '0' in this mode.

Mode 2: Motorola bus with control signals /CS, R/W, /DS is selected by setting ALE to VDD .

Mode 3: Intel bus with seperated address bus (A0) and data bus (D7 .. D0) and control signals /CS, /WR, /RD is selected by setting ALE to GND .

Mode 4: Intel bus with multiplexed address bus and data bus with control signals /CS, /WR, /RD, ALE. The first rising edge on ALE switches into this mode. A0 must be '0'. ALE = '1' latches the address. The multiplexed address / data bus is D7 .. D0 .

2.2 Register access

In mode 2 and mode 3 the HFC-S mini has 2 addresses. The lower address (A0 = '0') is used for data read / write. The higher address (A0 = '1') is write only and is used for register selection. Registers are selected by first setting A0 to '1' and then writing the address of the desired register to the data bus D7 .. D0 . All following accesses to the HFC-S mini with A0 = '0' are read / write operations concerning this register.

In mode 4, all internal registers can directly be accessed. In mode 2 and mode 3, first, the address of the desired register must be written to the address with A0 = '1'. Afterwards data can be read / written from / to that register by reading / writing the address with A0 = '0'. In mode 4, A0 must always be '0'.

The function of the control signals is shown in Table 2.3. Except in mode 4, ALE is assumed to be stable after reset.

Please note !

Every asynchronous register read access should be done multiple times until two consecutive read accesses result in the same value. Only this way it is ensured that the read bits are stable.

This information applies to the following registers:

A_USAGE (0x1A),	A_F1 (0x0C),	A_F2 (0x0D),
A_Z1 (0x04),	A_Z2 (0x06),	R_B1_RX (0x2C),
R_B2_RX (0x2D),	R_AUX1_RX (0x2E),	R_AUX2_RX (0x2F),
R_F0_CNTL (0x18),	R_F0_CNTH (0x19),	R_FILL (0x1B),
R_CI_RX (0x28),	R_MON1_RX (0x2A),	R_MON2_RX (0x2B),
R_ST_RD_STA (0x30),	R_ST_SQ_RD (0x34).	

As there is neither a lock nor a latch mechanism implemented with the 16 bit counter F0, it is recommended to read these counter registers in a sequence R_F0_CNTH – R_F0_CNTL – R_F0_CNTL – R_F0_CNTH – until both registers returned the same value twice. This sequence detects a carry event of both registers.

2.3 Signal and timing Characteristics

Table 2.3 shows the interface signals for the different microprocessor interface modes. Timing characteristics are shown in Figures 2.4 and 2.5 for mode 2 and mode 3. Figures 2.6 and 2.7 show mode 4 timing characteristics.

Table 2.3: *Function of the microprocessor interface control signals (X = don't care)*

/RD /DS	/WR R/W	/CS	ALE	Operation	Mode
X	X	1	X	no data access	all
1	1	X	0	no data access	all
0	1	0	1	read data	2
0	0	0	1	write data	2
0	1	0	0	read data	3
1	0	0	0	write data	3
0	1	0	0	read data	4
1	0	0	0	write data	4
X	X	X	1 *	write address	4

*: first 1-pulse switches into mode 4

2.3.1 Register read access in mode 2 (Motorola) and mode 3 (Intel)

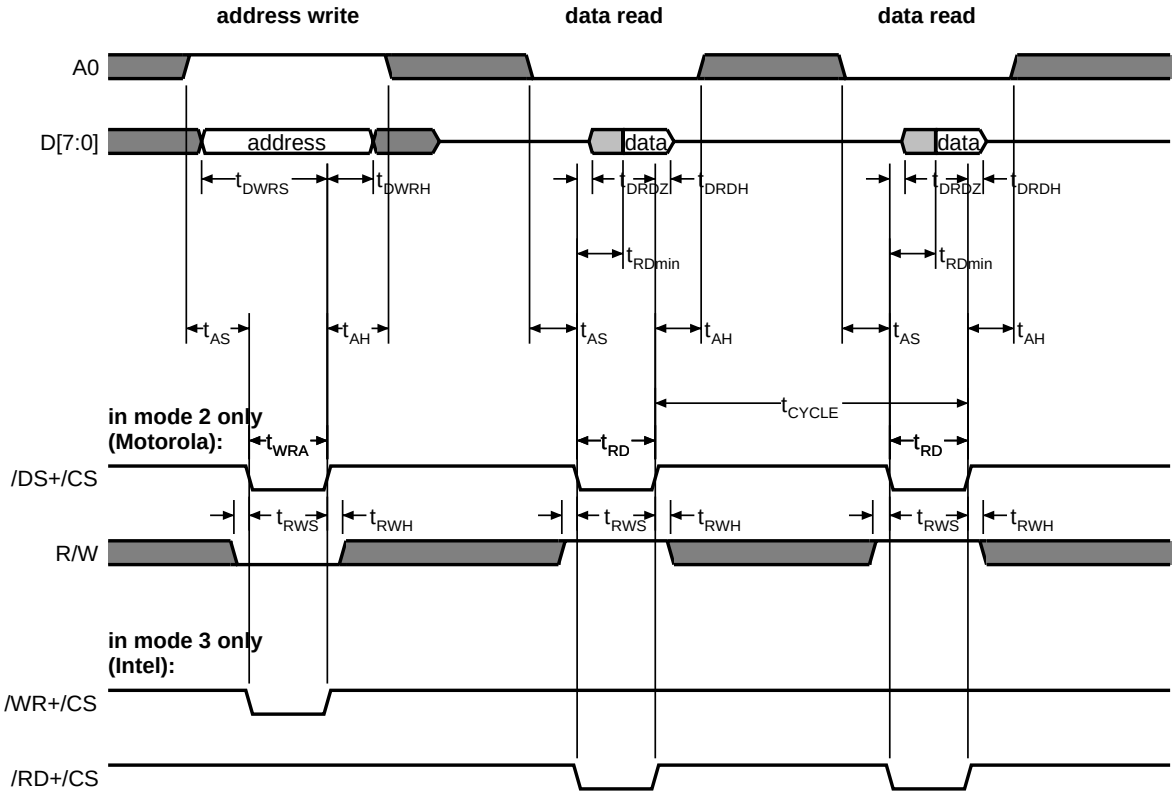


Figure 2.1: *Read access in mode 2 (Motorola) and mode 3 (Intel)*

t_{CLKI} is the CLKI clock period which is normally 40.69 ns (24.576 MHz system clock).

> **Important !**
>
> All read accesses with register address bit 7 equal '1' (registers A_FIFO_DATA, A_FIFO_DATA_NOINC and R_RAM_DATA) have a cycle time
>
> $$t_{CYCLE} \geq 6 \cdot t_{CLKI}$$
>
> between two consecutive ⌟⌜ of t_{RD} (end of read access).

> **Hint !**
>
> If the same register as in the last register read / write access is used again, the register address write is not required.

Table 2.4: *Symbols of read accesses in Figure 2.1*

Symbol	min / ns	max / ns	Characteristic
t_{AS}	10		A0 valid to /DS+/CS (/WR+/CS) ⌝ setup time
t_{AH}	10		Address hold time after /DS+/CS (/WR+/CS) ⌟
t_{WRA}	20		Write time for address write
t_{DWRS}	30		Write data setup time to /DS+/CS (/WR+/CS) ⌟
t_{DWRH}	10		Write data hold time from /DS+/CS (/WR+/CS) ⌟
t_{RD}			Read time:
	$2 \cdot t_{CLKI}$		address bit [7] = '0' (address range 0 ... 0x7F: normal register access)
	20		address bits [7,6] = '10' (address range 0x80 ... 0xBF: FIFO data access)
	$6 \cdot t_{CLKI}$		address bits [7,6] = '11' (address range 0xC0 ... 0xFF: direct internal RAM access) *
t_{CYCLE}			/DS+/CS (/RD+/CS) ⌟ to next end of data access
			address bit [7] = '0' (address range 0 ... 0x7F: normal register access)
	$6 \cdot t_{CLKI}$		address bits [7,6] = '10' (address range 0x80 ... 0xBF: FIFO data access)
	$6 \cdot t_{CLKI}$		address bits [7,6] = '11' (address range 0xC0 ... 0xFF: direct internal RAM access) *
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 ⌝
t_{RWH}	2		R/W hold time after /DS+/CS ⌟

*: Normally this time needs not to be matched because a direct RAM access is not needed.

2.3.2 Register write access in mode 2 (Motorola) and mode 3 (Intel)

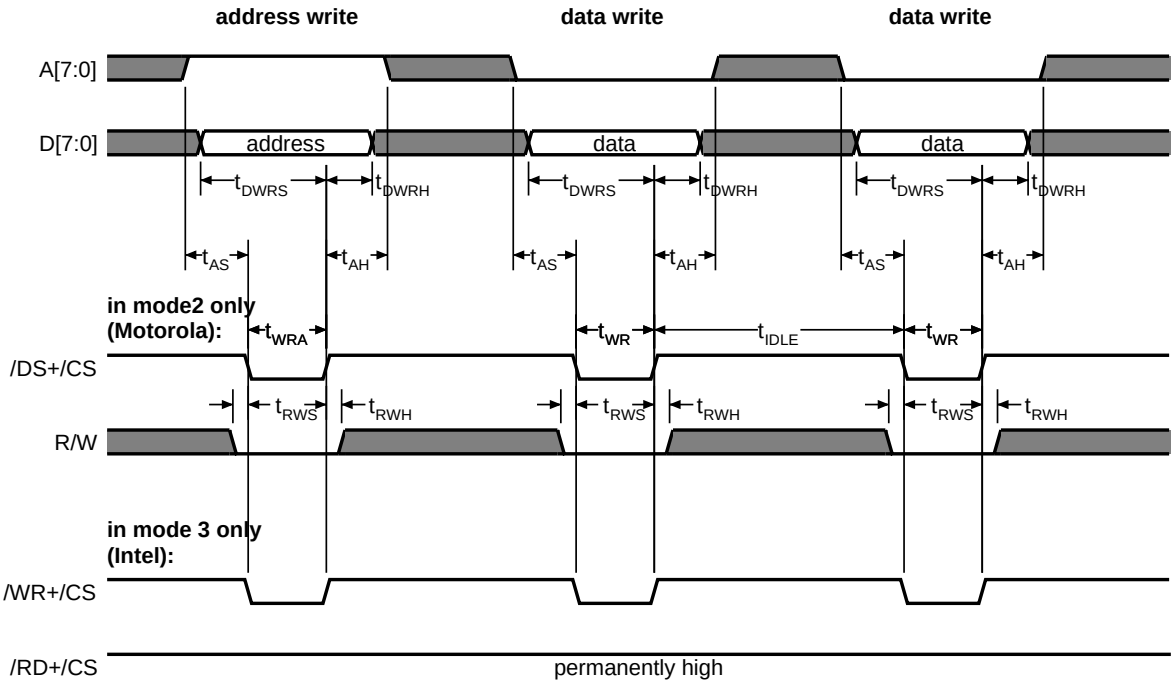


Figure 2.2: *Write access in mode 2 (Motorola) and mode 3 (Intel)*

Table 2.5: *Symbols of write accesses in Figure 2.2*

Symbol	min / ns	max / ns	Characteristic
t_{AS}	10		A0 valid to /DS+/CS (/WR+/CS) ⌜ setup time
t_{AH}	10		Address hold time 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}	10		Write data hold time from /DS+/CS (/WR+/CS) ⌜
t_{WR}	20		Write time
t_{IDLE}			/DS+/CS (/WR+/CS) high time between two data accesses
	20		address bit [7] = '0' (address range 0 ... 0x7F: normal register access)
	$5 \cdot t_{CLKI}$		address bits [7,6] = '10' (address range 0x80 ... 0xBF: FIFO data access)
	$5 \cdot t_{CLKI}$		address bits [7,6] = '11' (address range 0xC0 ... 0xFF)
t_{RWS}	2		R/W setup time to /DS+/CS ⌞
t_{RWH}	2		R/W hold time after /DS+/CS ⌜

t_{CLKI} is the CLKI clock period which is normally 40.69 ns (24.576 MHz system clock).

Important !

All write accesses with register address bit 7 equal '1' (registers A_CH_MSK, A_CON_HDLC, A_HDLC_PAR, A_CHANNEL A_FIFO_DATA, A_FIFO_DATA_NOINC and R_RAM_DATA) have a idle time

$$t_{IDLE} \geq 5 \cdot t_{CLKI}$$

between ⌟ of t_{WR} (end of write access) and the next ⌞ of t_{WR} (start of write access) .

Hint !

If the same register as in the last register read / write access is used again, the register address write is not required.

2.3.3 Register read access in mode 4 (Intel, multiplexed)

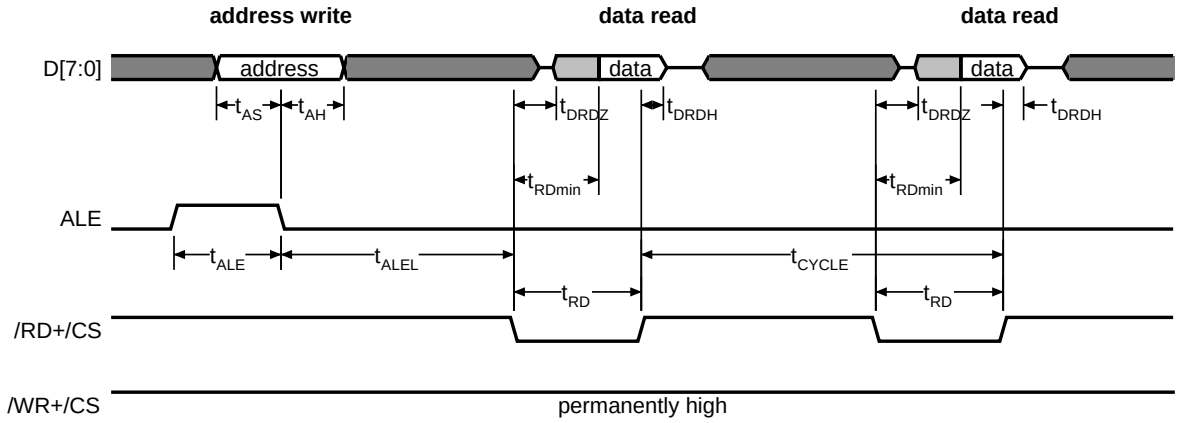


Figure 2.3: *Read access in mode 4 (Intel, multiplexed)*

Table 2.6: *Symbols of read accesses in Figure 2.3*

Symbol	min / ns	max / ns	Characteristic
t_{ALE}	10		Address latch time
t_{ALEL}	0		ALE ⎿ to /RD+/CS ⎿
t_{AS}	10		Address valid to ALE ⎿ setup time
t_{AH}	10		Address hold time after ALE ⎿
t_{DRDZ}	3		/RD+/CS ⎿ to data buffer turn on time
t_{DRDH}	2	15	/RD+/CS ⎍ to data buffer turn off time
t_{RD}			Read time:
	$2 \cdot t_{CLKI}$		address bit [7] = '0' (address range 0 ... 0x7F: normal register access)
	20		address bits [7,6] = '10' (address range 0x80 ... 0xBF: FIFO data access)
	$5 \cdot t_{CLKI}$		address bits [7,6] = '11' (address range 0xC0 ... 0xFF: direct internal RAM access) *
t_{CYCLE}			Cycle time between two consecutive /RD+/CS ⎍
			address bits [7] = '0' (address range 0 ... 0x7F: normal register access)
	$6 \cdot t_{CLKI}$		address bits [7,6] = '10' (address range 0x80 ... 0xBF: FIFO data access)
	$6 \cdot t_{CLKI}$		address bits [7,6] = '11' (address range 0xC0 ... 0xFF: direct internal RAM access) *

*: Normally this time needs not to be matched because a direct RAM access is not needed.

t_{CLKI} is the CLKI clock period which is normally 40.69 ns (24.576 MHz system clock).

Important !

A0 must be '0' during the whole register read cycle. It should be connected to GND in mode 4.

2.3.4 Register write access in mode 4 (Intel, multiplexed)

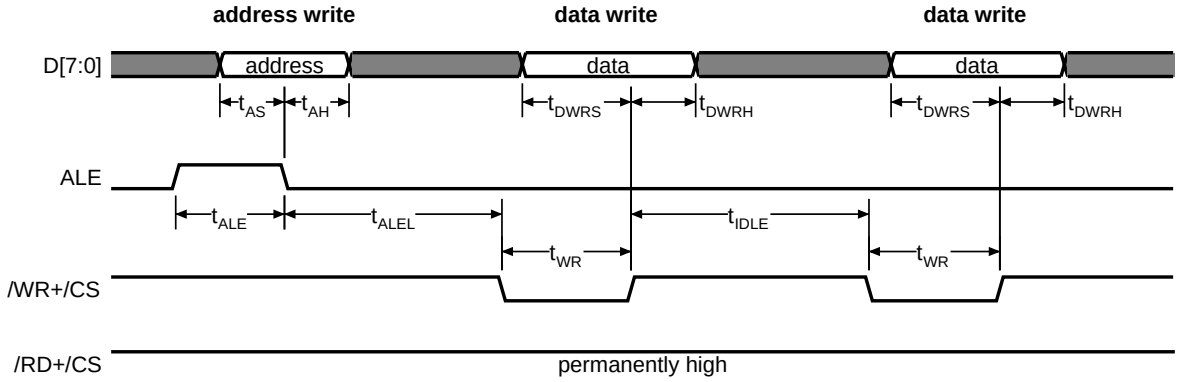


Figure 2.4: *Write access in mode 4 (Intel, multiplexed)*

Table 2.7: *Symbols of write accesses in Figure 2.4*

Symbol	min / ns	max / ns	Characteristic
t_{ALE}	10		Address latch time
t_{ALEL}	0		ALE ⅂ to /WR+/CS ⅂
t_{AS}	10		Address valid to ALE ⅂ setup time
t_{AH}	10		Address hold time after /WR+/CS ⅃
t_{DWRS}	20		Write data setup time to /WR+/CS ⅃
t_{DWRH}	10		Write data hold time from /WR+/CS ⅃
t_{WR}	20		Write time
t_{IDLE}			/WR+/CS high time between two data accesses
	20		address bit [7] = '0' (address range 0 ... 0x7F: normal register access)
	$5 \cdot t_{CLKI}$		address bits [7,6] = '10' (address range 0x80 ... 0xBF: FIFO data access)
	$5 \cdot t_{CLKI}$		address bits [7,6] = '11' (address range 0xC0 ... 0xFF)

t_{CLKI} is the CLKI clock period which is normally 40.69 ns (24.576 MHz system clock).

> **Important !**
>
> A0 must be '0' during the whole register read cycle. It should be connected to GND.

2.4 Examples of microprocessor connection circuitries

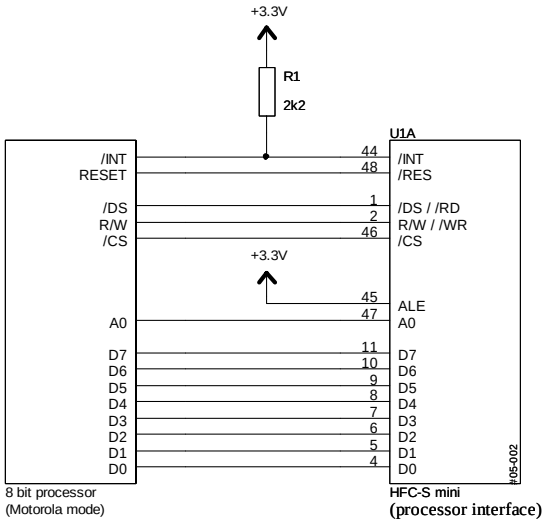


Figure 2.5: *8 bit Motorola processor circuitry example (mode 2)*

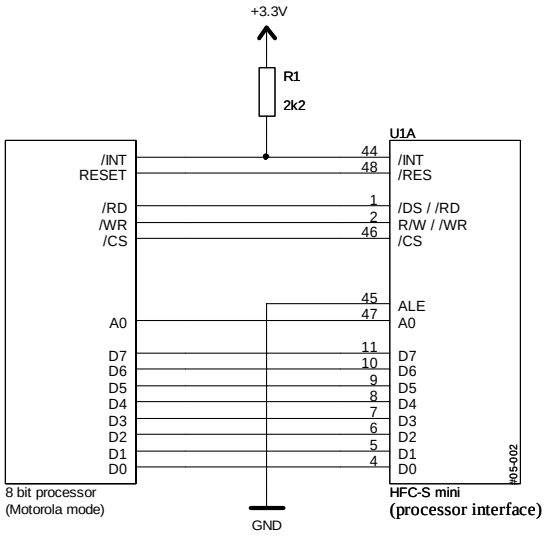


Figure 2.6: *8 bit Intel processor circuitry example (mode 3, non-multiplexed)*

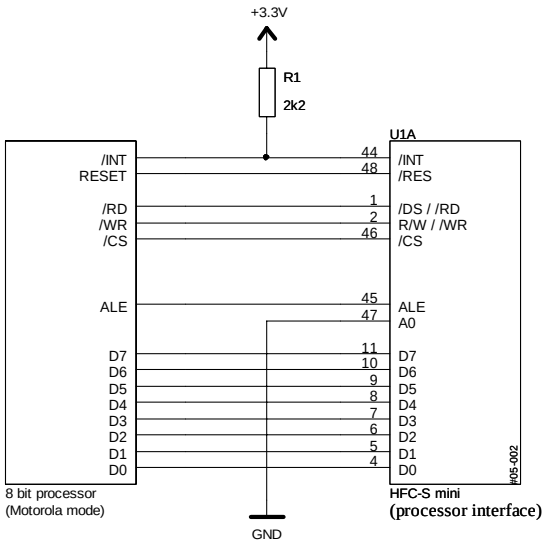


Figure 2.7: *8 bit Intel processor circuitry example (mode 4, multiplexed)*

2.5 Register description

> **Please note !**
>
> Due to a thorough revision of the HFC-S mini data sheet, some registers had to be renamed. Please see remarks on page 15.

2.5.1 Write only registers

R_CIRM	(w)	0x00

Reset register

Bits	Reset value	Name	Description
2..0	0	(reserved)	Must be '000'.
3	0	**V_SRES**	**Soft reset** The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
7..4	0	(reserved)	Must be '0000'.

R_RAM_ADDR0 (formerly **RAM_ADR_L**)	(w)	0x08

Low address byte for direct RAM access

Direct access to the internal RAM can be performed with registers R_RAM_ADDR0, R_RAM_ADDR1 and R_RAM_DATA. This function is normally not used.

Bits	Reset value	Name	Description
7..0	0	**V_RAM_ADDR0**	**Address bits 7..0** Address bits 10..8 must be specified in register R_RAM_ADDR1.

R_RAM_ADDR1 **(w)** 0x09
(formerly **RAM_ADR_H**)

High address bits for direct RAM access and access configuration

Direct access to the internal RAM can be performed with registers R_RAM_ADDR0, R_RAM_ADDR1 and R_RAM_DATA. This function is normally not used.

Bits	Reset value	Name	Description
2..0	0	**V_RAM_ADDR1**	**Address bits 10..8**
5..3	0	(reserved)	Must be '000'.
6	0	**V_ADDR_RES**	**Address reset** '0' = normal operation '1' = address bits 10..0 are set to zero This bit is automatically cleared.
7	0	**V_ADDR_INC**	**Address increment** '0' = no address increment '1' = automatic increment of the address after every write or read access to register R_RAM_DATA

2.5.2 Read only register

R_CHIP_ID (formerly **CHIP_ID**)	**(r)**	0x16

Chip identification register

Bits	Reset value	Name	Description
3..0	0	(reserved)	
7..4		**V_CHIP_ID**	**Chip identification code** '0101' means HFC-S mini.

2.5.3 Read / write register

R_RAM_DATA (formerly **RAM_DATA**)	**(r/w)**	0xC0

SRAM data byte

Direct access to the internal RAM can be performed with registers R_RAM_ADDR0, R_RAM_ADDR1 and R_RAM_DATA. This function is normally not used.

Bits	Reset value	Name	Description
7..0		**V_RAM_DATA**	**SRAM data access** The address must be written into the registers R_RAM_ADDR0 and R_RAM_ADDR1 in advance. FIFOs should be disabled before accessing the RAM directly.

Chapter 3

HFC-S mini data flow

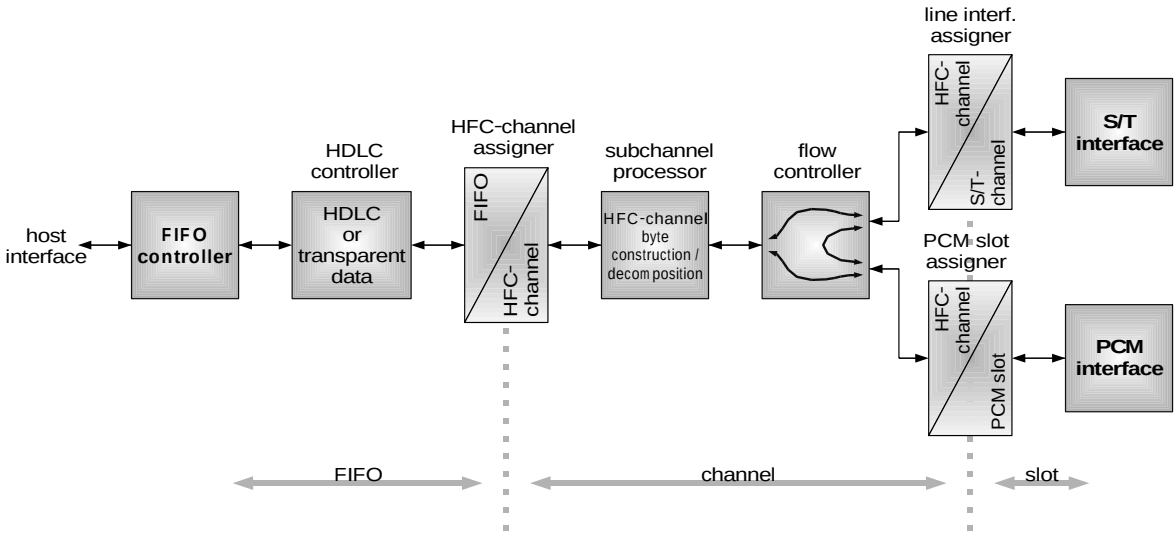


Figure 3.1: *Data flow block diagram*

3.1 Data flow concept

3.1.1 Overview

The HFC-S mini has a programmable data flow unit, in which the FIFOs are connected to the PCM and the S/T interface. Moreover, the data flow unit can directly connect PCM and S/T interface or two PCM time slots with each other [1].

The fundamental features of the HFC-S mini data flow are as follows:

- Programmable interconnection capability exists between FIFOs, PCM time slots and S/T-channels.
- In transmit and receive direction there are
  - 4 FIFOs each,
  - 32, 64 or 128 PCM time slots each and
  - 4 HFC-channels each to connect the above-mentioned data interfaces.
- Two data flow modes are available to satisfy different application tasks.
- Subchannel processing can be used for bitwise data handling.

The complete HFC-S mini data flow block diagram is shown in Figure 3.1.

FIFO handling and HDLC controller, the PCM and the S/T interface 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 S/T interface assigner.

3.1.2 Term definitions

Figure 3.2 clarifies the relationship and the differences between the numbering of FIFOs and PCM time slots as well as the naming of HFC-channels and the S/T-channels. The inner circle symbolizes the HFC-channel oriented part of the data flow, while the outer circle shows the connections of three data sources and data drains respectively.

FIFO: The FIFOs are buffers between the microprocessor bus interface and the PCM and S/T interface. The HDLC controllers are located on the non host bus side of the FIFOs. The FIFOs are numbered 0 . . 3. Furthermore, data directions transmit and receive are associated with every FIFO number.

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.

S/T-channels: The four S/T-channels are named B1-, B2, D- and E-channel and exist in both data directions each. [2]

HFC-channel: HFC-channels are used to define data paths between FIFOs on the one side and PCM and S/T interface on the other side. The HFC-channels are named following the S/T-channels as there is a fixed connection between every S/T-channel and a HFC-channel.

Each FIFO, HFC-channel and PCM time slot exist for transmit and receive direction. The data rate is always 8 kByte/s for every S/T-channel and every PCM time slot. FIFOs, HFC-channels, S/T-channel

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

[2] The transmit E-channel is handled automatically in NT mode. So there is no E-channel in transmit direction.

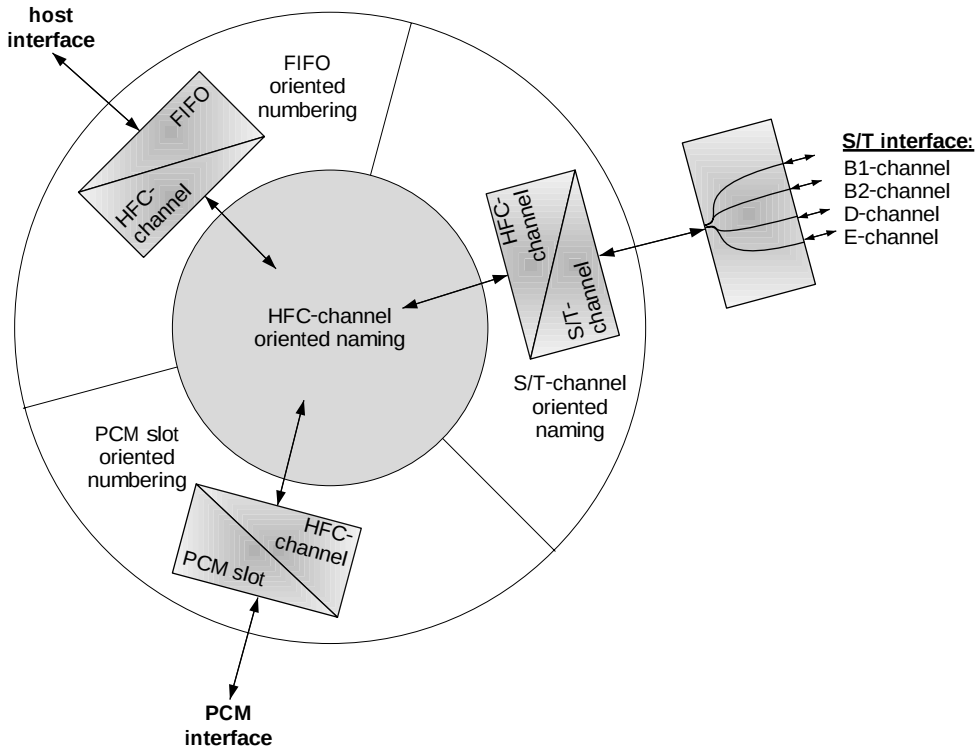


Figure 3.2: *Areas of FIFO, HFC-channel and PCM time slot oriented numbering*

and PCM time slots have always a width of 8 bit.

3.2 Flow controller

3.2.1 Overview

The various connections between FIFOs, S/T-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 S/T interface, FIFOs and the PCM interface and between the S/T interface and the PCM interface. The bitmap V_DATA_FLOW of the register A_CON_HDLC (which exists for each FIFO) configures these connections. The connection 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 / name 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 3.3.2.

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 S/T and PCM interface 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 from or received to the S/T and PCM interfaces. With every 125 μs cycle the switching buffers are swaped.

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-S/T connection delays each data byte for 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:

1. FIFO[0,TX]
2. FIFO[0,RX]
3. FIFO[1,TX]
4. FIFO[1,RX]
5. FIFO[2,TX]
6. FIFO[2,RX]
7. FIFO[3,TX]
8. FIFO[3,RX]

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.

3.2.4 Transmit FIFO operation (FIFO in transmit data direction)

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

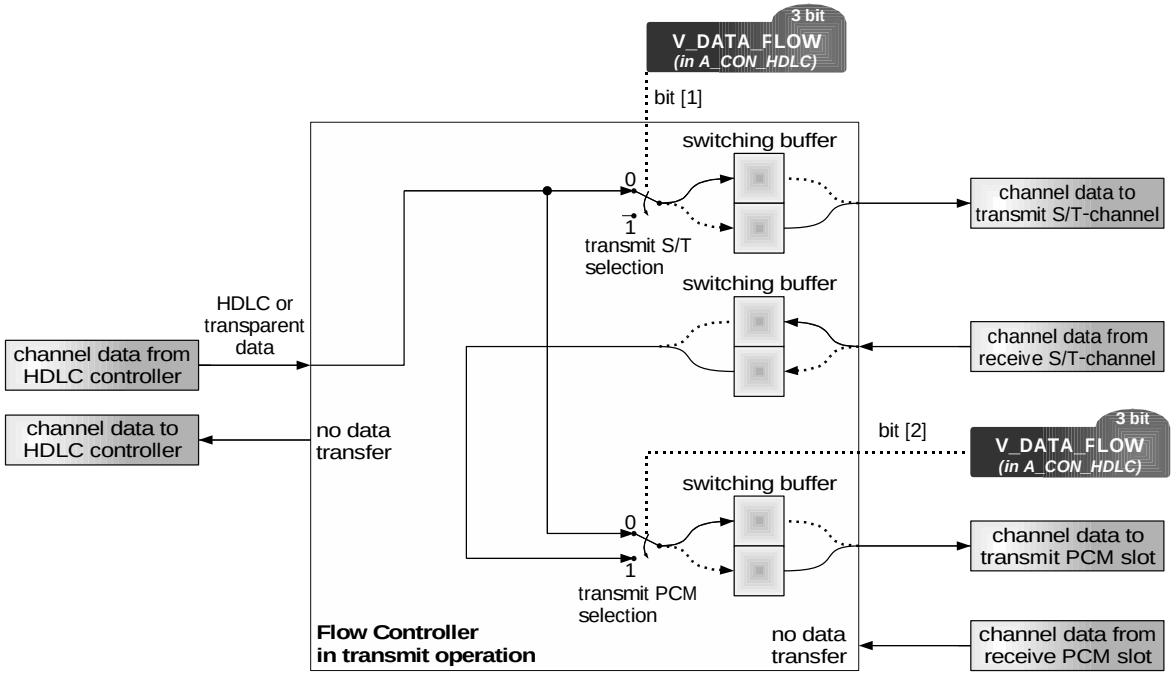


Figure 3.3: *The flow controller in transmit FIFO operation*

- FIFO data is only transmitted to the S/T interface if V_DATA_FLOW[1] = 0.
- The PCM interface can transmit a data byte which comes either from the FIFO or from the S/T interface. Bit V_DATA_FLOW[2] selects the source for the PCM transmit slot (see Figure 3.3). The receiving S/T-channel has always the same number as the transmitting S/T-channel.
- The bit V_DATA_FLOW[0] is ignored in transmit FIFO operation.

3.2.5 Receive FIFO operation (FIFO in receive data direction)

Figure 3.4 shows the flow controller structure in receive FIFO operation. The two switches are controlled by the bitmap V_DATA_FLOW[1 . . 0] of the register A_CON_HDLC[*n*,RX] where *n* is a FIFO number. Receive FIFO operation is configured with V_FIFO_DIR = 1 in register R_FIFO. FIFO data can either be received from the S/T or PCM interface. Furthermore, data can be transmitted from the PCM interface to the S/T interface.

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

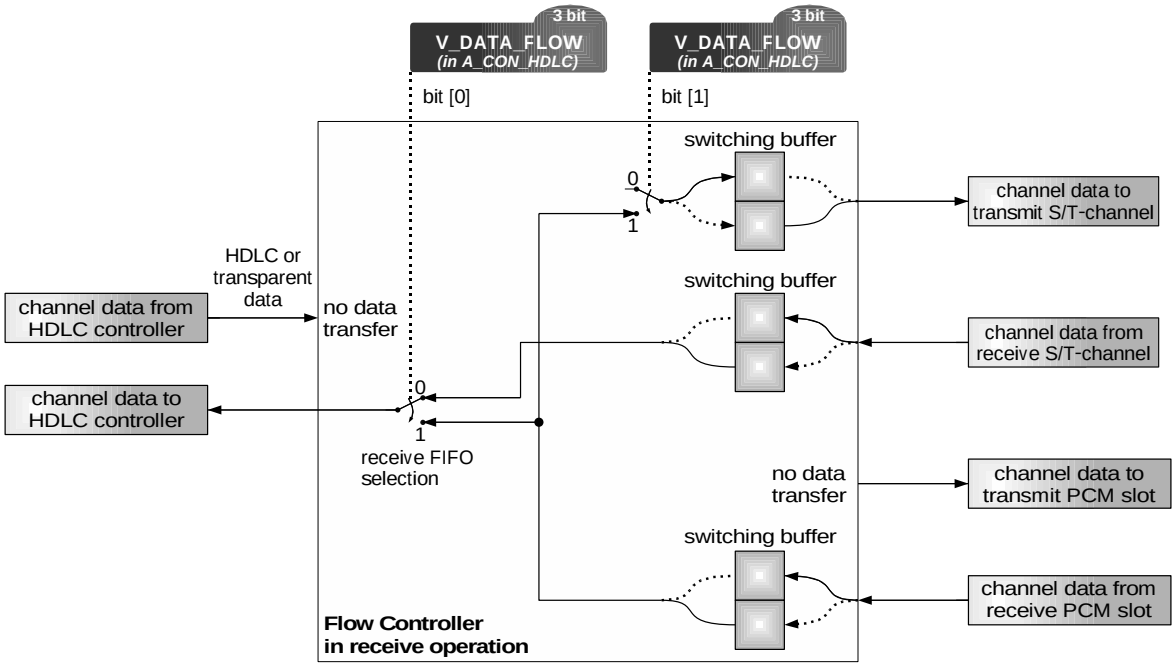


Figure 3.4: *The flow controller in receive FIFO operation*

3.2.6 Connection summary

Table 3.1 shows the flow controller connections as a whole. Bidirectional connections [3] 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.

Table 3.1: *Flow controller connectivity*

V_DATA_FLOW	Transmit (V_FIFO_DIR = 0)		Receive (V_FIFO_DIR = 1)	
'000'	FIFO → PCM	FIFO → S/T	FIFO ← S/T	
'001'	FIFO → S/T	FIFO → PCM	FIFO ← PCM	
'010'	FIFO → PCM		FIFO ← S/T	S/T ← PCM
'011'		FIFO → PCM	FIFO ← PCM	S/T ← PCM
'100'	S/T → PCM	FIFO → S/T	FIFO ← S/T	
'101'	FIFO → S/T	S/T → PCM	FIFO ← PCM	
'110'		S/T → PCM	S/T ← PCM	FIFO ← S/T
'111'		S/T → PCM	S/T ← PCM	FIFO ← PCM

[3] 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.

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.

Table 3.2: *V_DATA_FLOW programming values for bidirectional connections*

Connection			V_FIFO_DIR	Required V_DATA_FLOW	Recommended V_DATA_FLOW
FIFO	→	S/T	'0' (TX)	'x0x'	'000'
FIFO	←	S/T	'1' (RX)	'xx0'	
FIFO	→	PCM	'0' (TX)	'0xx'	'001'
FIFO	←	PCM	'1' (RX)	'xx1'	
S/T	→	PCM	'0' (TX)	'1xx'	'110'
S/T	←	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, S/T-channel 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 every HFC-channel to an arbitrary PCM time slot within the selected time slot block. Block selection is done with bitmap V_SL_BL in register R_PCM_MD2 as shown in Table 3.3.

Table 3.3: *Selection of PCM time slot block*

		available with		
V_SL_BL	**PCM time slots**	**PCM 30**	**PCM 64**	**PCM 128**
'00'	0..31	yes	yes	yes
'01'	32..63	no	yes	yes
'10'	64..95	no	no	yes
'11'	96..131	no	no	yes

The HFC-channel[B1,TX] is connected to PCM slot[50,TX], e.g., with the following register setup:

Register setup:			
R_PCM_MD2	: V_SL_BL = 1		select time slot block 1 (slots 32..63)
R_B1_TX_SL	: V_B1_TX_SL = 18		connect to time slot 50 (V_SL_BL ·32+ V_B1_TX_SL)

This example assumes that either PCM 64 or PCM 128 is chosen.

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

Two PCM time slots can be connected to each other. The following example shows how to set up a PCM loop with HFC-channel[AUX1] which connects PCM slot[50,RX] to slot[52,TX], e.g.:

Register setup:			
R_PCM_MD2	: V_SL_BL = 1		select time slot block 1 (slots 32..63)
R_AUX1_RX_SL	: V_AUX1_RX_SL = 18		connect to time slot 50 (V_SL_BL ·32+ V_AUX1_RX_SL)
R_AUX1_TX_SL	: V_AUX1_TX_SL = 20		connect to time slot 52 (V_SL_BL ·32+ V_AUX1_TX_SL)
R_PCM_MD1	: V_AUX1_MIR = 1		enable mirroring from HFC-channel[AUX1,RX] to HFC-channel[AUX1,TX]

HFC-channel[AUX2] can be used for a PCM loop in the same way. The mirroring of this channel must be enabled with V_AUX2_MIR = '1' in register R_PCM_MD1.

3.3.3 S/T interface assigner

Table 3.4 shows the assignment between HFC-channels and the S/T-channel. There is no possibility to change this allocation, so there is no register for programming the S/T interface assigner.

Table 3.4: *S/T interface assigner*

HFC-channel		S/T-channel	
name	direction	channel	direction
[B1,TX]		B1	TX
[B1,RX]		B1	RX
[B2,TX]		B2	TX
[B2,RX]		B2	RX
[AUX1,TX]		D	TX
[AUX1,RX]		D	RX
[AUX2,TX]		–	TX
[AUX2,RX]		E	RX

3.4 Data flow modes

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

- *Simple Mode* (SM) and
- *Channel Select Mode* (CSM).

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

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 S/T-channel or one PCM time slot can be connected to each other.

Simple Mode is selected with V_CSM = '0' in register R_DF_MD. The HFC-channel assigner is not programmable in this mode and has the fixed connections as shown in Table 3.5.

Table 3.5: *HFC-channel assigner in* Simple Mode

FIFO		HFC-channel	
number	direction	name	direction
[0,TX]		B1	TX
[0,RX]		B1	RX
[1,TX]		B2	TX
[1,RX]		B2	RX
[2,TX]		AUX1	TX
[2,RX]		AUX1	RX
[3,TX]		AUX2	TX
[3,RX]		AUX2	RX

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

3.4.1.2 Subchannel processing

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 the 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 on page 66 for details concerning the subchannel processor.

3.4.1.3 Example for SM

Figure 3.5 shows an example with four bidirectional connections (❶ FIFO-to-S/T, ❷ FIFO-to-PCM, ❸ PCM-to-S/T and ❹ PCM-to-PCM). The FIFO box on the left side contains the number and direction information of the used FIFOs. The S/T and PCM boxes on the right side contain the S/T-channels and PCM time slots 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 3.5 are configured in bidirectional data pathes due to typical applications of the HFC-S mini. 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 either V_HDLC_TRP = 1 (transparent mode and implicit FIFO enable) or V_TRP_IRQ $\neq 0$ (explicit FIFO enable) in register A_CON_HDLC[FIFO].

❶ FIFO-to-S/T

As HFC-channel and FIFO numbers have a fixed connection in SM, a selected S/T-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-S/T connection, the desired S/T-channel has to be chosen and the linked FIFO (see Table 3.4) has to be programmed. Due to the user's requirements, V_FIFO0_TX_REV and V_FIFO0_RX_REV in register R_FIFO_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.

If HDLC mode is chosen, the FIFO must be enabled with V_TRP_IRQ $\neq 0$.

Register setup:				(SM ❶ TX)
R_FIFO	: V_FIFO_DIR	= 0		(transmit FIFO)
	: V_FIFO_NUM	= 0		(FIFO #0)
A_CON_HDLC[0,TX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0		(HDLC mode)
	: V_TRP_IRQ	= 1		(enable FIFO)
	: V_DATA_FLOW	= '000'		(FIFO → S/T, FIFO → PCM)

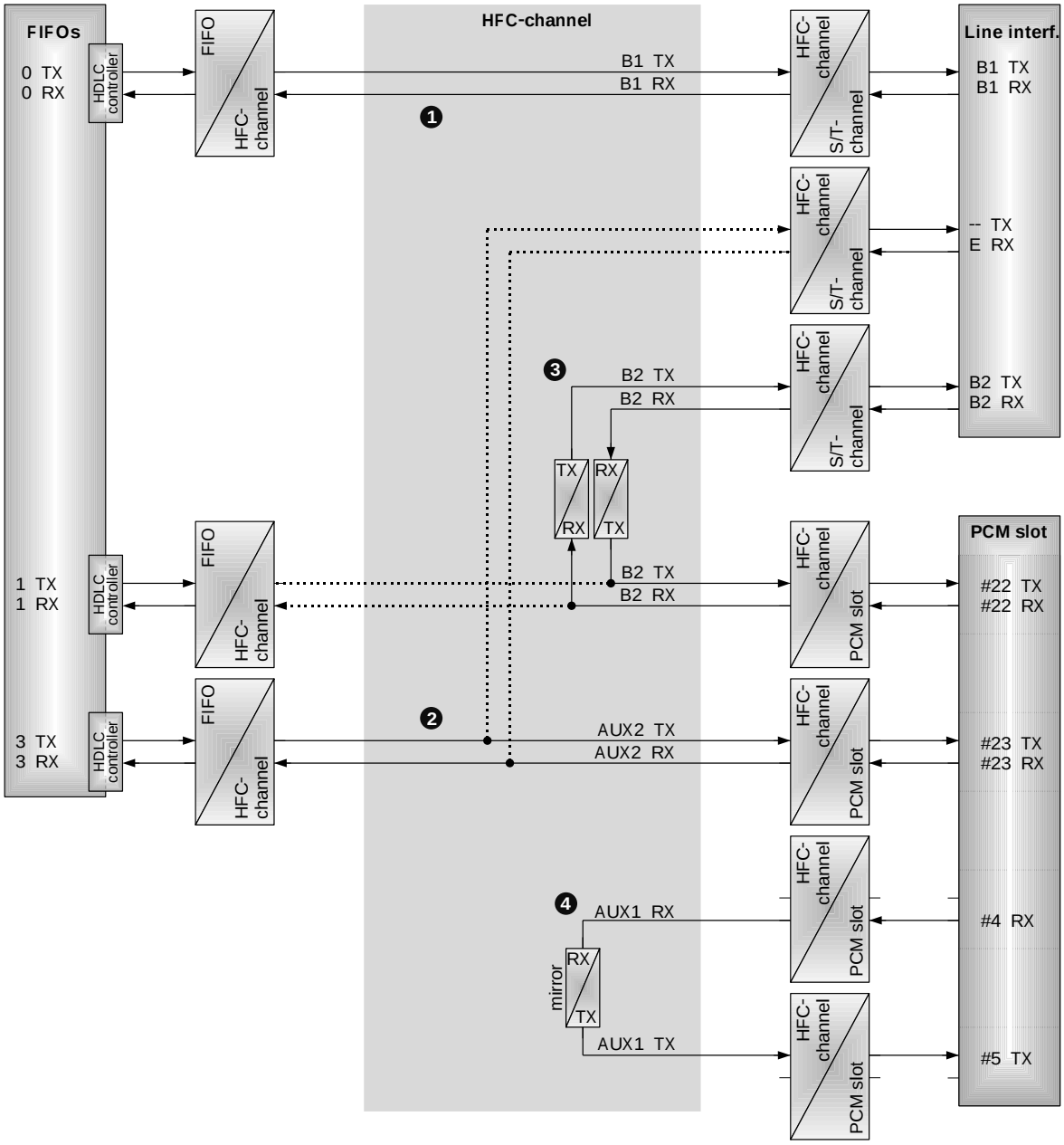


Figure 3.5: *SM example*

Register setup:				(SM ❶ RX)
R_FIFO	: V_FIFO_DIR	= 1	(receive FIFO)	
	: V_FIFO_NUM	= 0	(FIFO #0)	
A_CON_HDLC[0,RX]	: V_IFF	= 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP	= 0	(HDLC mode)	
	: V_TRP_IRQ	= 1	(enable FIFO)	
	: V_DATA_FLOW	= '000'	(FIFO ← S/T)	

❷ FIFO-to-PCM

The FIFO-to-PCM connection has a programmable connection between the involved HFC-channels and PCM time slots.

As the S/T interface assigner links the HFC-channels to the S/T-channels, every used HFC-channel blocks the connected S/T-channel. In this example the E-channel of the S/T interface is blocked in both transmit and receive directions.

Again, V_FIFO3_TX_REV, V_FIFO3_RX_REV and V_HDLC_TRP can freely be chosen according to the user's requirements. As in the previous setting, a periodic interrupt in transperant 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	= 3	(FIFO #3)	
A_CON_HDLC[3,TX]	: V_IFF	= 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP	= 0	(HDLC mode)	
	: V_TRP_IRQ	= 1	(enable FIFO)	
	: V_DATA_FLOW	= '001'	(FIFO → S/T, FIFO → PCM)	
R_AUX2_TX_SL	: V_AUX2_TX_SL	= 23	(slot #23)	
	: V_AUX2_TX_ROUT	= '10'	(data to pin STIO1)	

Register setup:				(SM ❷ RX)
R_FIFO	: V_FIFO_DIR	= 1	(receive FIFO)	
	: V_FIFO_NUM	= 3	(FIFO #3)	
A_CON_HDLC[3,RX]	: V_IFF	= 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP	= 0	(HDLC mode)	
	: V_TRP_IRQ	= 1	(enable FIFO)	
	: V_DATA_FLOW	= '001'	(FIFO ← PCM)	
R_AUX2_RX_SL	: V_AUX2_RX_SL	= 23	(slot #23)	
	: V_AUX2_RX_ROUT	= '10'	(data from pin STIO2)	

❸ PCM-to-S/T

A direct PCM-to-S/T coupling is shown in the third connection set. The array registers of FIFO[1,TX] and FIFO[1,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 either

V_HDLC_TRP = 1 (transparent mode and implicit FIFO enable) or V_TRP_IRQ $\neq$ 0 (explicit FIFO enable) 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 S/T interface neither in receive nor in transmit direction. A PCM-to-S/T connection always operates in transparent mode.

For a PCM-to-S/T 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, S/T 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	=	1	(FIFO #1)
A_CON_HDLC[1,TX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	1	(transparent mode)
	: V_TRP_IRQ	=	0	(interrupt disabled)
	: V_DATA_FLOW	=	'110'	(S/T → PCM)
R_B2_TX_SL	: V_B2_TX_SL	=	22	(slot #22)
	: V_B2_TX_ROUT	=	'10'	(data to pin STIO1)

Register setup:				(SM ❸ RX)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	1	(FIFO #1)
A_CON_HDLC[1,RX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	1	(transparent mode)
	: V_TRP_IRQ	=	0	(interrupt disabled)
	: V_DATA_FLOW	=	'110'	(FIFO ← S/T, S/T ← PCM)
R_B2_TX_SL	: V_B2_TX_SL	=	22	(slot #22)
	: V_B2_TX_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[AUX1] is used to connect PCM slot[4,RX] to PCM slot[5,TX].

Register setup:				(SM ❹ loop
R_AUX1_RX_SL :	V_AUX1_RX_SL	= 4		(assign PCM slot[4,RX])
:	V_AUX1_RX_ROUT	= '11'		(receive data from STIO1)
R_AUX1_TX_SL :	V_AUX1_TX_SL	= 5		(assign PCM slot[5,TX])
:	V_AUX1_TX_ROUT	= '11'		(transmit data to STIO2)
R_PCM_MD1 :	V_AUX1_MIR	= 1		(enable mirroring for HFC-channel[AUX1])

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 3.6 (left side). Beyond this, it is possible to connect several FIFOs to one HFC-channel (Fig. 3.6, right side). This works in transmit and receive direction and can be used to connect one 8 kByte/s S/T-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 3.6: *HFC-channel assigner in CSM*

Channel Select Mode is selected with V_CSM = '1' in register R_DF_MD.

3.4.2.2 HFC-channel assigner

The connection between a FIFO and a HFC-channel can be established by the A_CHANNEL register which exists for every FIFO. For a selected FIFO, the HFC-channel to be connected must be written to V_CH_NUM of the register A_CHANNEL. Typically, the data direction in V_CH_DIR is the same as the FIFO data direction V_FIFO_DIR in 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_DIR	=	V_FIFO_DIR
:	V_CH_NUM	=	n

A direct connection between a PCM time slot and an S/T-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_NUM 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_CH_MSK.

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 66 for details concerning the subchannel processor.

3.4.2.4 Example for CSM

The example for a *Channel Select Mode* configuration in Figure 3.7 shows three bidirectional connections (❶ FIFO-to-S/T, ❷ FIFO-to-PCM and ❸ PCM-to-S/T). 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.

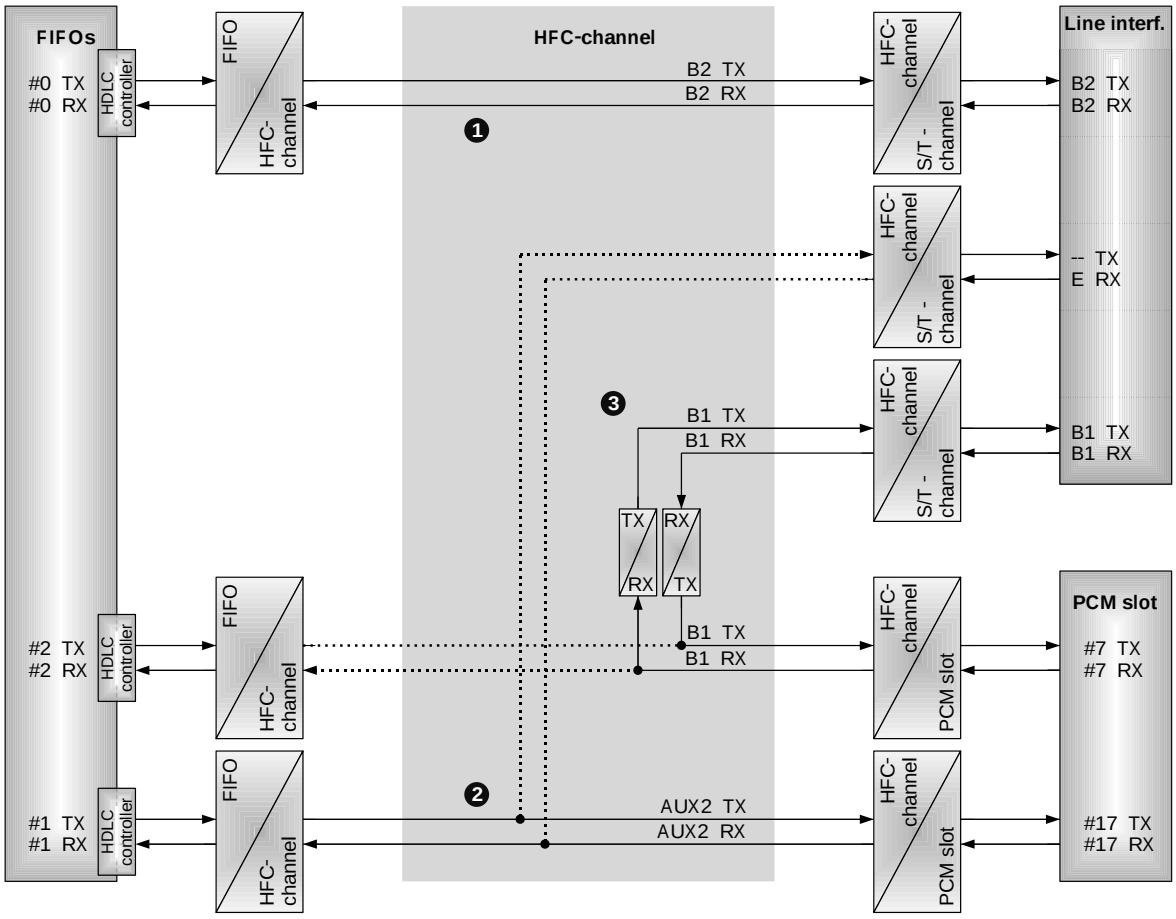


Figure 3.7: *CSM example*

The following settings demonstrate the required register values to establish the connections. All involved FIFOs have to be enabled with either V_HDLC_TRP = 1 (transparent mode and implicit

FIFO enable) or V_TRP_IRQ $\neq 0$ (explicit FIFO enable) in register A_CON_HDLC[FIFO].

❶ FIFO-to-S/T

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

Due to the user's requirements, R_FIFO_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.

If HDLC mode is chosen, the FIFO must be enabled with V_TRP_IRQ $\neq 0$.

Register setup:				(CSM ❶ TX)
R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	0	(FIFO #0)
A_CON_HDLC[0,TX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	0	(HDLC mode)
	: V_TRP_IRQ	=	1	(enable FIFO)
	: V_DATA_FLOW	=	'000'	(FIFO → S/T, FIFO → PCM)
A_CHANNEL[0,TX]	: V_CH_DIR	=	0	(transmit HFC-channel)
	: V_CH_NUM	=	1	(HFC-channel[B2])

Register setup:				(CSM ❶ RX)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	0	(FIFO #0)
A_CON_HDLC[0,RX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	0	(HDLC mode)
	: V_TRP_IRQ	=	1	(enable FIFO)
	: V_DATA_FLOW	=	'000'	(FIFO ← S/T)
A_CHANNEL[0,RX]	: V_CH_DIR	=	1	(receive HFC-channel)
	: V_CH_NUM	=	1	(HFC-channel[B2])

❷ FIFO-to-PCM

The FIFO-to-PCM connection blocks one transmit and one receive S/T-channel. In this example, HFC-channels that are assigned to E-channels are used.

Again, R_FIFO_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_TRP_IRQ = 1.

Register setup:				(CSM ❷ TX)
R_FIFO	: V_FIFO_DIR	= 0		(transmit FIFO)
	: V_FIFO_NUM	= 1		(FIFO #1)
A_CON_HDLC[1,TX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0		(HDLC mode)
	: V_TRP_IRQ	= 1		(enable FIFO)
	: V_DATA_FLOW	= '001'		(FIFO → S/T, FIFO → PCM)
A_CHANNEL[1,TX]	: V_CH_DIR	= 0		(transmit HFC-channel)
	: V_CH_NUM	= 3		(HFC-channel[AUX2])
R_AUX2_TX_SL	: V_AUX2_TX_SL	= 17		(PCM slot[17,TX])
	: V_AUX2_TX_ROUT	= '10'		(data to pin STIO1)

Register setup:				(CSM ❷ RX)
R_FIFO	: V_FIFO_DIR	= 1		(receive FIFO)
	: V_FIFO_NUM	= 1		(FIFO #1)
A_CON_HDLC[1,RX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0		(HDLC mode)
	: V_TRP_IRQ	= 1		(enable FIFO)
	: V_DATA_FLOW	= '001'		(FIFO ← PCM)
A_CHANNEL[1,RX]	: V_CH_DIR	= 1		(receive HFC-channel)
	: V_CH_NUM	= 3		(HFC-channel[AUX2])
R_AUX2_RX_SL	: V_AUX2_RX_SL	= 17		(PCM slot[17,RX])
	: V_AUX2_RX_ROUT	= '10'		(data from pin STIO2)

❸ PCM-to-S/T

The PCM-to-S/T connection blocks one transmit and one receive FIFO. Although there is no data stored in these FIFOs, they must be enabled to switch on the data transmission between the PCM and the S/T 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 S/T interface neither in receive nor in transmit direction. A PCM-to-S/T connection operates always in transparent mode.

For a PCM-to-S/T 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. [4]

[4] It is not forbidden to connect the blocked FIFOs at the S/T 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.

Register setup:				(CSM ❸ TX)
R_FIFO	: V_FIFO_DIR	= 0	(transmit FIFO)	
	: V_FIFO_NUM	= 2	(FIFO #2)	
A_CON_HDLC[2,TX]	: V_IFF	= 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP	= 1	(transparent mode)	
	: V_TRP_IRQ	= 0	(interrupt disabled)	
	: V_DATA_FLOW	= '110'	(S/T → PCM)	
A_CHANNEL[2,TX]	: V_CH_DIR	= 0	(transmit HFC-channel)	
	: V_CH_NUM	= 0	(HFC-channel[B1])	
R_B1_TX_SL	: V_B1_TX_SL	= 7	(PCM slot[7,TX])	
	: V_B1_TX_ROUT	= '10'	(data to pin STIO1)	

Register setup:				(CSM ❸ RX)
R_FIFO	: V_FIFO_DIR	= 1	(receive FIFO)	
	: V_FIFO_NUM	= 2	(FIFO #2)	
A_CON_HDLC[2,RX]	: V_IFF	= 0	(0x7E as inter frame fill)	
	: V_HDLC_TRP	= 1	(transparent mode)	
	: V_TRP_IRQ	= 0	(interrupt disabled)	
	: V_DATA_FLOW	= '110'	(FIFO ← S/T, S/T ← PCM)	
A_CHANNEL[2,RX]	: V_CH_DIR	= 1	(receive HFC-channel)	
	: V_CH_NUM	= 0	(HFC-channel[B1])	
R_B1_RX_SL	: V_B1_RX_SL	= 7	(PCM slot[7,RX])	
	: V_B1_RX_ROUT	= '10'	(data from pin STIO2)	

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 and
- every used PCM time slot requires one HFC-channel.

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 or CSM) 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-S/T 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 an simplified example in Figure 3.8. 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.

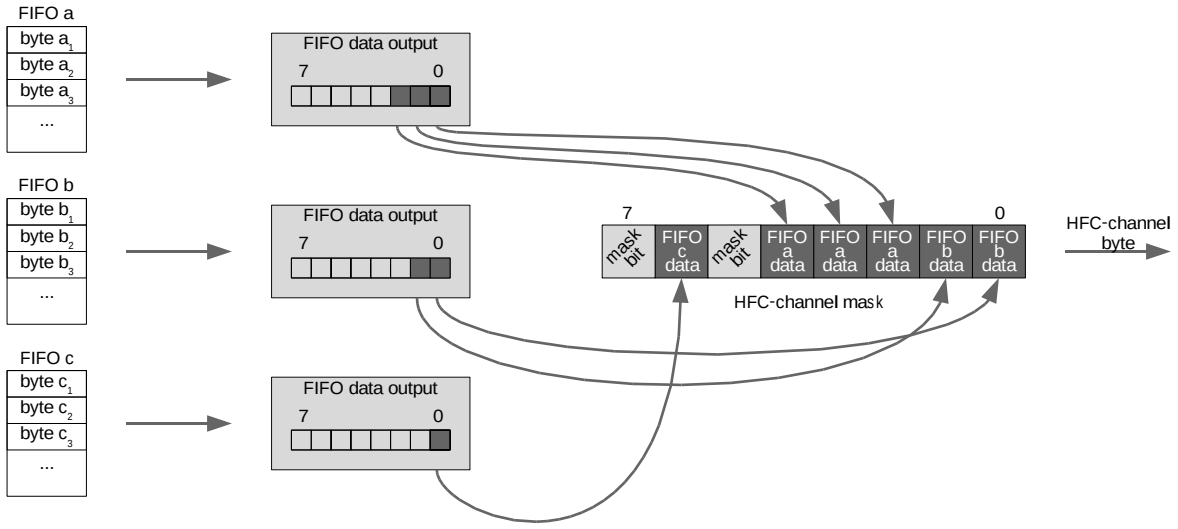


Figure 3.8: *General structure of the subchannel processor shown with an example of three connected FIFOs*

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. 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.

3.5.1.1 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_HDLC_PAR[FIFO].

The HFC-channel mask can be stored in register A_CH_MSK[FIFO]. This mask is only used for transmit data. The processed FIFO bits are stored in this register, so it must be re-initialized after changing the settings in A_HDLC_PAR[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 register R_FIFO first. The desired mask byte *m* can be written with A_CH_MSK = *m* after this index selection.

> **Important !**
>
> Typically, the 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 (V_FIFO_NUM: 0 = B1, 1 = B2, 2 = AUX1, 3 = AUX2).

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

> **Important !**
>
> For normal data transmission the register A_HDLC_PAR must be set to 0x00. To use 56 kbit/s restricted mode for U.S. ISDN lines the register A_HDLC_PAR must be set to 0x07 for B-channels.

3.5.2 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 3.9 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_HDLC_PAR[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.

3.5.2.1 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.

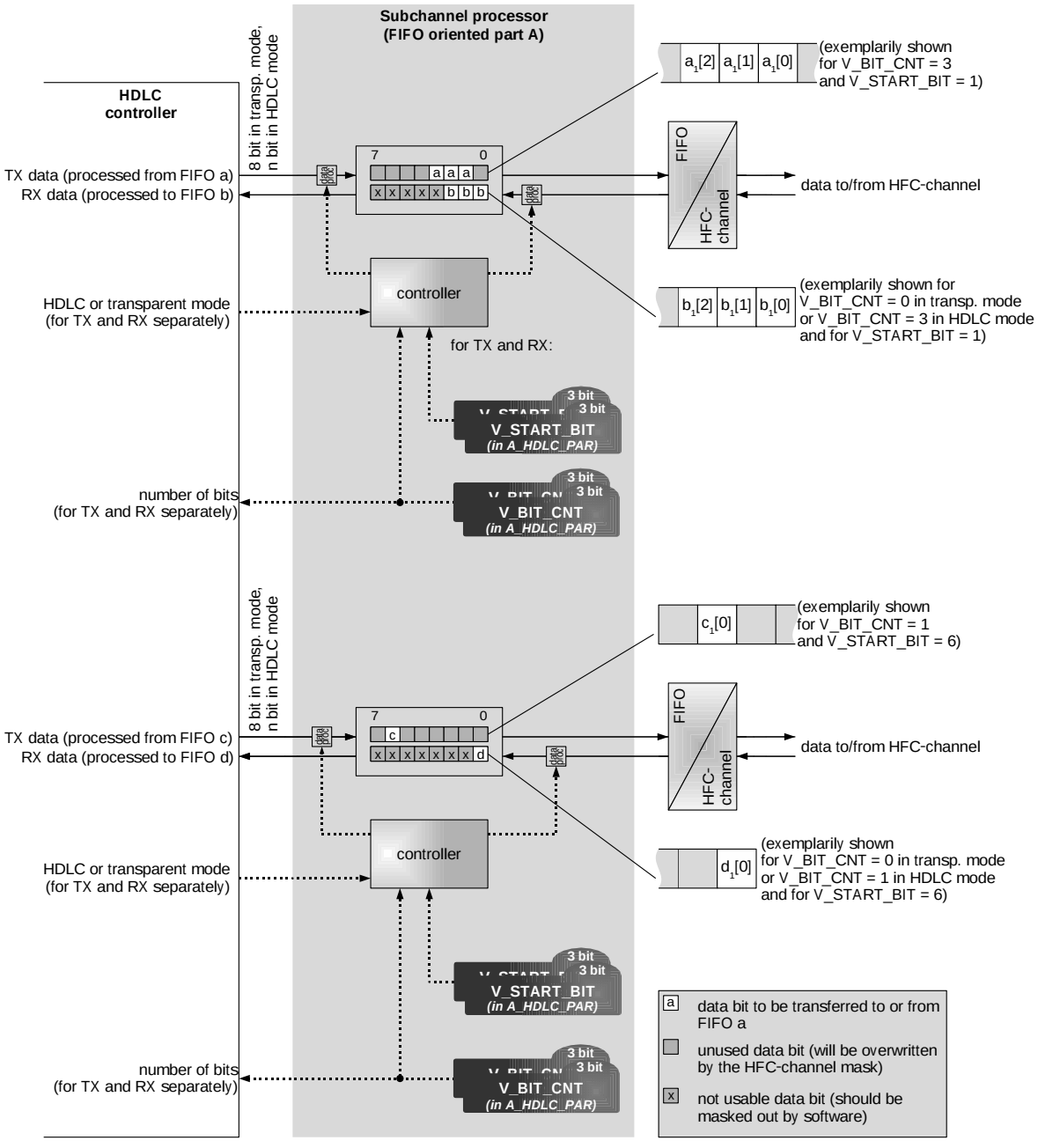


Figure 3.9: *Part A of the subchannel processor*

3.5.2.2 FIFO transmit operation in HDLC mode

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

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

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

3.5.2.3 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.

3.5.2.4 FIFO receive operation in HDLC mode

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

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

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

3.5.3 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 S/T interface). On the other side, to / from every connected HFC-channel assigner one byte is transferred in both transmit and receive directions. Figure 3.10 shows the block diagram of this module.

3.5.3.1 FIFO transmit operation in SM

As FIFOs and HFC-channels are not programmable 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.

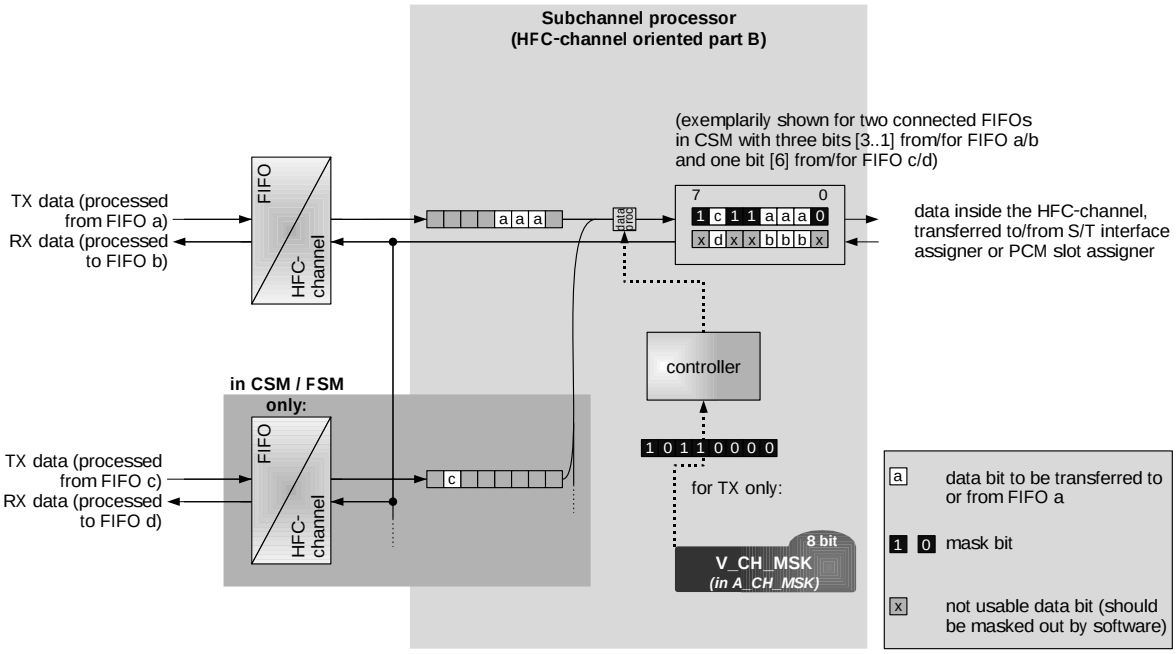


Figure 3.10: *Part B of the subchannel processor*

3.5.3.2 FIFO transmit operation in CSM

In *Channel Select 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.

3.5.3.3 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.

3.5.3.4 FIFO receive operation in CSM

If there are several FIFOs connected to one receive HFC-channel in *Channel Select 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.

3.5.4 Subchannel example for SM

The subchannel processing example in Figure 3.11 shows two bidirectional configurations (❶ FIFO-to-S/T 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.

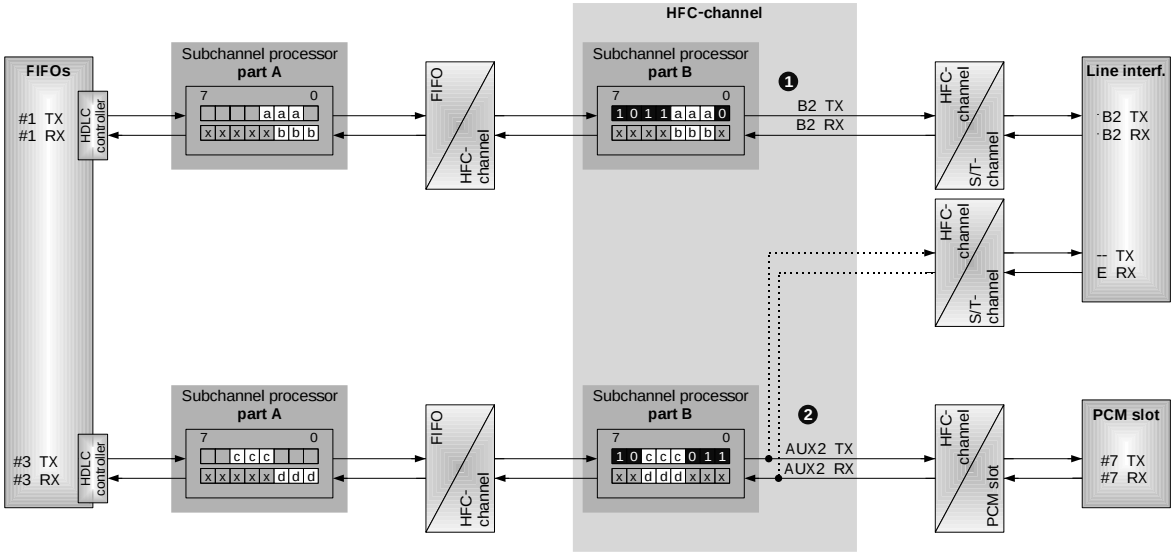


Figure 3.11: *SM example with subchannel processor*

❶ FIFO-to-S/T (TX)

The first setting shows a FIFO-to-S/T data transmission in transparent mode.

The register A_HDLC_PAR[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 the register R_FIFO. The mask bits [3 . . 1] are *don't care* because they are overwritten from the FIFO data.

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

Register setup:				(SM ❶ TX)
R_FIFO	: V_FIFO_DIR	= 0		(transmit FIFO)
	: V_FIFO_NUM	= 1		(FIFO #1)
A_CON_HDLC[1,TX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 1		(transparent mode)
	: V_TRP_IRQ	= 1		(interrupt all 16 bytes)
	: V_DATA_FLOW	= '000'		(FIFO → S/T, FIFO → PCM)
A_HDLC_PAR[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[B2])
A_CH_MSK[1,TX]	: V_CH_MSK	= '10110000'		(mask byte)

❶ FIFO-to-S/T (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_HDLC_PAR. 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)
A_CON_HDLC[1,RX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 1		(transparent mode)
	: V_TRP_IRQ	= 1		(interrupt all 16 bytes)
	: V_DATA_FLOW	= '000'		(FIFO ← S/T)
A_HDLC_PAR[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 [5]. The bitmap V_BIT_CNT in register A_HDLC_PAR[FIFO] defines three FIFO data bits to be transmitted during every 125 μs cycle. The bit field location in the HFC-channel data byte is specified by the bitmap V_START_BIT in the same register.

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

Table 3.6: *Subchannel processing according to Figure 3.11 (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 3.11 (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
...					...			

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 the 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[3,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	= 3		(FIFO #3)
A_CON_HDLC[3,TX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0		(HDLC mode)
	: V_TRP_IRQ	= 1		(enable FIFO)
	: V_DATA_FLOW	= '001'		(FIFO → S/T, FIFO → PCM)
A_HDLC_PAR[3,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	= 3		(HFC-channel[AUX2])
A_CH_MSK[3,TX]	: V_CH_MSK	= '10110011'		(mask byte)
R_AUX2_TX_SL	: V_AUX2_TX_SL	= 7		(PCM slot[7,TX])
	: V_AUX2_TX_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 configured with the bitmaps V_BIT_CNT = 3 and V_START_BIT = 3 in register A_HDLC_PAR. 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[3,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	= 3		(FIFO #3)
A_CON_HDLC[3,RX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0		(HDLC mode)
	: V_TRP_IRQ	= 1		(enable FIFO)
	: V_DATA_FLOW	= '001'		(FIFO ← PCM)
A_HDLC_PAR[3,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_AUX2_RX_SL	: V_AUX2_RX_SL	= 7		(PCM slot[7,RX])
	: V_AUX2_RX_ROUT	= '10'		(data from pin STIO2)

Table 3.8: *Subchannel processing according to Figure 3.11 (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
...				...				

Table 3.9: *Subchannel processing according to Figure 3.11 (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
...				...				

3.5.5 Subchannel example for CSM

In *Channel Select Mode* up to 4 FIFOs can be assigned to one HFC-channel. The example in Figure 3.12 shows two bidirectional configurations (❶ FIFO-to-S/T and ❷ FIFO-to-PCM) with two FIFOs per direction each.

❶ **FIFO-to-S/T (TX)**

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

The registers A_HDLC_PAR[FIFO] of FIFO[0,TX] and FIFO[1,TX] define both, the number of bits to be extracted from the FIFO data bytes and their position in the HFC-channel byte.

The HFC-channel mask in register A_CH_MSK defines the bit values that are not used for FIFO data. The array register must be selected by writing the HFC-channel into the 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[0,TX] is a_1, the second byte is a_2, and so on. FIFO[1,TX] is represented by the data bytes c_1, c_2, and so on.

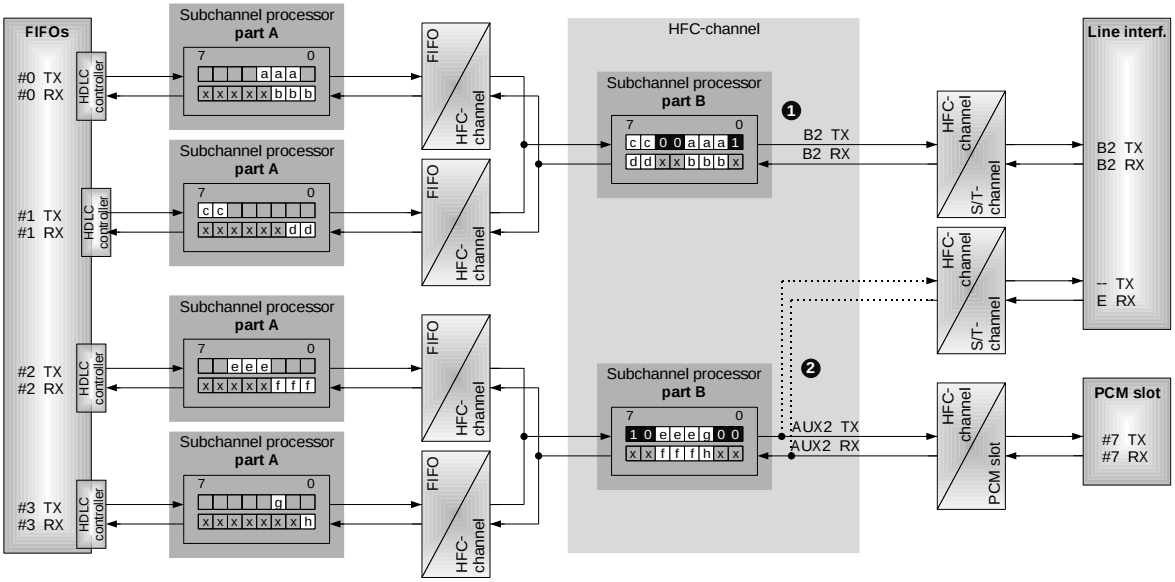


Figure 3.12: *CSM example with subchannel processor*

Register setup:				(CSM ❶ TX)
R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	0	(FIFO #0)
A_CON_HDLC[0,TX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	1	(transparent mode)
	: V_TRP_IRQ	=	1	(interrupt all 16 bytes)
	: V_DATA_FLOW	=	'000'	(FIFO → S/T, FIFO → PCM)
A_CHANNEL[0,TX]	: V_CH_DIR	=	0	(transmit HFC-channel)
	: V_CH_NUM	=	1	(HFC-channel[B2])
A_HDLC_PAR[0,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	=	1	(FIFO #1)
A_CON_HDLC[1,TX]	: V_IFF	=	0	(0x7E as inter frame fill)
	: V_HDLC_TRP	=	1	(transparent mode)
	: V_TRP_IRQ	=	1	(interrupt all 16 bytes)
	: V_DATA_FLOW	=	'000'	(FIFO → S/T, FIFO → PCM)
A_CHANNEL[1,TX]	: V_CH_DIR	=	0	(transmit HFC-channel)
	: V_CH_NUM	=	1	(HFC-channel[B2])
A_HDLC_PAR[1,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	=	0	(HFC-channel[B1])
A_CH_MSK[0,TX]	: V_CH_MSK	=	'0000 0001'	(mask byte)

❶ FIFO-to-S/T (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_HDLC_PAR. 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[0,RX] is b_1, the second byte is b_2, and so on. FIFO[1,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	= 0		(FIFO #0)
A_CON_HDLC[0,RX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 1		(transparent mode)
	: V_TRP_IRQ	= 1		(interrupt all 16 bytes)
	: V_DATA_FLOW	= '000'		(FIFO ← S/T)
A_CHANNEL[0,RX]	: V_CH_DIR	= 1		(receive HFC-channel)
	: V_CH_NUM	= 1		(HFC-channel[B2])
A_HDLC_PAR[0,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	= 1		(FIFO #1)
A_CON_HDLC[1,RX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 1		(transparent mode)
	: V_TRP_IRQ	= 1		(interrupt all 16 bytes)
	: V_DATA_FLOW	= '000'		(FIFO ← S/T)
A_CHANNEL[1,RX]	: V_CH_DIR	= 1		(receive HFC-channel)
	: V_CH_NUM	= 1		(HFC-channel[B2])
A_HDLC_PAR[1,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)

❷ FIFO-to-PCM (TX)

A FIFO-to-PCM configuration in HDLC mode with two FIFOs in transmit and receive direction each is shown in the second example setting [6].

The registers A_HDLC_PAR[FIFO] of FIFO[2,TX] and FIFO[3,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 the

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

Table 3.10: *Subchannel processing according to Figure 3.12 (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
...				...				

Table 3.11: *Subchannel processing according to Figure 3.12 (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
...				...				

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[2,TX] is e_1, the second byte is e_2, and so on. FIFO[3,TX] transmits bytes g_1, g_2, and so on. In HDLC mode, FIFO bytes are dispersed among several HFC-channel bytes.

Register setup:			(CSM ❷ TX)
R_FIFO	: V_FIFO_DIR	= 0	(transmit FIFO)
	: V_FIFO_NUM	= 2	(FIFO #2)
A_CON_HDLC[2,TX]	: V_IFF	= 0	(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0	(HDLC mode)
	: V_TRP_IRQ	= 1	(enable FIFO)
	: V_DATA_FLOW	= '001'	(FIFO → S/T, FIFO → PCM)
A_CHANNEL[2,TX]	: V_CH_DIR	= 0	(transmit HFC-channel)
	: V_CH_NUM	= 3	(HFC-channel[AUX2])
A_HDLC_PAR[2,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	= 3	(FIFO #3)
A_CON_HDLC[3,TX]	: V_IFF	= 0	(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0	(HDLC mode)
	: V_TRP_IRQ	= 1	(enable FIFO)
	: V_DATA_FLOW	= '001'	(FIFO → S/T, FIFO → PCM)
A_CHANNEL[3,TX]	: V_CH_DIR	= 0	(transmit HFC-channel)
	: V_CH_NUM	= 3	(HFC-channel[AUX2])
A_HDLC_PAR[3,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	= 3	(HFC-channel[AUX2])
A_CH_MSK[3,TX]	: V_CH_MSK	= '1000 1100'	(mask byte)
R_AUX2_TX_SL	: V_AUX2_TX_SL	= 7	(PCM slot[7,TX])
	: V_AUX2_TX_ROUT	= '10'	(data to pin STIO1)

❷ FIFO-to-PCM (RX)

HFC-channel[AUX2,RX] receives data bits that are to be distributed to FIFO[2,RX] and FIFO[3,RX].

The registers A_HDLC_PAR[FIFO] of FIFO[2,RX] and FIFO[3,RX] define the numbers of valid data bits and their positions in the HFC-channel byte. These bits are dispersed to FIFO[2,RX] and FIFO[3,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[2,RX] is f_1, the second byte is f_2, and so on. FIFO[3,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	= 2		(FIFO #2)
A_CON_HDLC[2,RX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0		(HDLC mode)
	: V_TRP_IRQ	= 1		(enable FIFO)
	: V_DATA_FLOW	= '001'		(FIFO ← PCM)
A_CHANNEL[2,RX]	: V_CH_DIR	= 1		(receive HFC-channel)
	: V_CH_NUM	= 3		(HFC-channel[AUX2])
A_HDLC_PAR[2,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	= 3		(FIFO #3)
A_CON_HDLC[3,RX]	: V_IFF	= 0		(0x7E as inter frame fill)
	: V_HDLC_TRP	= 0		(HDLC mode)
	: V_TRP_IRQ	= 1		(enable FIFO)
	: V_DATA_FLOW	= '001'		(FIFO ← PCM)
A_CHANNEL[3,RX]	: V_CH_DIR	= 1		(receive HFC-channel)
	: V_CH_NUM	= 3		(HFC-channel[AUX2])
A_HDLC_PAR[3,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_AUX2_RX_SL	: V_AUX2_RX_SL	= 7		(PCM slot[7,RX])
	: V_AUX2_RX_ROUT	= '10'		(data from pin STIO2)

Table 3.12: *Subchannel processing according to Figure 3.12 (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 3.12 (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 HFC-S mini FIFO registers*

Write only registers:

Address	Name	Page
0x0B	R_FIFO_REV	91
0x0C	R_FIFO_THRES	92
0x0D	R_DF_MD	93
0x0E	A_INC_RES_FIFO	93
0x0F	R_FIFO	94
0x84	A_FIFO_DATA_NOINC	94
0xF4	A_CH_MSK	95
0xFA	A_CON_HDLC	96
0xFB	A_HDLC_PAR	98
0xFC	A_CHANNEL	99

Read only registers:

Address	Name	Page
0x04	A_Z1	99
0x06	A_Z2	100
0x0C	A_F1	100
0x0D	A_F2	100
0x1A	A_USAGE	101
0x1B	R_FILL	101

Read / write register:

Address	Name	Page
0x80	A_FIFO_DATA	102

4.1 Overview

There are four receive FIFOs and four transmit FIFOs with eight HDLC controllers in whole. The HDLC circuits are located on the S/T-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 S/T-interface or to the PCM bus (transmit FIFO operation). Automatic zero deletion is done in HDLC mode when the HDLC data comes from the S/T-interface or PCM bus (receive FIFO operation).

There is a transmit and a receive FIFO for each B-channel, the D-channel (AUX1-channel) and the additional E-channel (AUX2-channel).

The FIFO control registers are used to select and control the FIFOs of the HFC-S mini. These registers exist for every FIFO number as well as for both receive and transmit directions. A FIFO is selected by the register R_FIFO.

All FIFOs are disabled after reset (hardware reset or soft reset). With the register A_CON_HDLC the selected FIFO is enabled by setting at least one of V_HDLC_TRP or V_TRP_IRQ to a value different from zero.

4.2 FIFO counters

The FIFOs are realized as ring buffers with 128 bytes each in the internal SRAM. They are controlled by counters. $Z1$ 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 $Z1 = Z2$ the FIFO is empty.

After every pulse on the F0IO signal, HDLC bytes are transferred to the S/T-interface (from a transmit FIFO) and HDLC bytes are read from the S/T-interface (transferred to a receive FIFO). A connection to the PCM interface is also possible.

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

Additionally, there are two 3 bit counters $F1$ and $F2$ for every FIFO for counting the HDLC frames. Up to 7 frames can be stored in every FIFO. The F-counters 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 counters is $Z1 = Z2 = Z_{max} = 0\times7F$ and $F1 = F2 = F_{max} = 7$. An increment leads to $Z_{min} = 0\times00$ or $F_{min} = 0\times00$ respectively.

This initialization of all FIFOs together can be carried out with a hardware or a soft reset. For this, the bit V_SRES in register R_CIRM has to be set. Individual FIFOs can be reset with bit V_RES_FIFO of the register A_INC_RES_FIFO.

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 the HFC-S mini. This means an access to FIFO control registers and data registers is not allowed until BUSY status is cleared (bit V_BUSY of R_STATUS register). The maximum duration takes 25 clock cycles ($\sim 2\,\mu$s). Status, interrupt and control registers can be read and written at any time.

4.3 FIFO operation

Important !

The HDLC controller does not work without F0IO and C4IO clocks.

If no external clock source is connected to these pins (PCM slave mode), V_PCM_MD of register R_PCM_MD0 must be '1' to select PCM master mode.

4.3.1 HDLC transmit FIFOs

Data can be transmitted from the microprocessor bus interface to the FIFO with write access to the registers A_FIFO_DATA or A_FIFO_DATA_NOINC. The HFC-S mini converts the data into HDLC code and tranfers it from the FIFO to the S/T or the PCM bus interface.

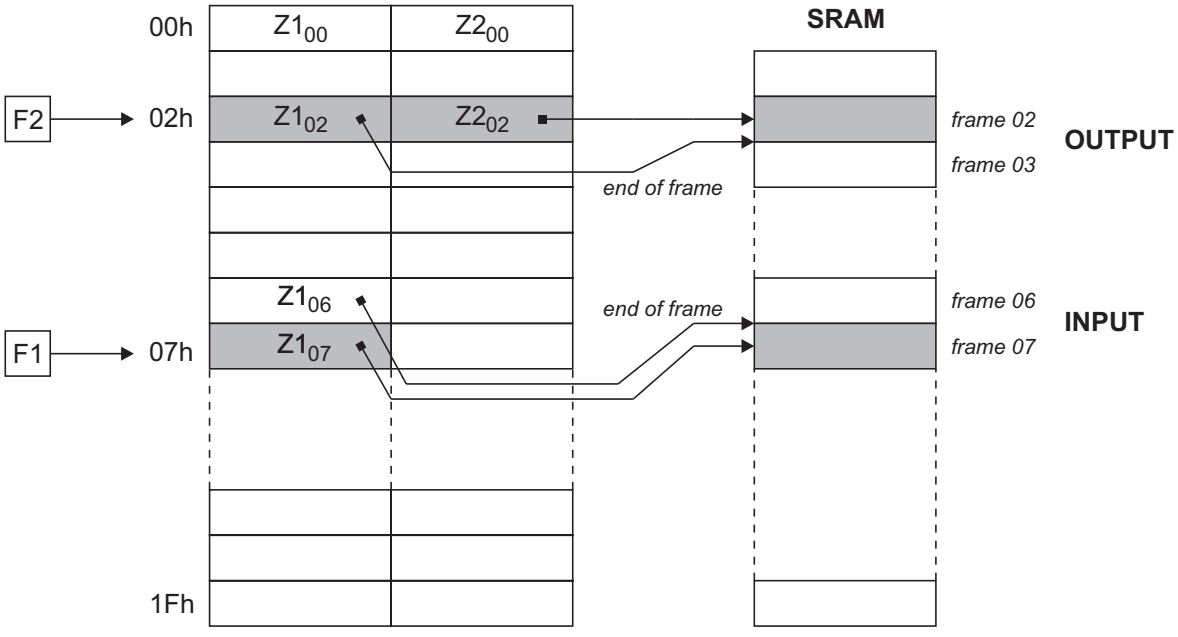


Figure 4.1: *FIFO organization*

The HFC-S mini checks $Z1$ and $Z2$. If $Z1 = Z2$ (FIFO empty) the HFC-S mini generates a HDLC flag ('01111110') or continuous '1's (depending on the bit V_IFF of the register A_CON_HDLC) and transmits it to the S/T-interface. In this case $Z2$ is not incremented. If also $F1 = F2$ only HDLC flags or continuous '1's are sent to the S/T-interface and all counters remain unchanged. If the frame counters are unequal; $F2$ is incremented and the HFC-S mini 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 when A_FIFO_DATA is used. 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 S/T-interface side of the HFC-S mini. $Z1(F2)$ 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 the 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. $Z1(F2)$ and $Z2(F2)$ can not be accessed.

4.3.2 Automatic D-channel frame repetition

The D-channel transmit FIFO has a special feature. If the S/T-interface signals a D-channel contention in TE mode before the CRC is sent, the $Z2$ counter is set to the starting address of the current frame and the HFC-S mini tries to repeat the entire frame automatically.

> **Please note !**
>
> The HFC-S mini 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.

4.3.3 FIFO full condition in HDLC transmit HFC-channels

There are two different FIFO full conditions. The first one is met when the FIFO contents comes up to 7 frames. There is no possibility for HFC-S mini to manage more frames even if the frames are very small. The driver software must check that there are never more than 7 HDLC frames in a FIFO.

The second limitation is the size of the FIFO (128 bytes each). FIFO full condition can be checked by reading the A_USAGE register. It shows the actually occupied FIFO space in bytes. Furthermore a threshold value can be set for all transmit and receive FIFOs in the R_FIFO_THRES register. Then the R_FILL register shows an indication of the filling level for each FIFO. After data processing from or to a FIFO, the R_FILL register must be updated with a change FIFO or an increment FIFO counter operation. After this it takes up to 250 μs until the bit value in the R_FILL register of the processed FIFO represents the actual state of the FIFO filling level.

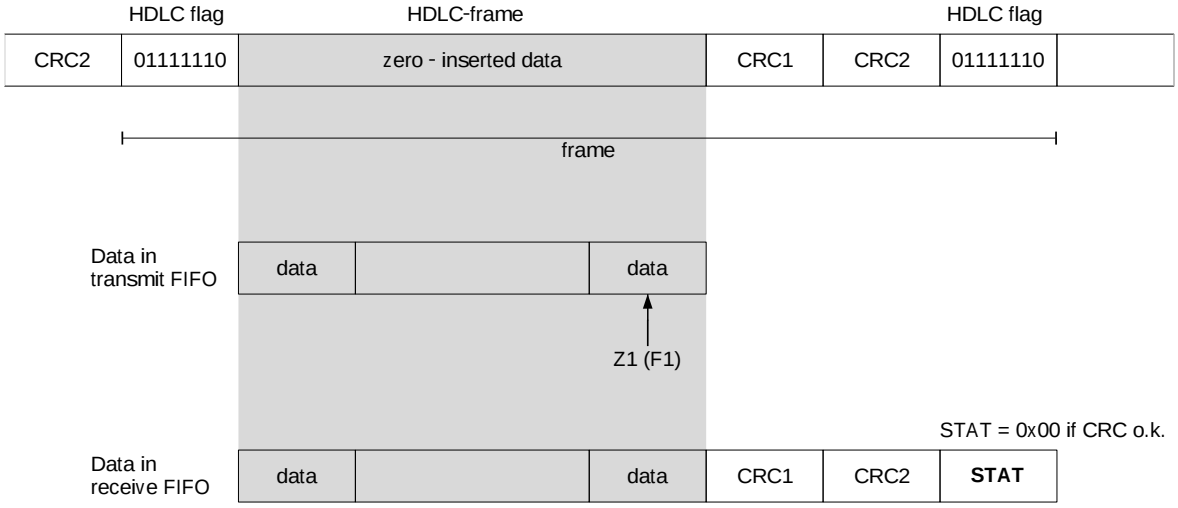


Figure 4.2: *FIFO data organization in HDLC mode*

4.3.4 HDLC receive FIFOs

The receive HFC-channels receive data from the S/T 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 microprocessor bus interface.

The HFC-S mini 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 the HFC-S mini into plain data. After the ending flag of a frame the HFC-S mini 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.

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 the HFC-S mini automatically and the next frame can be received.

After reading a frame via the microprocessor 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 S/T-interface side of the HFC-S mini. $Z2(F2)$ is used for the frame which is just beeing transmitted to the microprocessor 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 the 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 the 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 the HFC-S mini. Otherwise the first byte of the FIFO will be taken from the internal buffer and may be invalid.

4.3.5 FIFO full condition in HDLC receive HFC-channels

Because of the ISDN B- and D-channels not having a hardware based flow control there is no possibility to stop input data if a receive FIFO is full.

Thus there is no FIFO full condition implemented in the HFC-S mini. The HFC-S mini assumes that the FIFOs are deep enough that the host processor's hardware and software is able to avoid any overflow of the receive FIFOs. Overflow conditions are again more than 7 input frames or a memory overflow of the FIFO because of excessive data (more than 128 bytes).

Because HDLC procedures only know a window size of 7 frames no more than 7 frames are sent without software intervention. Due to the great size of the HFC-S mini FIFOs it is easy to poll the HFC-S mini even in large time intervalls without having to fear a FIFO overflow condition.

The register R_FILL indicates if the fill level of some FIFOs exceeds the number of bytes defined in the R_FIFO_THRES register. A byte overflow can be avoided by polling this register. After data processing from or to a FIFO, the R_FILL register must be updated with a change FIFO or an increment FIFO counter operation. After this, it takes up to 250 μs until the bit value in the R_FILL register of the processed FIFO represents the actual state of the FIFO filling level. The register A_USAGE shows the actually occupied FIFO space in bytes. A byte overflow can be avoided by polling this register.

To avoid any undetected FIFO overflows the software driver should check $F1 - F2$, i.e. the number of frames in the FIFO. If $F1 - F2$ is less than the number in the last reading, an overflow took place if there was no reading of a frame in between.

After a detected FIFO overflow condition this FIFO must be reset by setting the FIFO reset bit V_RES_FIFO in register A_INC_RES_FIFO.

4.3.6 Transparent mode of the HFC-S mini

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

dress. 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 S/T or PCM bus interface.

Transmit and receive transparent data can be done in two ways. The usual way is transporting FIFO data to the S/T-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 register R_FIFO_REV.

> **Important !**
>
> For normal data transmission the register A_HDLC_PAR must be set to 0x00. To use 56 kbit/s restricted mode for U.S. ISDN lines the register A_HDLC_PAR must be set to 0x07 for B-channels.
>
> For the D-channel, the A_HDLC_PAR register must be set to 0x02.

4.3.7 Reading F- and Z-counters

For all asynchronous host accesses to the HFC-S mini there is a small chance that a register is changed just in the moment when it is read. Because of slightly different delays of individual bits, it is even possible that the read value is fully invalid. Therefore we advise to read a F- or Z-counter register until two consecutive readings find the same value.

This is not necessary for a time period of at least 125 μs after writing R_FIFO. It is also not necessary for Z-counters of receive FIFOs if $F1 \neq F2$. Then a whole frame has been received and the counters $Z1(F2)$ and $Z2(F2)$ are stable and valid.

4.4 Direct RAM access

FIFO data is read or written with access to the registers A_FIFO_DATA or A_FIFO_DATA_NOINC normally.

Test programs, e.g., might require direct memory access. The following example shows, how four consecutive bytes are written with values $x1 \ldots x4$ to the addresses $a \ldots a+3$:

Register setup:

R_RAM_ADDR0 :	V_RAM_ADDR0	$= a[7..0]$	write lower address byte
R_RAM_ADDR1 :	V_RAM_ADDR1	$= a[10..8]$	write upper address bits
:	V_ADDR_RES	$= 0$	normal operation
:	V_ADDR_INC	$= 1$	automatic address increment
R_RAM_DATA :	V_RAM_DATA	$= x1$	write $x1$ to address a
:	V_RAM_DATA	$= x2$	write $x2$ to address $a+1$
:	V_RAM_DATA	$= x3$	write $x3$ to address $a+2$
:	V_RAM_DATA	$= x4$	write $x4$ to address $a+3$

The internal RAM contains eight FIFOs in the address area 0..0x3FF ($8 \cdot 128$ byte $= 1024$ byte). RAM address are composed of the FIFO number, data direction and the address within a FIFO as shown in Table 4.2.

Table 4.2: *RAM address composition*

Address bits	Meaning	Register correspondence
0	Data direction ('0' = TX, '1' = RX)	V_FIFO_DIR in register R_FIFO
2..1	FIFO number (0..3)	V_FIFO_NUM in register R_FIFO
9..3	FIFO byte (0..127)	–
10	Must be '0', for internal HFC-S mini usage only	

Before reading or writing to the RAM, the corresponding FIFO should be disabled.

4.5 Register description

> **Please note !**
>
> Due to a thorough revision of the HFC-S mini data sheet, some registers had to be renamed. Please see remarks on page 15.

4.5.1 Write only registers

R_FIFO_REV **(w)** 0x0B
(formerly **F_CROSS**)

Bit order configuration register

The bit order can be reversed for every FIFO separately. LSB first is used in HDLC mode while MSB first is used in transparent mode. The bit order is being reversed for the data stored into a receive FIFO or when the data is read from a transmit FIFO.

'0' = normal bit order (LSB first)
'1' = reversed bit order (MSB first)

Bits	Reset value	Name	Description
0	0	**V_FIFO0_TX_REV**	**Bit order for FIFO[0,TX]**
1	0	**V_FIFO0_RX_REV**	**Bit order for FIFO[0,RX]**
2	0	**V_FIFO1_TX_REV**	**Bit order for FIFO[1,TX]**
3	0	**V_FIFO1_RX_REV**	**Bit order for FIFO[1,RX]**
4	0	**V_FIFO2_TX_REV**	**Bit order for FIFO[2,TX]**
5	0	**V_FIFO2_RX_REV**	**Bit order for FIFO[2,RX]**
6	0	**V_FIFO3_TX_REV**	**Bit order for FIFO[3,TX]**
7	0	**V_FIFO3_RX_REV**	**Bit order for FIFO[3,RX]**

R_FIFO_THRES (formerly F_THRES)	(w)	0x0C

FIFO fill level control register

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

Bits	Reset value	Name	Description
3..0	1	V_THRES_TX	**Threshold for all transmit FIFOs** '0000' = 0 bytes '0001' = 8 bytes '0010' = 16 bytes '0011' = 24 bytes '0100' = 32 bytes '0101' = 40 bytes '0110' = 48 bytes '0111' = 56 bytes '1000' = 64 bytes '1001' = 72 bytes '1010' = 80 bytes '1011' = 88 bytes '1100' = 96 bytes '1101' = 104 bytes '1110' = 112 bytes '1111' = 120 bytes The corresponding bit in theR_FILL registeris set if the number of bytes in a transmit FIFO is greater or equal than specified here.
7..4		V_THRES_RX	**Threshold for all receive FIFOs** '0000' = 0 bytes '0001' = 8 bytes '0010' = 16 bytes '0011' = 24 bytes '0100' = 32 bytes '0101' = 40 bytes '0110' = 48 bytes '0111' = 56 bytes '1000' = 64 bytes '1001' = 72 bytes '1010' = 80 bytes '1011' = 88 bytes '1100' = 96 bytes '1101' = 104 bytes '1110' = 112 bytes '1111' = 120 bytes The corresponding bit in theR_FILL registeris set if the number of bytes in a receive FIFO is greater or equal than specified here.

R_DF_MD (w) 0x0D
(formerly **F_MODE**)

Data flow mode selection register

Bits	Reset value	Name	Description
6..0	0	(reserved)	Must be 0.
7	0	**V_CSM**	**Channel Select Mode (CSM)** '0' = Simple Mode (SM) '1' = Channel Select Mode (CSM)

A_INC_RES_FIFO [FIFO] (w) 0x0E
(formerly **INC_RES_F**)

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.
7..2	0	(reserved)	Must be 0.

R_FIFO (w) 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.

Bits	Reset value	Name	Description
0	0	**V_FIFO_DIR**	**FIFO data direction** '0' = transmit FIFO data '1' = receive FIFO data
2..1	0	**V_FIFO_NUM**	**FIFO number**
7..3	0	(reserved)	Must be 0.

A_FIFO_DATA_NOINC (w) 0x84
(formerly **FIF_DATA**)

FIFO data register

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

Bits	Reset value	Name	Description
7..0		**V_FIFO_DATA_NOINC**	**Data byte** Write data to the FIFO selected in the R_FIFO register 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 automatically. See A_FIFO_DATA on page 102 for FIFO data access with automatic *Z*-counter increment.)

A_CH_MSK [FIFO] (w) 0xF4
(formerly **CH_MSK**)

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

This register is ignored for receive FIFOs.

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

A_CON_HDLC [FIFO] (w) 0xFA
(formerly **CON_HDLC**)

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 the 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 the D-channel this bit must be '0'.
3..2	0	**V_TRP_IRQ**	**Transparent mode interrupt selection** **In transparent mode:** The FIFO is enabled and an interrupt is generated all 2^n bytes when the bits [n-1:0] of the Z1-counter (in transmit direction) or the Z2-counter (in receive direction) change from all '1's to all '0's. $n =$ V_TRP_IRQ $+ 3$. '00' = all $2^3 = 8$ bytes an interrupt is generated '01' = all $2^4 = 16$ bytes an interrupt is generated '10' = all $2^5 = 32$ bytes an interrupt is generated '11' = all $2^6 = 64$ bytes an interrupt is generated **In HDLC mode:** An interrupt is generated at end of frame. '00' = FIFO disabled, no interrupt '01'..'11' = FIFO enabled, end of frame interrupt
4	0	(reserved)	Must be '0'.

(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 the register R_FIFO): '000', '001' = FIFO → S/T, FIFO → PCM '010', '011' = FIFO → PCM '100', '101' = FIFO → S/T, S/T → PCM '110', '111' = S/T → PCM In receive operation (V_FIFO_DIR = 1 in the register R_FIFO): '000', '100' = FIFO ← S/T '001', '101' = FIFO ← PCM '010', '110' = FIFO ← S/T, S/T ← PCM '011', '111' = FIFO ← PCM, S/T ← PCM

(For details on bitmap V_DATA_FLOW see Fig. 3.3 and 3.4 on page 51.)

Important !

A FIFO is disabled if V_HDLC_TRP + V_TRP_IRQ = 0 in register A_CON_HDLC[FIFO]. This setting is useful to reduce RAM accesses if a FIFO is not used at all.

If HFC-channel data is routed through the switches of the flow controller (Fig. 3.3 and 3.4) the FIFO must be enabled. That applies to all connections except the PCM-to-PCM data transmission.

A_HDLC_PAR [FIFO] (w) 0xFB
(formerly **HDLC_PAR**)

HDLC parameters of the selected FIFO

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

Note: This register must be 0x02 for the D-channel and the E-channel.

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 frame (useful only in transparent mode)
7	0	**V_INV_DATA**	**Invert data** '0' = normal data transmission '1' = inverted data transmission

A_CHANNEL [FIFO] **(w)** 0xFC

HFC-channel assignment for the selected FIFO

This register is only used in *Channel Select Mode* (CSM).

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

Bits	Reset value	Name	Description
0		**V_CH_DIR**	**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_DIR = *direction* of FIFO[*x*,*direction*].
2..1		**V_CH_NUM**	**HFC-channel number / name** 0 = HFC-channel[B1] 1 = HFC-channel[B2] 2 = HFC-channel[AUX1] 3 = HFC-channel[AUX2] **Reset value:** This bitmap is reset to the same value as the current FIFO, i.e. V_CH_NUM = *x* of FIFO[*x*,*direction*].
7..3	0	(reserved)	Must be 0.

4.5.2 Read only registers

A_Z1 [FIFO] **(r)** 0x04
(formerly **FIF_Z1**)

FIFO input counter *Z*1

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

Bits	Reset value	Name	Description
7..0	0x7F	**V_Z1**	**Counter value of** *Z*1 (0..0x7F)

A_Z2 [FIFO] (formerly **FIF_Z2**) **(r)** 0x06

FIFO output counter *Z*2

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

Bits	Reset value	Name	Description
7..0	0x7F	**V_Z2**	**Counter value of** *Z*2 (0..0x7F)

A_F1 [FIFO] (formerly **FIF_F1**) **(r)** 0x0C

FIFO input HDLC frame counter *F*1

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

Bits	Reset value	Name	Description
7..0	7	**V_F1**	**Counter value of** *F*1 (0..7) Up to 7 HDLC frames can be stored in every FIFO.

A_F2 [FIFO] (formerly **FIF_F2**) **(r)** 0x0D

FIFO output HDLC frame counter *F*2

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

Bits	Reset value	Name	Description
7..0	7	**V_F2**	**Counter value of** *F*2 (0..7) Up to 7 HDLC frames can be stored in every FIFO.

A_USAGE [FIFO] **(r)** 0x1A
(formerly **F_USAGE**)

FIFO fill level

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

Bits	Reset value	Name	Description
7..0	0	**V_USAGE**	**Number of bytes currently stored in the FIFO** (0..128)

R_FILL **(r)** 0x1B
(formerly **F_FILL**)

FIFO fill level

When a bit is set to '1', more than the specified number of bytes is currently beeing 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.

Note: After data reading from or writing to a FIFO, the R_FILL register must be updated with a change FIFO or an increment FIFO counter operation. After this it takes up to 250 μs until R_FILL of the processed FIFO represents the actual state of the FIFO fill level.

Bits	Reset value	Name	Description
0	0	**V_FIFO0_TX_FILL**	**FIFO[0,TX] fill level**
1	0	**V_FIFO0_RX_FILL**	**FIFO[0,RX] fill level**
2	0	**V_FIFO1_TX_FILL**	**FIFO[1,TX] fill level**
3	0	**V_FIFO1_RX_FILL**	**FIFO[1,RX] fill level**
4	0	**V_FIFO2_TX_FILL**	**FIFO[2,TX] fill level**
5	0	**V_FIFO2_RX_FILL**	**FIFO[2,RX] fill level**
6	0	**V_FIFO3_TX_FILL**	**FIFO[3,TX] fill level**
7	0	**V_FIFO3_RX_FILL**	**FIFO[3,RX] fill level**

4.5.3 Read / write register

A_FIFO_DATA [FIFO] (formerly **FIF_DATA**)	**(r/w)**	0x80

FIFO data register

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

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

Chapter 5

S/T interface

Table 5.1: *Overview of the S/T interface pins*

Number	Name	Description	Number	Name	Description
13	TX2_HI	Transmitter output 2	20	R2	Receiver input 2
14	/TX1_LO	Ground driver for transmitter 1	21	LEV_R2	Level detect for R2
15	/TX_EN	Transmitter enable	22	LEV_R1	Level detect for R1
16	/TX2_LO	Ground driver for transmitter 2	23	R1	Receiver input 1
17	TX1_HI	Transmitter output 1	25	ADJ_LEV	Level generator
			28	AWAKE	Wakeup input

Table 5.2: *Overview of the S/T interface registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x30	R_ST_WR_STA	124	0x30	R_ST_RD_STA	130
0x31	R_ST_CTRL0	125	0x34	R_ST_SQ_RD	131
0x32	R_ST_CTRL1	126	0x3C	R_ST_B1_RX	131
0x33	R_ST_CTRL2	127	0x3D	R_ST_B2_RX	132
0x34	R_ST_SQ_WR	127	0x3E	R_ST_D_RX	132
0x37	R_ST_CLK_DLY	128	0x3F	R_ST_E_RX	132
0x3C	R_ST_B1_TX	129			
0x3D	R_ST_B2_TX	129			
0x3E	R_ST_D_TX	129			

5.1 Overview

The S/T interface module consists of the receive and transmit data pathes with a clock processing unit each, the clock distribution unit and the S/T state machine. The overall connection of these units is shown in Figure 5.1.

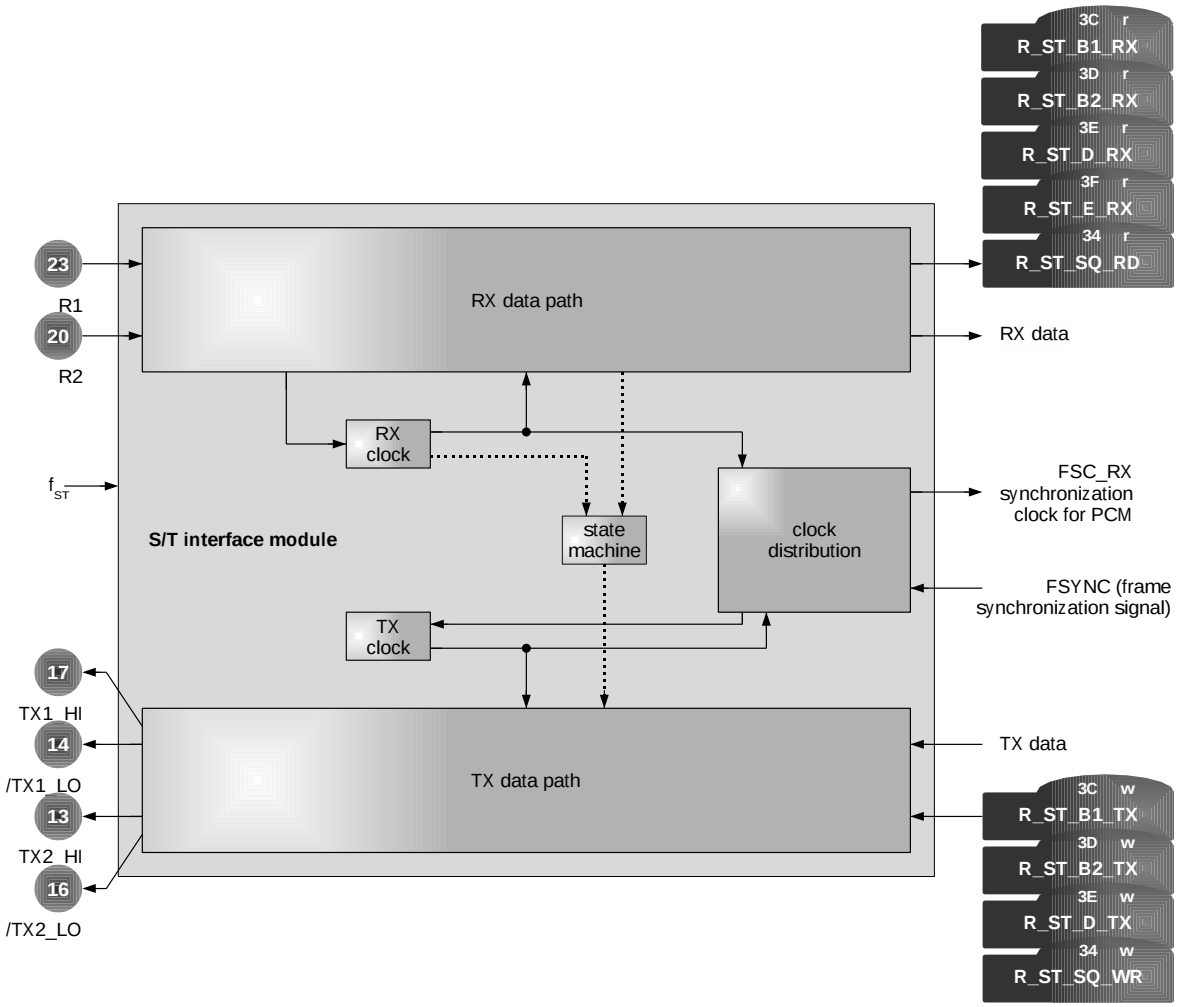


Figure 5.1: *Overview of the S/T interface module*

Received data from the pins R1 and R2 is passed through the RX data path to the switching buffer (see Figures 3.3 and 3.3 in Section 3.2). A bit clock and a frame clock are derived from the received data. 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 can be generated from several clocks which are obtained from the clock distribution 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.

A detailed block diagram of the S/T interface module is shown in Figure 5.3.

5.2 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.

A modified AMI code is used for transmission. A binary '1' is represented by 0 V level. A binary '0' is represented by alternating signal of 750 mV nominal on the line side.

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

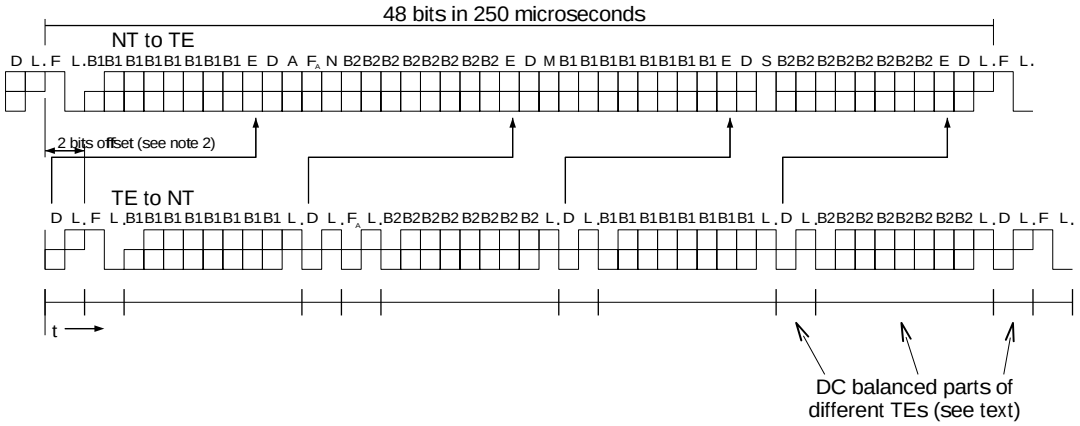


Figure 5.2: *Frame structure at reference point S and T (see legend in Table 5.3 and specification [3])*

The nominal 2 bit offset is as seen from the TE in Figure 5.2. The offset can be adjusted in TE mode with the bitmap V_ST_CLK_DLY of the register R_ST_CLK_DLY. The corresponding offset at the NT may be bigger 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 DC-balance bit at the end of the frame because all data comes from the same source. The L-bit at the beginning of the frame belongs to the preceding F-bit and is used for code violation.

5.3 Data transmission

To transfer any data over the B-channels they have to be enabled for transmission by setting V_B1_EN or V_B2_EN in register R_ST_CTRL0. Receive is enabled by setting V_B1_RX_EN or V_B2_RX_EN in register R_ST_CTRL2.

5.4 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.

Table 5.3: *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)

The F_A- and M-bits are used to identify the multiframes. Table 5.4 shows the position of the multiframe bits. A detailed specification of the multiframe structure is given in [3].

Multiframe transmission must be enabled with V_SQ_EN = '1' in register R_ST_CTRL0).

5.5 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 [1]:

INFO 0: No signal on line, continuous '1's.

INFO 1: Continous signal at nominal bit rate with a '00111111' pattern which has a positive zero first and a negative zero following.

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

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

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

[1] Please see [3] for a detailed description of the INFO signals.

Table 5.4: *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 [3] defines five subchannels for the S-multiframe. Only subchannel 1 is used from the /chip. All other subchannels are set to '0'.

5.6 State machine

The HFC-S mini is equiped with an S/T interface according to ITU-T I.430 [3] and ETSI TBR03 [1] specifications. It can be configured into TE or NT mode by setting V_ST_MD in register R_ST_CTRL0.

The current Fx or Gx state can be read out of the register R_ST_RD_STA. However, it is possible to overwrite automatic state transition logic of the state machine by setting the bit V_ST_LD_STA of the register R_ST_WR_STA.

Activation and deactivation can be initiated by writing the bitmap V_ST_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 Fx / Gx state machine, the register R_ST_CLK_DLY must be set. For the recommended external S/T circuitry, the default value is 0x0E for TE and 0x6C for NT.

> **Important !**
>
> The S/T state machine is stuck at '0' after a reset. In this state the HFC-S mini sends no signal (INFO 0) on the S/T line and is not able to activate it by incoming INFO x. Writing a '0' to bit V_ST_LD_STA of the R_ST_WR_STA register 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 of the R_ST_RD_STA register or it can permanently be enabled by setting bit V_G2_G3_EN of the R_ST_CTRL1 register.

It is not allowed to switch on the state machine before the S/T interface has finished its initialization procedure. Otherwise, this can lead to fatal consequences with current ISDN connections on the S/T bus.

Incoming INFO 0 at state F7 causes a state change to F3 normally. Sometimes an intermediate state F6 occurs. In this case another state change to F3 comes up within 1 ms and F6 can be ignored. V_INFO0 in register R_ST_RD_STA should be checked with every state change from F7 to F6. When this bit is set, the state should be checked again after about 1 ms. When it is F3, the intermediate state F6 has to be ignored. [2]

Tables 5.5 and 5.6 show the S/T interface state matrix in NT and TE mode.

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

Table 5.5: *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.

[*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_SET_G2_G3 of the R_ST_WR_STA register must be set to allow this transition or V_G2_G3_EN in the register R_ST_CTRL1 must be set to allow automatic transition G 2 → G 3.

Table 5.6: *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	|	|	—	|	—
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.

5.7 Synchronization clocks

A detailed view inside the line interface block diagram of Figure 5.1 is shown in Figure 5.3. All clocks are derived from the clock $f_{ST} = 6.144\,\text{MHz}$. Frame synchronization is accomplished by evaluating the code violations in the S/T frame.

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 synchronized to the F0IO input signal in NT mode.

TE mode: A TE is always taken as synchronization source for ISDN applications because it delivers the clock from the central office switch. Thus the 192 kHz bit clock as well as the 8 kHz frame clock 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.

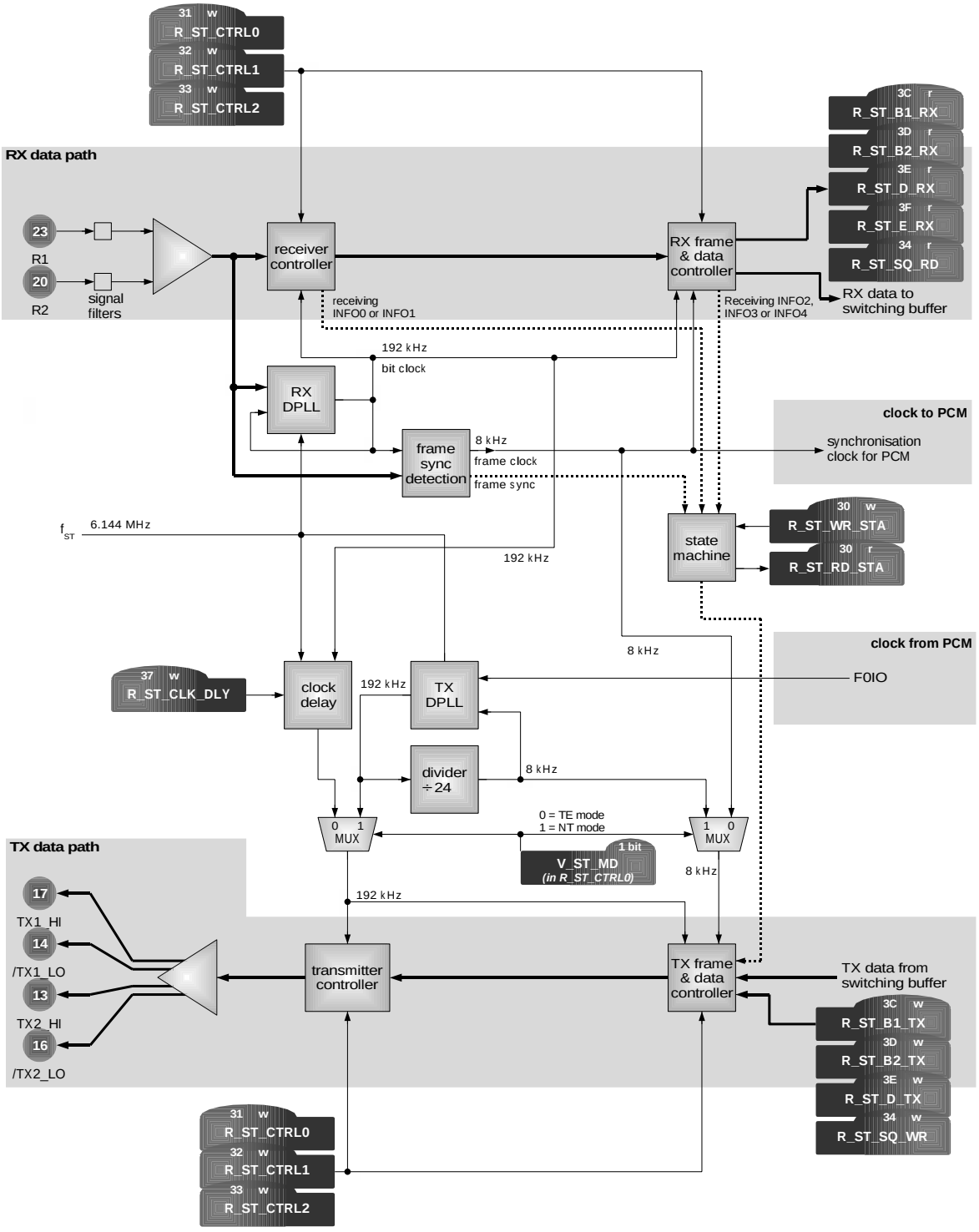


Figure 5.3: *Block diagram of the S/T interface module*

5.8 S/T transformers

Customers of Cologne Chip can choose from a variety of S/T transformers for ISDN Basic Rate Interface. All transformers are compatible to the HFC 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 receiver circuitry)

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 5.7. 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 5.7: *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

Type	Device
Dual Transformer Module with choke:	
	APC 48301
Hybrid Transformer Module with choke:	
	APC 5568-3V
	APC 5568-5V
	APC 5568DS-3V (SMD)
	APC 5568DS-5V (SMD)

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)

Type	Device
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 5.7: *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)

Type	Device
Dual Transformer Module without choke:	
	MUJ-103A-000 (SMD)
	MAJ-403-000 (SMD)
	MSJ-403A-000 (SMD)
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 (SMD, 3 kV)

(continued on next page)

Table 5.7: *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 5.7: *S/T transformer part numbers and manufacturers*

(continued from previous page)

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

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)
	543 80 008 00 (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.

5.9 External circuitries

In order to comply to the physical requirements of ITU-T I.430 [3] recommendation and considering the national requirements concerning overvoltage protection and electromagnetic compatibility (EMC), HFC-S mini needs some external circuitry, which are shown in this section.

5.9.1 External receive circuitry

The standard external receive circuitry for TE and NT mode is shown in Figure 5.4. All components are chosen for 3.3 V power supply.

Some components need different values when 5 V power supply is used. Please ask the support team of Cologne Chip for more information.

C1 and C2 are for reduction of high frequency input noise and should be placed as close as possible to the HFC-S mini.

WAKE_UP_1 and WAKE_UP_2 are for connection to the optional wakeup circuitry shown in Figure 5.7.

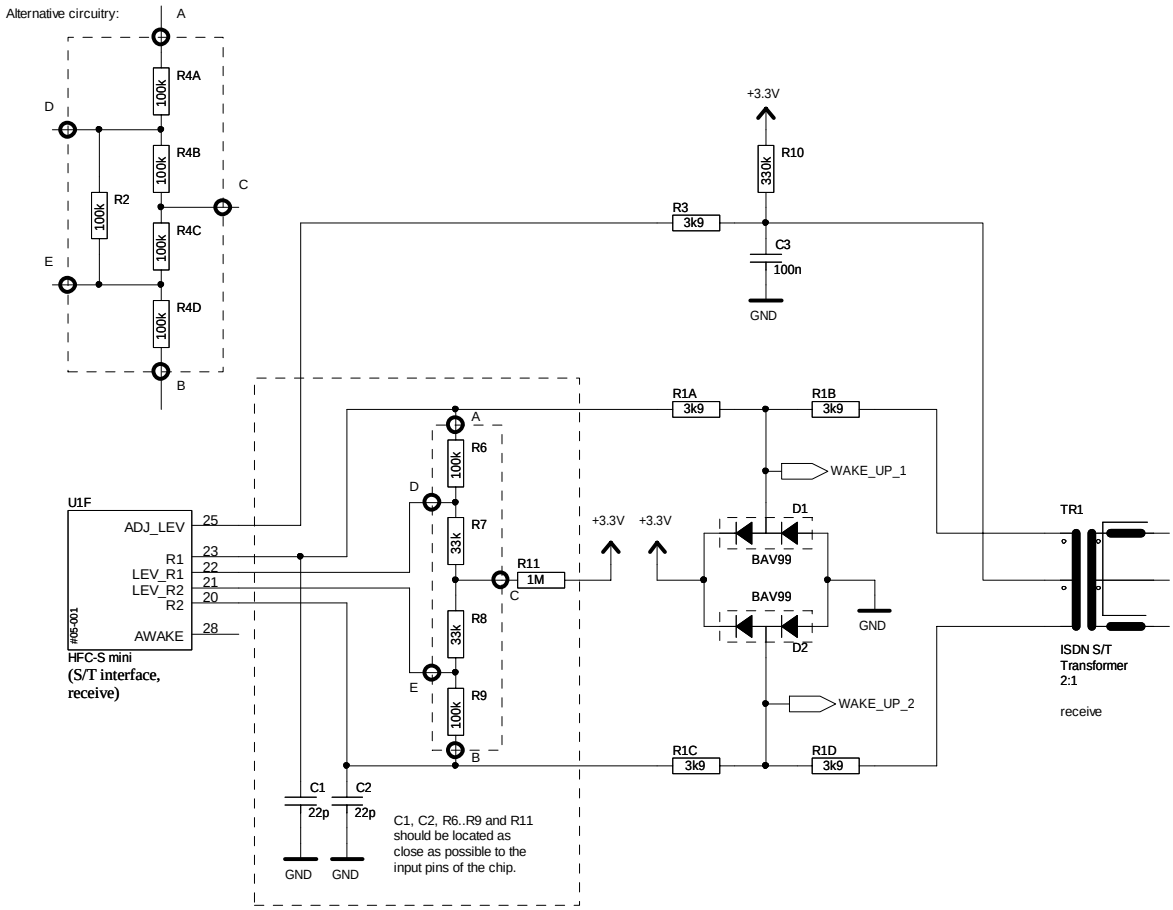


Figure 5.4: *External S/T receive circuitry for TE and NT mode*

5.9.2 External transmit circuitry

The standard external transmit circuitry for TE and NT mode is shown in Figure 5.5. All components are chosen for 3.3 V power supply.

Some components need different values when 5 V power supply is used. Please ask the support team of Cologne Chip for more information.

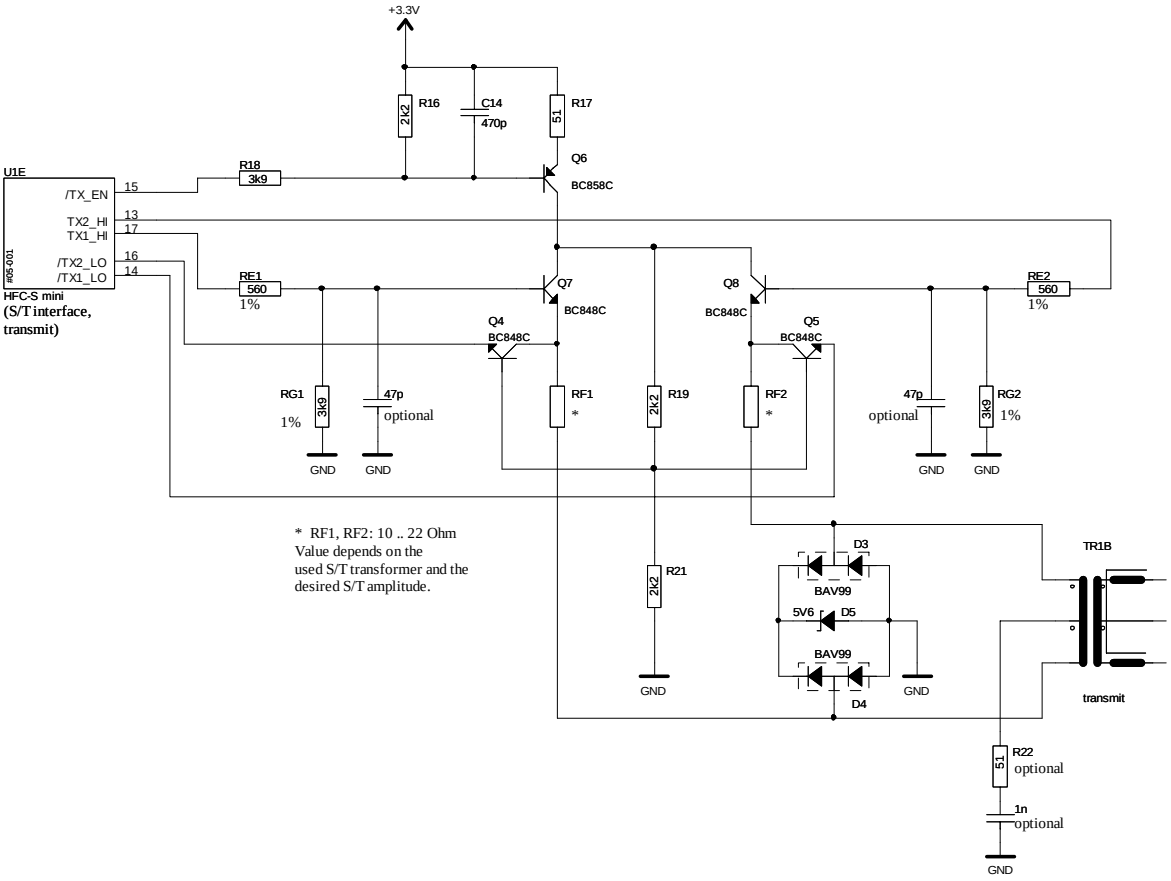


Figure 5.5: *External S/T transmit circuitry for TE and NT mode*

If the S/T interface is only used in NT mode, the simplified circuitry which is shown in Figure 5.6 can be used. Again only dimensioning for 3.3 V power supply is shown here.

5.9.3 External wakeup circuitry

The wakeup circuitry is optional. It enables the HFC-S mini to wake up by incoming INFO x (non INFO 0) signals on the S/T interface.

WAKE_UP_1 and WAKE_UP_2 are inputs from the receiver circuitry.

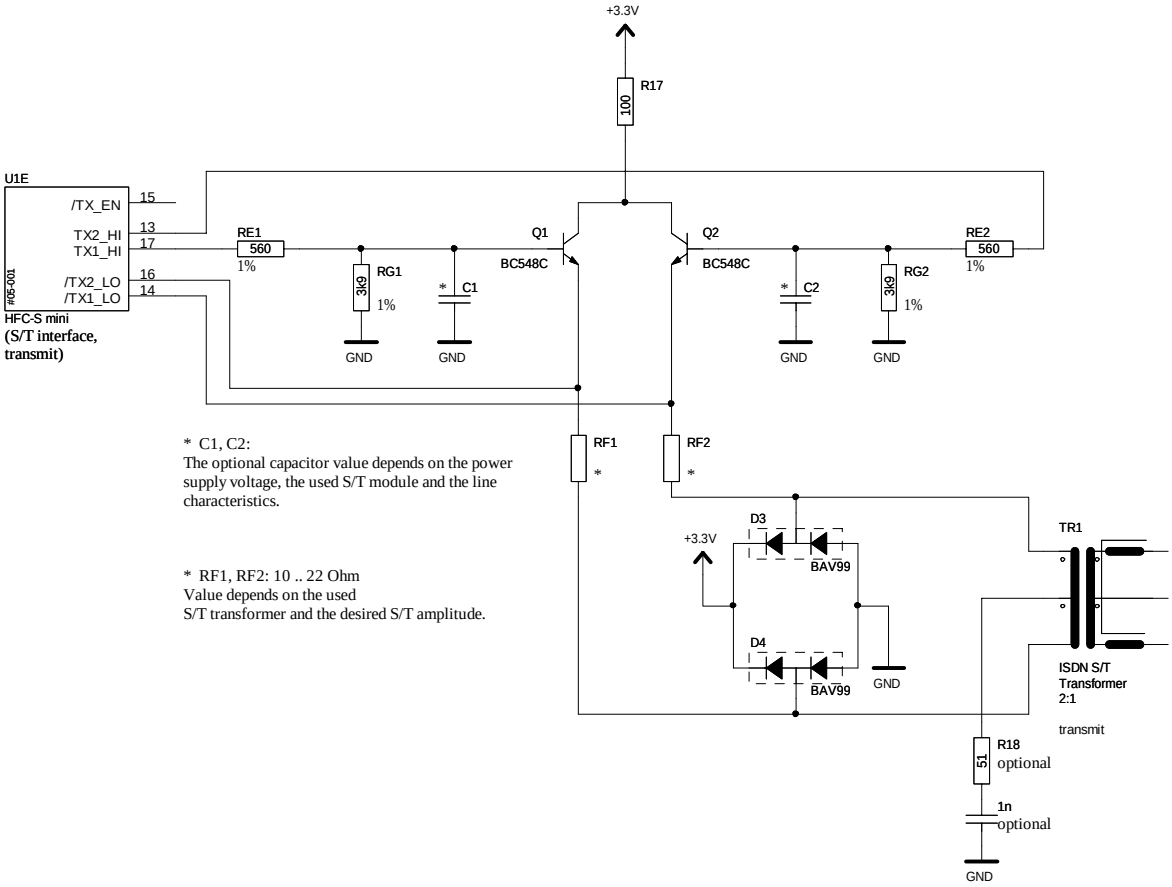


Figure 5.6: *External S/T transmit circuitry for NT mode only*

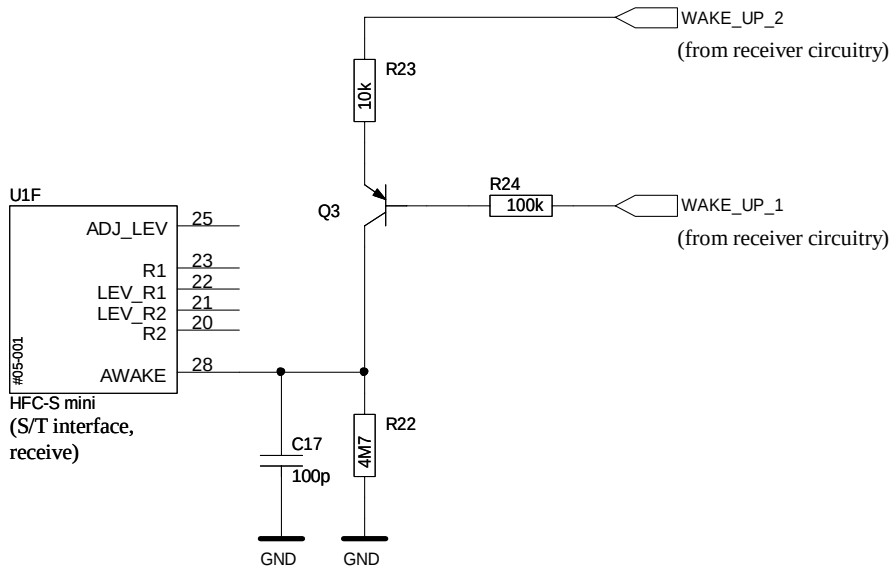


Figure 5.7: *External wakeup circuitry*

5.9.4 Transformer and ISDN jack connection

Figure 5.8 and 5.9 show the connection circuitry of the transformer and the ISDN jack in TE and NT mode [3]. The termination resistors R1 and R2 are optional.

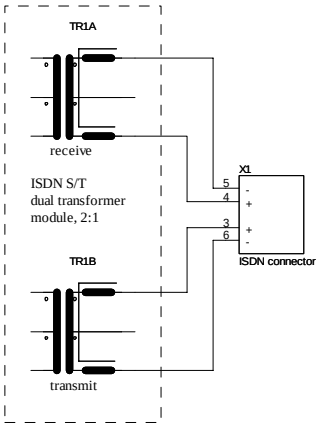


Figure 5.8: *Transformer and connector circuitry in TE mode*

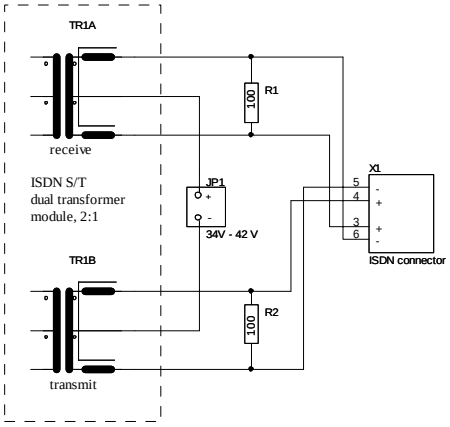
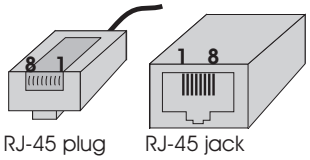


Figure 5.9: *Transformer and connector circuitry in NT mode (shown with optional 100 Ω termination, whole bus termination must be 50 Ω)*

[3] 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



5.10 Test loop

For electrical tests of layer 1, it is useful to set up an S/T test loop for the B1- and B2-channels. The test loop shown in Figure 5.10 receives data on the B1- and the B2-channel of the S/T interface. Data is routed to the PCM interface and looped back over the STIO1 pin. The register setup which has to be implemented is shown below.

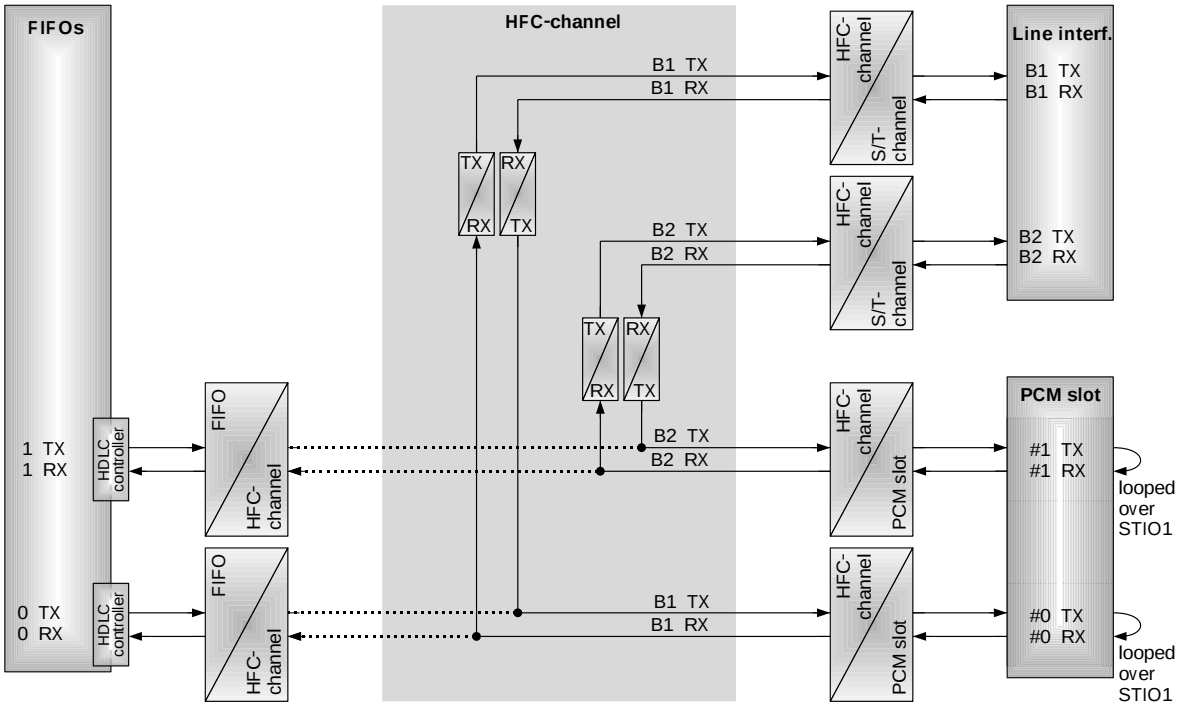


Figure 5.10: *Test loop configuration*

Register setup:				Part 1 (S/T configuration)
R_ST_CLK_DLY	: V_ST_CLK_DLY	= 0xF		Adjust the phase offset between receive and transmit direction (the value depends on the external circuitry)
	: V_ST_SMPL	= 0		
R_ST_CTRL0	: V_B1_EN	= 1		Enable B1 send data
	: V_B2_EN	= 1		Enable B2 send data
	: V_ST_MD	= 0		TE mode
	: V_D_PRIO	= 0		D-channel with high priority (8/9)
	: V_SQ_EN	= 0		Disable S/Q-bits
	: V_96KHZ	= 0		Disable 96 kHz test signal
	: V_TX_LI	= 1		Non capacitive line mode (depends on the external S/T circuitry)
	: V_ST_STOP	= 0		Oscillator active
R_ST_WR_STA	: V_ST_SET_STA	= 0		Set state F0
	: V_ST_LD_STA	= 0		Enable state machine (release S/T state machine for activation over the S/T interface by incoming INFO 2 or INFO 4.)
	: V_ST_ACT	= 0		(No operation)
	: V_SET_G2_G3	= 0		(No operation)
R_ST_CTRL2	: V_B1_RX_EN	= 1		Enable B1-channel receive
	: V_B2_RX_EN	= 1		Enable B2-channel receive

Register setup:				Part 2 (FIFO setup and data flow configuration)
R_FIFO	: V_FIFO_DIR	= 0		Select transmit FIFO
	: V_FIFO_NUM	= 0		Select FIFO #0 (connected to HFC-channel[B1,TX])
A_CON_HDLC[FIFO]	: V_IFF	= 0		Write HDLC flags 0x7E
	: V_HDLC_TRP	= 0		Select HDLC mode
	: V_TRP_IRQ	= 1		Enable FIFO
	: V_DATA_FLOW	= 6		Connect S/T ⟶ PCM
R_FIFO	: V_FIFO_DIR	= 1		Select receive FIFO
	: V_FIFO_NUM	= 0		Select FIFO #0 (connected to HFC-channel[B1,RX])
A_CON_HDLC[FIFO]	: V_IFF	= 0		Write HDLC flags 0x7E
	: V_HDLC_TRP	= 0		Select HDLC mode
	: V_TRP_IRQ	= 1		Enable FIFO
	: V_DATA_FLOW	= 6		Connect PCM ⟶ S/T
R_FIFO	: V_FIFO_DIR	= 0		Select transmit FIFO
	: V_FIFO_NUM	= 1		Select FIFO #1 (connected to HFC-channel[B2,TX])
A_CON_HDLC[FIFO]	: V_IFF	= 0		Write HDLC flags 0x7E
	: V_HDLC_TRP	= 0		Select HDLC mode
	: V_TRP_IRQ	= 1		Enable FIFO
	: V_DATA_FLOW	= 6		Connect S/T ⟶ PCM
R_FIFO	: V_FIFO_DIR	= 1		Select receive FIFO
	: V_FIFO_NUM	= 1		Select FIFO #1 (connected to HFC-channel[B2,RX])
A_CON_HDLC[FIFO]	: V_IFF	= 0		Write HDLC flags 0x7E
	: V_HDLC_TRP	= 0		Select HDLC mode
	: V_TRP_IRQ	= 1		Enable FIFO
	: V_DATA_FLOW	= 6		Connect PCM ⟶ S/T

Register setup:			Part 3 (PCM loop)
R_B1_TX_SL	: V_B1_TX_SL	= 0	Connect HFC-channel[B1,TX] to PCM slot[0,TX]
	: V_B1_TX_ROUT	= '10'	
			STIO1 is data output
R_B1_RX_SL	: V_B1_RX_SL	= 0	Connect HFC-channel[B1,RX] to PCM slot[0,RX]
	: V_B1_RX_ROUT	= '11'	
			STIO1 is data input
R_B2_TX_SL	: V_B2_TX_SL	= 1	Connect HFC-channel[B2,TX] to PCM slot[1,TX]
	: V_B2_TX_ROUT	= '10'	
			STIO1 is data output
R_B2_RX_SL	: V_B2_RX_SL	= 1	Connect HFC-channel[B2,RX] to PCM slot[1,RX]
	: V_B2_RX_ROUT	= '11'	
			STIO1 is data input

Register setup:			Part 4 (PCM configuration)
R_PCM_MD0	: V_PCM_MD	= 1	Select PCM master mode
	: V_C4_POL	= 0	Pin F0IO is sampled on negative clock transition of C4IO
	: V_F0_NEG	= 0	
			F0IO with positive pulse polarity
	: V_F0_LEN	= 0	
			F0IO pulse width is one C4IO clock
	: V_SL_CODECA	= 0	(Value doesn't matter)
	: V_SL_CODECB	= 0	(Value doesn't matter)

5.11 Register description

> **Please note !**
>
> Due to a thorough revision of the HFC-S mini data sheet, some registers had to be renamed. Please see remarks on page 15.

5.11.1 Write only registers

R_ST_WR_STA (w) 0x30
(formerly **STATES**)

S/T state machine register

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

Bits	Reset value	Name	Description
3..0	0	**V_ST_SET_STA**	**Binary value of the new state** (NT: Gx, TE: Fx) V_ST_LD_STA must also be set to load the state.
4	1	**V_ST_LD_STA**	**Load the new state** '1' = load the prepared state (V_ST_SET_STA) and stop 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 state machine for automatic transitions according to state matrix (V_ST_SET_STA is ignored) **Note:** After writing an invalid state, the state machine goes to deactivated state (G1, F2).
6..5	0	**V_ST_ACT**	**Start activation / deactivation** '00' = no operation '01' = no operation '10' = start deactivation '11' = start activation This bitmap is automatically cleared after activation / deactivation.
7	0	**V_SET_G2_G3**	**Allow G2 to G3 transition** '0' = no operation '1' = allow transition from G2 to G3 in NT mode This bit is automatically cleared after the transition and has no function in TE mode.

(See initialization comment in Section 5.6 on page 108.)

R_ST_CTRL0 (w) 0x31
(formerly **SCTRL**)

Control register of the S/T interface, register 0

Bits	Reset value	Name	Description
0	0	**V_B1_EN**	**B1-channel transmit** '0' = B1 send data disabled (permanent '1's sent in activated states) '1' = B1 send data enabled
1	0	**V_B2_EN**	**B2-channel transmit** '0' = B2 send data disabled (permanent '1's sent in activated states) '1' = B2 send data enabled
2	0	**V_ST_MD**	**S/T interface mode** '0' = TE mode '1' = NT mode
3	0	**V_D_PRIO**	**D-channel priority** '0' = high priority 8/9 '1' = low priority 10/11
4	0	**V_SQ_EN**	**S/Q bits transmission** '0' = S/Q bits disabled '1' = S/Q bits (multiframe) enabled
5	0	**V_96KHZ**	**96 kHz test signal** '0' = normal operation '1' = send 96 kHz test signal (alternating zeros)
6	0	**V_TX_LI**	**Transmitter line setup** This bit must be configured depending on the used S/T module and circuitry to match the 400 Ω pulse mask test. '0' = capacitive line mode '1' = non capacitive line mode
7	0	**V_ST_STOP**	**Power down** '0' = normal operation, oscillator active '1' = power down, oscillator stopped The oscillator is restarted when AWAKE input becomes '1' or on any write access to the HFC-S mini.

R_ST_CTRL1 (w) 0x32
(formerly **SCTRL_E**)

Control register of the S/T interface, register 1

Bits	Reset value	Name	Description
0	0	**V_G2_G3_EN**	**Force G2 to G3 transition** Force automatic transition from G2 to G3 '0' = V_SET_G2_G3 of the register R_ST_WR_STA must be set to allow transitions from G2 to G3 '1' = transitions from G2 to G3 are allowed without V_SET_G2_G3 being set
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'
3	0	**V_E_IGNO**	**Ignore E-channel data** TE mode: '0' = normal operation '1' = D-channel always sends data regardless of the received E-channel bit NT mode: '0' = insert 8/9 ones between the flags of two D-channel frames (interframe fill) '1' = send D-channel frames without 8/9 ones in between (transparent mode) **Note:** This bit should be set to '1' in NT mode or for S/T-to-PCM / PCM-to-S/T data flow.
4	0	**V_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'
6..5	0	(reserved)	Must be '00'.
7	0	**V_B12_SWAP**	**Swap B-channels** '0' = normal operation '1' = swap B1- and B2-channel of the S/T interface

R_ST_CTRL2 (w) 0x33
(formerly **SCTRL_R**)

Control register of the S/T interface, register 2

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
5 . . 2	0	(reserved)	Must be '0000'.
6	0	**V_FO_INFO0**	**Force INFO 0** '0' = transmitter enabled according to the actual S/T state (normal operation) '1' = transmitter sends always INFO 0 (transmitter disabled, high impedance)
7	0	(reserved)	Must be '0'.

R_ST_SQ_WR (w) 0x34
(formerly **SQ_SEND**)

S/Q multiframe register

Bits	Reset value	Name	Description
3 . . 0	0	**V_ST_SQ_WR**	**S/Q bits** TE mode: bits [3 . . 0] are Q bits [Q1,Q2,Q3,Q4] NT mode: bits [3 . . 0] are S bits [S1,S2,S3,S4]
7 . . 4	0	(reserved)	Must be '0000'.

R_ST_CLK_DLY (formerly CLKDEL)	(w)	0x37

Clock control register of the S/T module

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

Bits	Reset value	Name	Description
3..0		V_ST_CLK_DLY	**S/T clock delay** **TE mode:** 4 bit delay value to adjust the 2 bit delay between receive and transmit direction. The delay of the external S/T interface circuit can be compensated. The lower the value the smaller the delay between receive and transmit direction. Typical values are 0xD..0xF. **NT mode:** Data sample point. The lower the value the earlier the input data is sampled. Recommended value is 0xC. For both modes the step size is 163 ns.
6..4		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. No compensation is chosen with V_ST_SMPL = 6 (recommended value). Step size is 163 ns.
7	0	(reserved)	Must be '0'.

Please note !

The register R_ST_CLK_DLY is not initialized after reset. It should be initialized as follows before activating the TE / NT state machine:

TE mode: 0x0D..0x0F (0x0F for S/T interface circuitry shown on page 117)

NT mode: 0x6C

R_ST_B1_TX (w) 0x3C
(formerly **B1_SEND**)

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.

Bits	Reset value	Name	Description
7..0		**V_ST_B1_TX**	**B1-channel data byte** Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

R_ST_B2_TX (w) 0x3D
(formerly **B2_SEND**)

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.

Bits	Reset value	Name	Description
7..0		**V_ST_B2_TX**	**B2-channel data byte** Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

R_ST_D_TX (w) 0x3E
(formerly **D_SEND**)

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.

Bits	Reset value	Name	Description
5..0	0	(reserved)	Must be 0.
7..6		**V_ST_D_TX**	**D-channel data byte** Data can be written during the non-processing phase (see V_PROC in register R_STATUS).

5.11.2 Read only registers

R_ST_RD_STA (formerly **STATES**)	(r)	0x30

S/T state machine register

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

Bits	Reset value	Name	Description
3..0	0	**V_ST_STA**	**S/T state** Binary value of current state (NT: Gx, TE: Fx)
4	0	**V_FR_SYNC**	**Frame synchronization** '0' = not synchronized '1' = synchronized
5	0	**V_T2_EXP**	**S/T timer T2 has expired** '1' = timer T2 has expired (NT mode only)
6	0	**V_INFO0**	**INFO0** '1' = receiving INFO0
7	0	**V_G2_G3**	**G2 to G3 transition allowed** '0' = no operation '1' = transition from G2 to G3 is allowed in NT mode This bit is automatically cleared after the transition and has no function in TE mode.

R_ST_SQ_RD (r) 0x34
(formerly **SQ_REC**)

S/Q multiframe register

Bits	Reset value	Name	Description
3 . . 0		**V_ST_SQ_RD**	**S/Q bits** TE mode: [3 . . 0] are S bits [S1,S2,S3,S4] NT mode: [3 . . 0] are Q bits [Q1,Q2,Q3,Q4]
4	0	**V_MF_RX_RDY**	**Receive multiframe ready** '1' = a complete S or Q multiframe has been received and is ready to get read Reading this register clears this bit.
6 . . 5		(reserved)	
7	0	**V_MF_TX_RDY**	**Transmit multiframe ready** '1' = ready to send a new S or Q multiframe Writing to R_ST_SQ_WR clears this bit.

R_ST_B1_RX (r) 0x3C
(formerly **B1_REC**)

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.

Bits	Reset value	Name	Description
7 . . 0		**V_ST_B1_RX**	**B1-channel data byte** Data is valid after a processing / non-processing transition (see V_PROC in register R_STATUS).

R_ST_B2_RX (r) 0x3D
(formerly **B2_REC**)

Receive register for 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.

Bits	Reset value	Name	Description
7..0		**V_ST_B2_RX**	**B2-channel data byte** Data is valid after a processing / non-processing transition (see V_PROC in register R_STATUS).

R_ST_D_RX (r) 0x3E
(formerly **D_REC**)

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.

Bits	Reset value	Name	Description
5..0		(reserved)	
7..6		**V_ST_D_RX**	**D-channel data byte** Data is valid after a processing / non-processing transition (see V_PROC in register R_STATUS).

R_ST_E_RX (r) 0x3F
(formerly **E_REC**)

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.

Bits	Reset value	Name	Description
5..0		(reserved)	
7..6		**V_ST_E_RX**	**E-channel data byte** Data is valid after a processing / non-processing transition (see V_PROC in register R_STATUS).

Chapter 6

PCM interface

Table 6.1: *Overview of the PCM related pins*

Number	Name	Description
30	C4IO	PCM double bit clock I/O
31	F0IO	PCM frame clock I/O (8 kHz)
32	STIO1	PCM data bus 1, I or O per time slot
33	STIO2	PCM data bus 2, I or O per time slot
34	F1_A	External CODEC A enable
34	C2O	PCM bit clock output
35	F1_B	External CODEC B enable
38	SYNC_I	Synchronization input (8 kHz)
43	SYNC_O	Synchronization output (8 kHz)

Second pin function:

Number	Name	Description
34	C2O	PCM bit clock output

Table 6.2: *Overview of the HFC-S mini PCM interface registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x14	R_PCM_MD0	159	0x18	R_F0_CNTL	173
0x15	R_PCM_MD1	161	0x19	R_F0_CNTH	173
0x16	R_PCM_MD2	163	0x28	R_CI_RX	173
0x20	R_B1_TX_SL	164	0x29	R_PCM_GCI_STA	174
0x21	R_B2_TX_SL	165	0x2A	R_MON1_RX	174
0x22	R_AUX1_TX_SL	166	0x2B	R_MON2_RX	175
0x23	R_AUX2_TX_SL	167	0x2C	R_B1_RX	175
0x24	R_B1_RX_SL	168	0x2D	R_B2_RX	175
0x25	R_B2_RX_SL	168	0x2E	R_AUX1_RX	176
0x26	R_AUX1_RX_SL	169	0x2F	R_AUX2_RX	176
0x27	R_AUX2_RX_SL	170			
0x28	R_CI_TX	170			
0x2A	R_MON1_TX	171			
0x2B	R_MON2_TX	171			
0x2C	R_B1_TX	171			
0x2D	R_B2_TX	172			
0x2E	R_AUX1_TX	172			
0x2F	R_AUX2_TX	172			

6.1 Overview

The HFC-S mini 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 as shown in Table 6.3. F0IO has always a frequency of 8 kHz. Each time slot has a width of eight bits.

Table 6.3: *PCM master mode*

V_PCM_DR in register R_PCM_MD1	C4IO clock	Number of time slots	Data rate
'00'	4.096 MHz	32	2 MBit/s
'01'	8.192 MHz	64	4 MBit/s
'10'	16.384 MHz	128	8 MBit/s
'11'			unused

The HFC-S mini 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.

Important !

As the F0IO pulse must be 170 ns at least in all cases where the S/T interface is used, the following restrictions must be fulfilled:

- In PCM master mode, V_F0_LEN in register R_PCM_MD0 must be '1' at 4 Mbit/s data rate. A data rate of 8 Mbit/s is not available.
- In PCM slave mode, any data rate of V_PCM_DR in register R_PCM_MD1 is selectable if the pulse length of F0IO is greater than 170 ns (V_F0_LEN = '1' in register V_F0_LEN).

6.2 PCM data flow

Every HFC-channel has a register which is used to assign a PCM time slot and to configure the data path between the HFC-channel and the STIO1 / STIO2 pins. This PCM data flow is shown in Figure 6.1.

HFC-channels and PCM time slot are always connected using the same data direction. Three data flow settings are available for each data direction, listed for R_B1_TX_SL and R_B1_RX_SL exemplarily:

- V_B1_TX_ROUT = '00' disables the HFC-channel and the PCM output buffers.
- V_B1_RX_ROUT = '00' disables the HFC-channel
- V_B1_TX_ROUT = '10' enables the STIO1 output buffer
- V_B1_RX_ROUT = '10' routes the data from STIO2

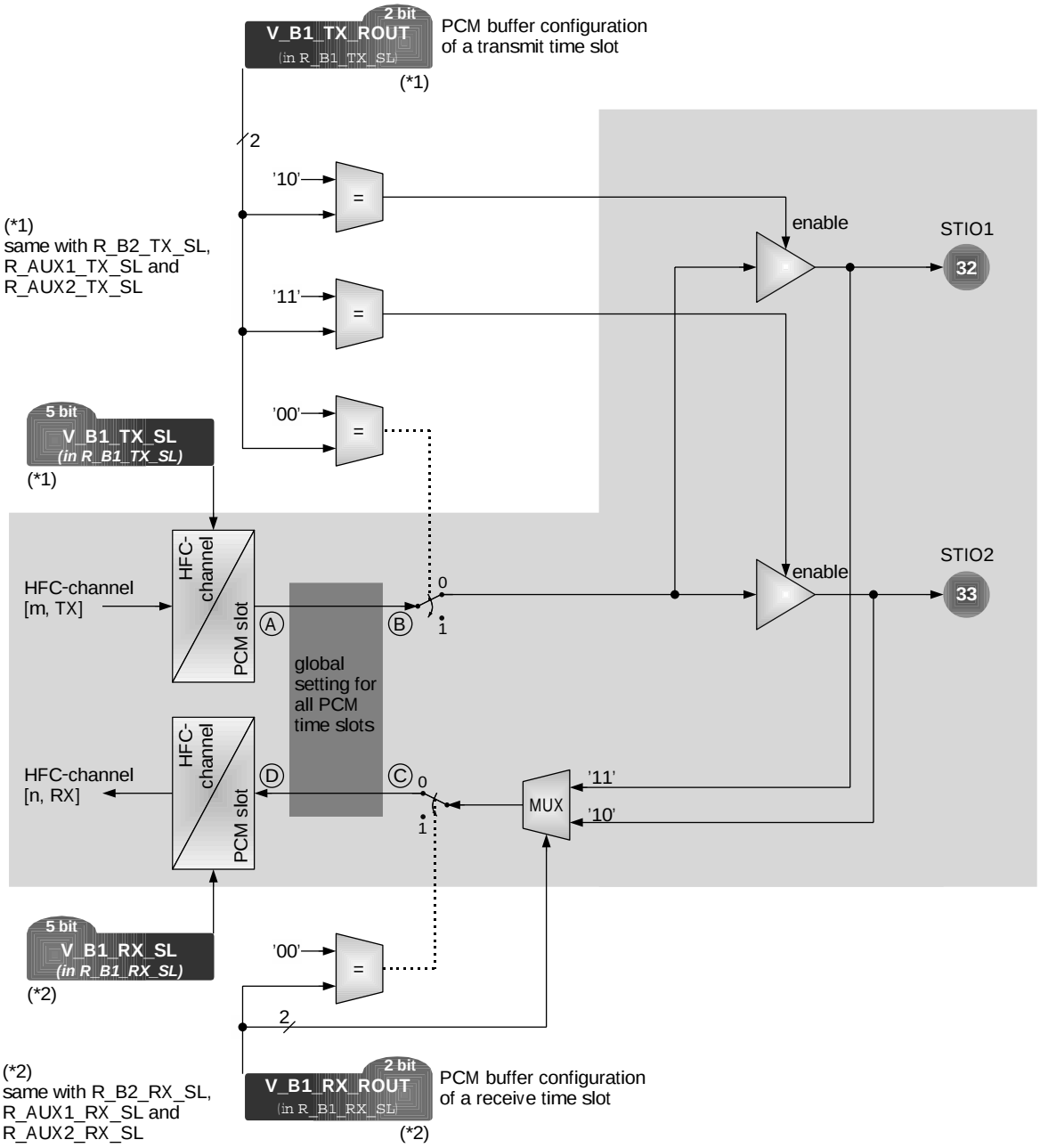


Figure 6.1: *PCM data flow for transmit and receive time slots (see Figure 6.2 for additional setting of all PCM time slots between Ⓐ..Ⓓ)*

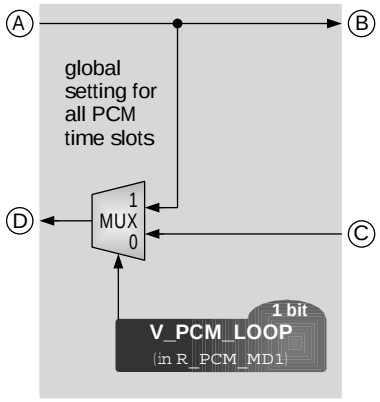


Figure 6.2: *Global settings for all PCM time slots (detail of Figure 6.1), normally used for test loop setup*

- V_B1_TX_ROUT = '11' enables the STIO2 output buffer
- V_B1_RX_ROUT = '11' routes the data from STIO1
- V_B1_TX_ROUT = '01' and V_B1_RX_ROUT = '01' are not used.

The PCM data flow programming is the same for R_B2_TX_SL / R_B2_RX_SL, R_AUX1_TX_SL / R_AUX1_RX_SL and R_AUX2_TX_SL / R_AUX2_RX_SL.

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.

6.3 PCM timing

The PCM interface of the HFC-S mini can operate either in slave mode or master mode. Slave mode is default selection after HFC-S mini reset.

To configure the HFC-S mini as PCM bus master, the bit V_PCM_MD of the register R_PCM_MD0 must be set to '1'. C4IO and F0IO signals are generated from the HFC-S mini in this case and both pins have output characteristic. The C4IO pulse polarity can be changed with V_C4_POL in register R_PCM_MD0.

Slave mode is selected with V_PCM_MD = '0'. C4IO and F0IO are input pins in slave mode. There must be external signals connected to C4IO and F0IO in this mode because these signals are used from the S/T interface and the flow controller as well.

The PCM bit rate is configured to either 2 MBit/s, 4 MBit/s or 8 MBit/s by the bitmap V_PCM_DR of the R_PCM_MD1 register. In 8 MBit/s mode the duty cycle of C4IO is $\frac{1}{3}/\frac{2}{3}$.

6.3.1 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 input and STIO2 as data output of the HFC-S mini. 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'.

Signal C2O is only available as second pin function on pin 34, when V_SL_CODECA = '11' in register R_PCM_MD0.

6.3.2 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 input and STIO2 as data output of the HFC-S mini. However, both pins can change their I/O characteristic with every PCM time slot.

The PCM frame starts at the first falling edge of C4IO after F0IO ⌜when V_F0_LEN = '0' in register R_PCM_MD0. F0IO can be of arbitrary length but must be at least one C4IO pulse long. F0IO is expected to start one C4IO clock earlier if bit V_F0_LEN = '1'.

If the S/T interface is synchronized from C4IO in NT mode, the frequency stability must be at least $\pm 10^{-4}$.

6.3.3 External CODECs

External CODECs can be connected to the PCM interface. The output pins F1_A and F1_B are used as CODEC enable signals. These signals are either compatible to OKI™ CODECs (signal F1_OKI in Figure 6.3) or they have the same shape as F0IO (signal F1_default in Figure 6.3). This shape type can separately be chosen with V_OKI_CODECA and V_OKI_CODECB in register R_PCM_MD2.

Figure 6.3 shows the CODEC enable signals located on PCM time slot #0. An arbitrary time slot can be chosen. The following example shows, how to connect the B2-channel to a CODEC on PCM time slot #12 using the F1_A pin to feed an OKI™-compatible shape signal:

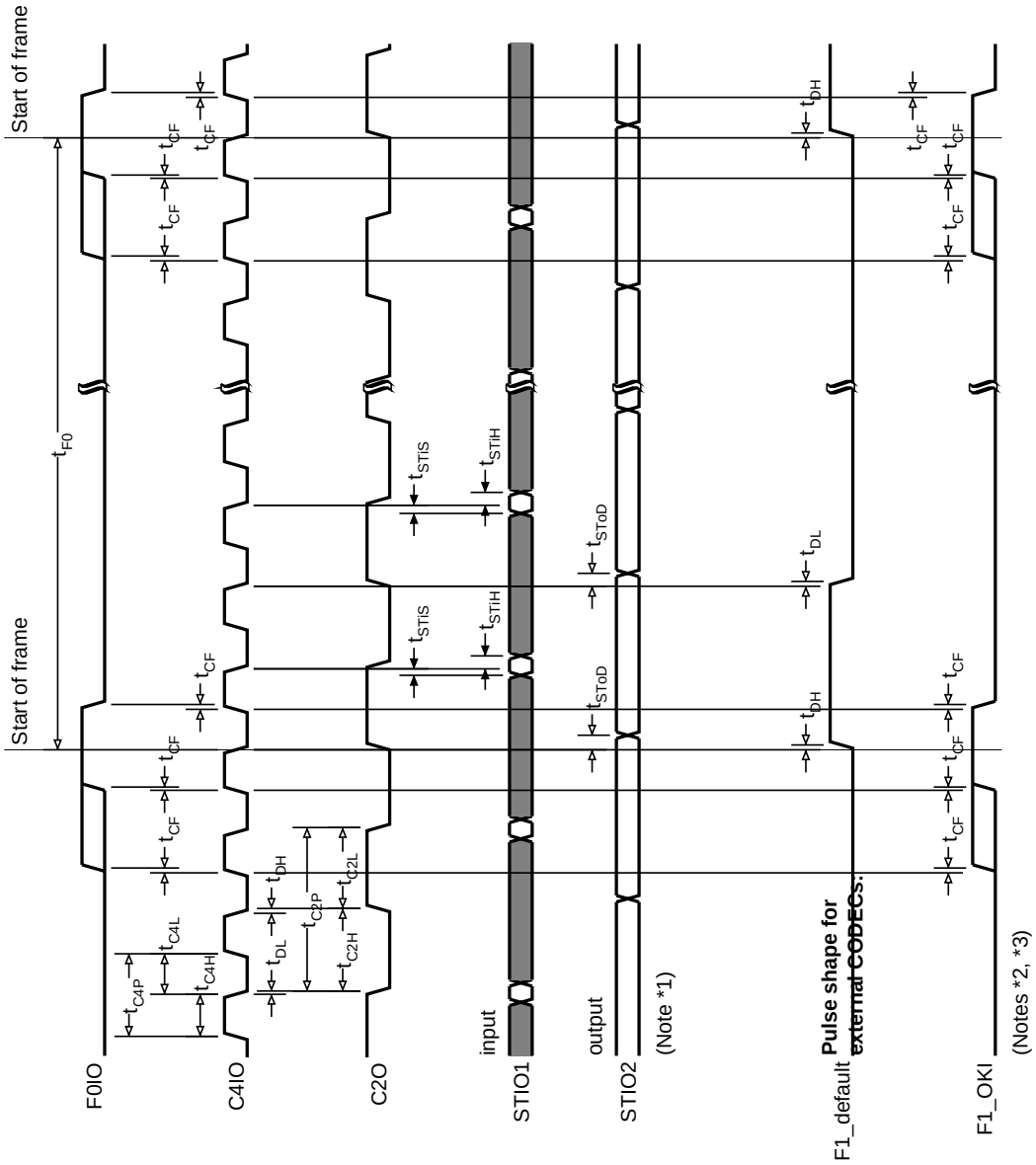


Figure 6.3: *PCM timing for master mode*

Notes to Figure 6.3:

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

*2 F1_default and F1_OKI are shown here in PCM time slot 0 exemplarily. Both signals can be assigned to an arbitrary time slot according to the actual PCM slot assigner programming.

*3 Pin F1_A can either be F1_default, F1_OKI or C2O.
Pin F1_B can either be F1_default or F1_OKI.

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, 50/50 duty cycle of system clock is assumed (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}	$2 \cdot t_C - 6 - t_{adj}$		$2 \cdot t_C + 6 + t_{adj}$	C2O high width
t_{C2L}	$2 \cdot t_C - 6 - t_{adj}$		$2 \cdot t_C + 6 + t_{adj}$	C2O low width
t_{C2P}	$4 \cdot t_C - 6 - t_{adj}$		$4 \cdot t_C + 6 + t_{adj}$	C2O 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_{DL}	3.92	5.04	7.54	C4IO ⌝ to C2O ⌝ delay
t_{DH}	3.87	5.07	7.69	C4IO ⌝ to C2O ⌟ 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 ⌝

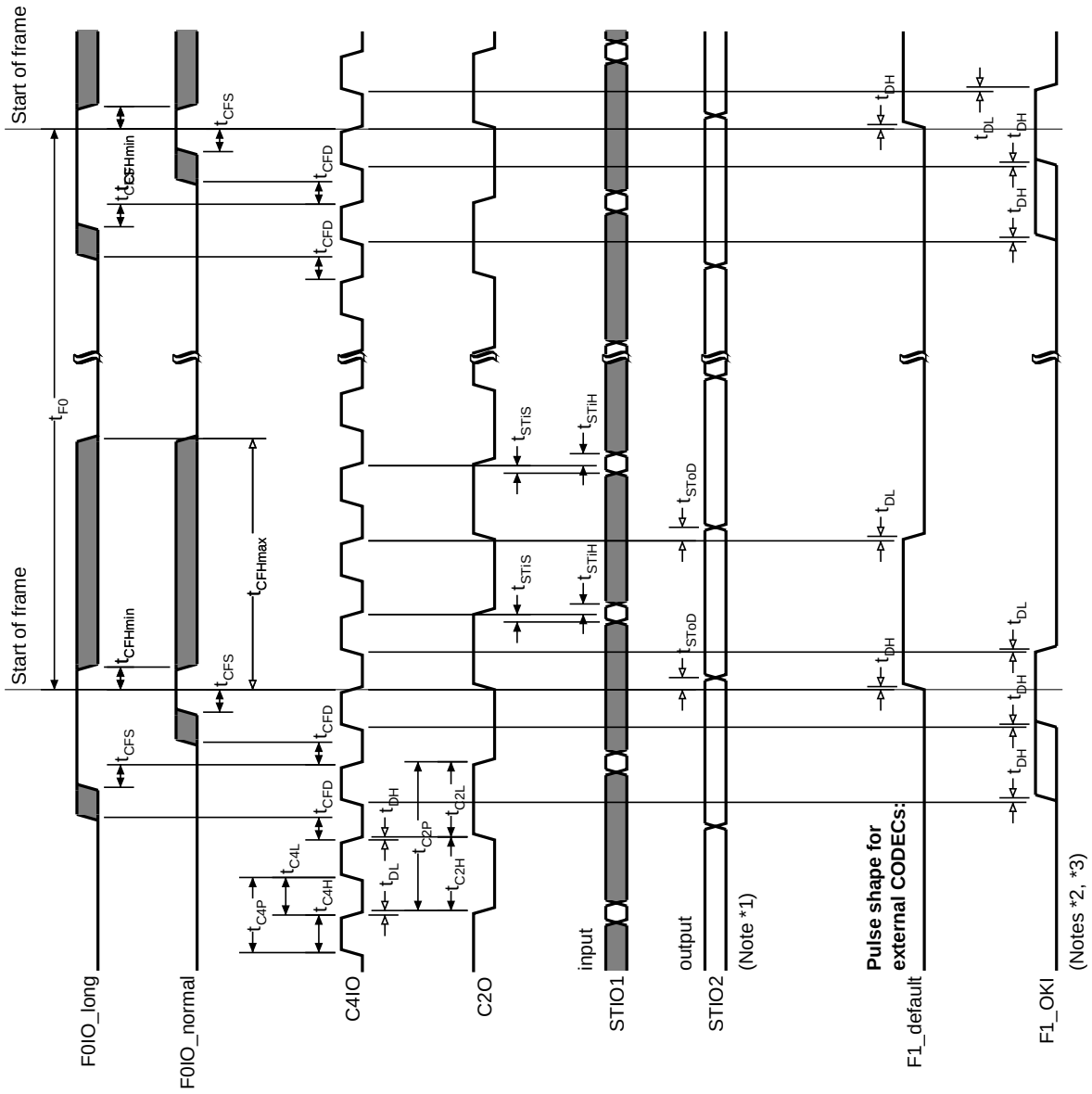


Figure 6.4: *PCM timing for slave mode*

Notes to Figure 6.4:

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

*2 F1_default and F1_OKI are shown here in PCM time slot 0 exemplarily. Both signals can be assigned to an arbitrary time slot according to the actual PCM slot assigner programming.

*3 Pin F1_A can either be F1_default, F1_OKI or C2O.
Pin F1_B can either be F1_default or F1_OKI.

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)
	122.058	122.070	122.082	4.096 MHz ±100 ppm C4IO clock for 2 MB/s
	61.029	61.035	61.041	8.192 MHz ±100 ppm C4IO clock for 4 MB/s
	30.515	30.518	30.521	16.384 MHz ±100 ppm 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}		$2 \cdot t_C$		C2O high width
t_{C2L}		$2 \cdot t_C$		C2O low width
t_{C2P}		$4 \cdot t_C$		C2O clock period
t_{F0}		125000		F0IO cycle time
t_{DL}	3.92	5.04	7.54	C4IO ↘ to C2O ↘ delay
t_{DH}	3.87	5.07	7.69	C4IO ↘ to C2O ↗ 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 ↘

Register setup:

R_B2_TX_SL	: V_B2_TX_SL	=	12	(assign HFC-channel[B2,TX] to PCM slot[12,TX])
	: V_B2_TX_ROUT	=	'10'	(STIO1 is data output)
R_B2_RX_SL	: V_B2_RX_SL	=	12	(assign HFC-channel[B2,RX] to PCM slot[12,RX])
	: V_B2_RX_ROUT	=	'10'	(STIO2 is data input)
R_PCM_MD0	: V_SL_CODECA	=	1	(move CODEC A shape signal to the PCM time slot which is assigned to HFC-channel[B2,RX])
R_PCM_MD2	: V_OKI_CODECA	=	1	(select OKI™-compatible shape signal for CODEC A)

Please note !

There is no F1_A shape signal for HFC-channel[AUX2].
V_SL_CODECA = '11' is used to route the C2O clock to pin 34 instead.

The pulse polarity of F1_A and F1_B is always the same as the F0IO polarity. This can be changed with V_F0_NEG in register R_PCM_MD0.

Hint

Automatic CODEC-to-CODEC connection (internal loop)

An automatic CODEC-to-CODEC connection can be set up. This requires HFC-channel[AUX1] and [AUX2] in mirror mode. Then, received data from a PCM time slot is mirrored in the assigned HFC-channel and transmitted on another PCM time slot.

6.4 PCM clock synchronization

6.4.1 Overview

The PCM clock synchronization is shown in Figure 6.5. It is associated with the S/T clock synchronization which is shown in Figure 5.3 on page 112.

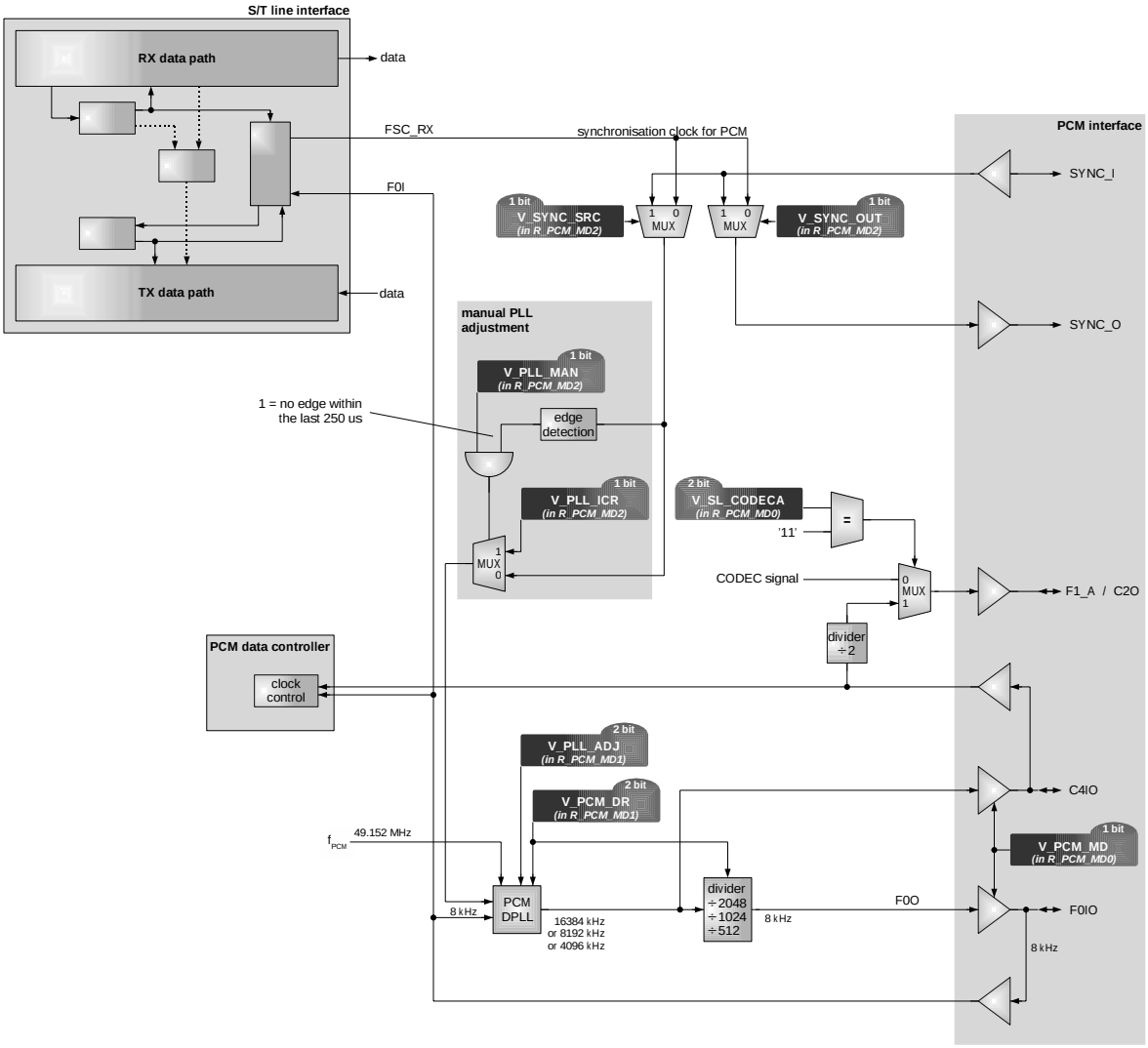


Figure 6.5: *PCM clock synchronization (see Figure 5.1 on page 104 for details on the S/T interface module)*

6.4.2 Synchronization source selection

Either the interface in TE mode or the SYNC_I signal can be chosen as synchronization source with an appropriate value in the bitmap V_SYNC_SRC of register R_PCM_MD2. If no synchronization source is available, the PCM DPLL is free-running.

6.4.3 Synchronization pulse specification

The SYNC_I signal must have a frequency of 8 kHz with an arbitrary duty cycle. Only the rising transition is used for synchronization.

The FSC_RX signal is generated from the incoming data stream. Is can be used as synchronization source for the PCM DPLL and, moreover, it can be feed to the SYNC_O pin. The signal has a frequency of 8 kHz and a minimum high pulse width of nominal one bit of the S/T data stream which is 5.208 μs.

6.4.4 PLL programming for F0IO generation

C4IO is adjusted from the PCM DPLL (see Figure 6.5 on page 144) during the last PCM time slot to synchronize the PCM interface with the S/T 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 the 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 HFC 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.

6.4.5 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.

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.

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

6.4.6 C2O generation

The C2O 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, C2O has its rising edge at the start of a frame. From this follows that C4IO has a falling edge with every edge of C2O .

6.4.7 Application examples for HFC-S mini synchronization schemes

Flexible synchronization schemes can be implemented with the clock pins SYNC_I and SYNC_O of HFC-S mini. 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, typically.

When multiple HFC-S mini are used within a system, all S/T and PCM clocks must be synchronized to a single synchronization source. When an S/T interface operates in TE mode and is connected to the central office, the synchronization source is obtained from the central office.

6.4.7.1 An existing system with internal PCM bus has to be expanded by an ISDN interface

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

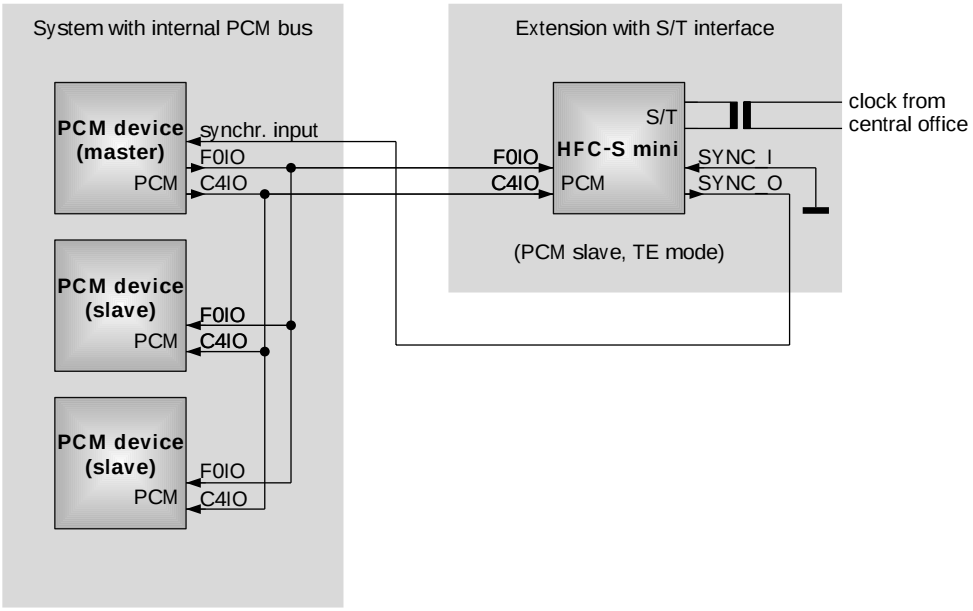


Figure 6.6: *Expanding an existing system with an S/T port (HFC-S mini as PCM slave)*

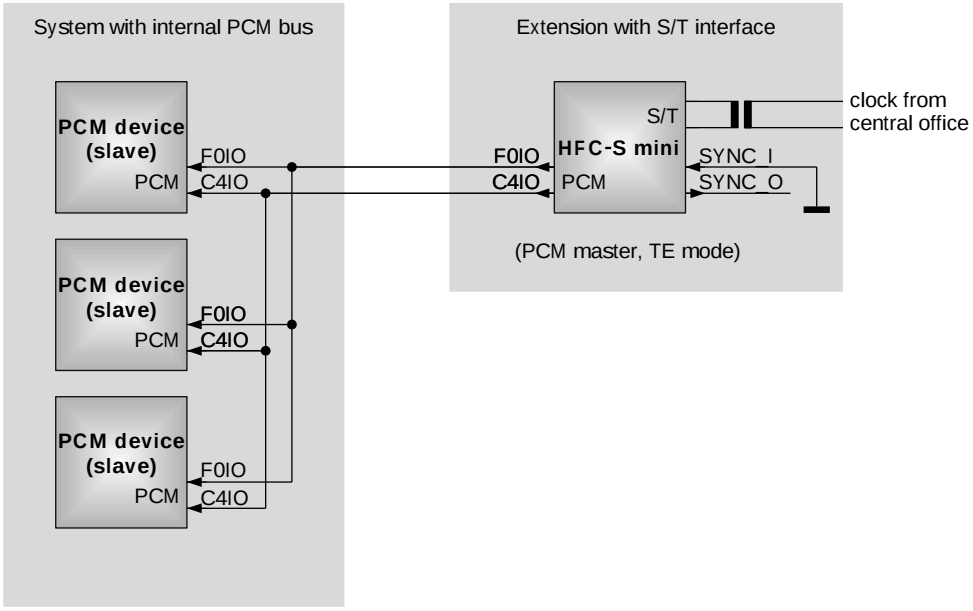


Figure 6.7: *Expanding an existing system with an S/T port (HFC-S mini as PCM master)*

The existing system can keep the PCM master when there is a synchronization input (Figure 6.6). This must be connected to the SYNC_O pin of HFC-S mini to synchronize the whole system to the clock derived from the received S/T signal.

SYNC_O is either derived from the central office or the DPLL of HFC-S mini is free-running. In both cases, HFC-S mini is the synchronization source for the whole system.

When there is no synchronization input available at the existing system, HFC-S mini must operate as PCM master and all other PCM devices in the system must be PCM slaves (Figure 6.7). Again, HFC-S mini is the synchronization source for the whole system.

6.4.7.2 Application with multiple HFC-S mini

Multiple HFC-S mini can be interconnected to build up a multi S/T 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 HFC-S mini can be in TE mode. The synchronization signals are daisy chained. The last HFC-S mini 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 HFC-S mini 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 HFC-S mini can be in PCM master mode.

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

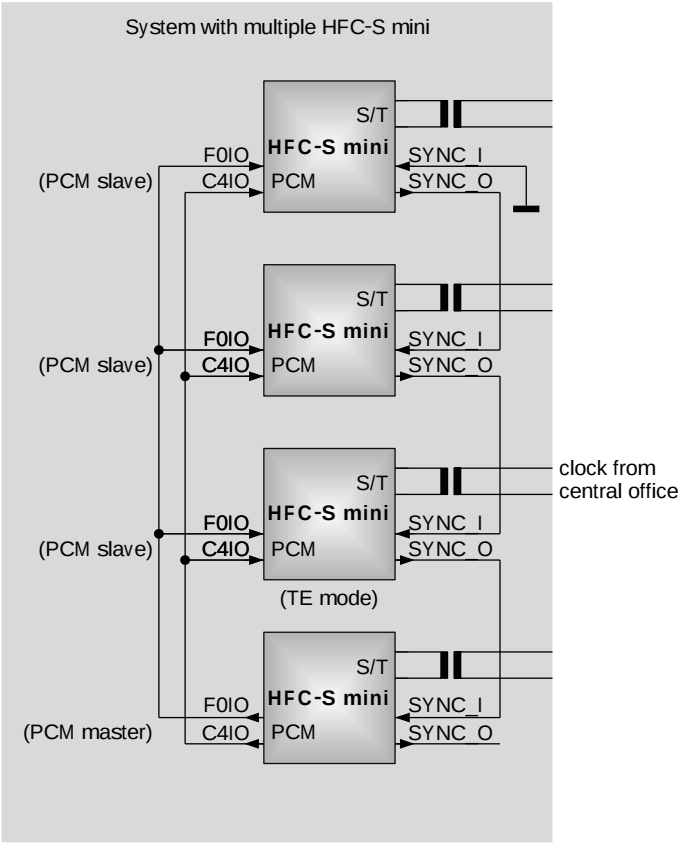


Figure 6.8: *Multiple HFC-S mini synchronized with an open loop of SYNC_O /SYNC_I*

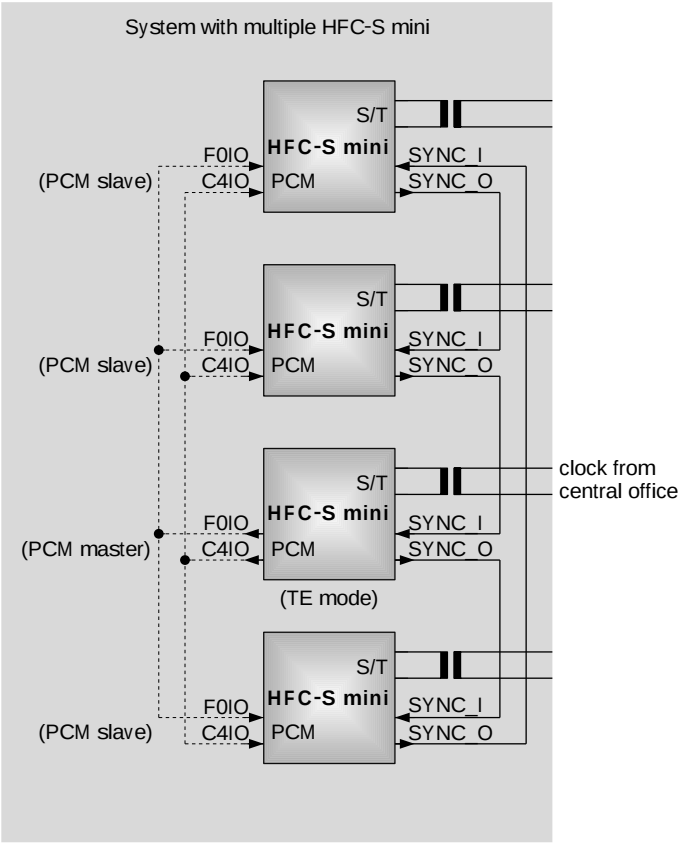


Figure 6.9: *Multiple HFC-S mini synchronized with an closed loop of SYNC_O /SYNC_I*

6.5 GCI interface function

6.5.1 Overview

The HFC-S mini is equipped with a simple GCI [2] (also known as IOM™-2 [3]) controller to support interconnection to U-chips or external CODECs.

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 [4]. The HFC-S mini GCI functionality has been implemented in respect to the IOM™-2 specification [2]. Only Line-Card mode is supported by HFC-S mini and only GCI-channel 0 (PCM time slots 0 . . 3) can be used.

The interconnection between the HFC-S mini and a GCI device uses four wires, typically:

- C4IO : Double bit rate clock
- F0IO : 8 kHz frame signal
- STIO1 : Data from HFC-S mini to the GCI device
- STIO2 : Data from the GCI device to HFC-S mini

> **Please note !**
>
> HFC-S mini can only operate as GCI master. Nevertheless, its PCM interface can be set to slave mode. In this case, another PCM device (beside the GCI slave device) connected to the HFC-S mini must operate in PCM master mode to feed the PCM clocks.

6.5.2 GCI frame structure

The GCI frame is embedded into the PCM time slot structure as shown in Figure 6.10. GCI uses the first four PCM time slots in a special way, all other PCM time slots are accessible as usual. PCM30 must be selected with V_PCM_DR = '00' in register R_PCM_MD1, because GCI conform devices expect a double bit clock of 4.096 MHz.

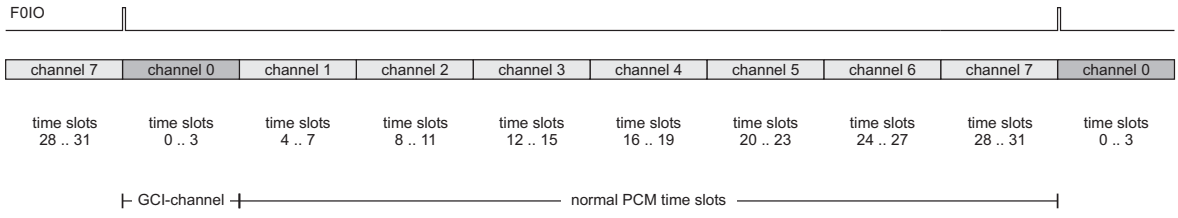


Figure 6.10: *Location of the GCI frame within the PCM time slots*

[2] GCI = General Circuit Interface
[3] IOM™-2= ISDN Oriented Modular revision 2, trademark of Infineon Technologies AG
[4] Alcatel (France), Siemens (Germany), Italtel (Italy) and Plessey (UK)

The GCI frame has a length of 4 bytes and is located at PCM time slots 0 . . 3. GCI functionality must be enabled with V_GCI_EN = '1' in register R_PCM_MD1. Then PCM time slots 2 and 3 are occupied and cannot be assigned to any HFC-channel.

The binary organization of the GCI frame is shown in Figure 6.11. Time slots 0 and 1 are used for B1- and B2-channel data as usual. Time slot 2 is occupied by the monitor channel and time slot 3 contains D-channel data, command / indication bits and the handshake bits MR and MX.

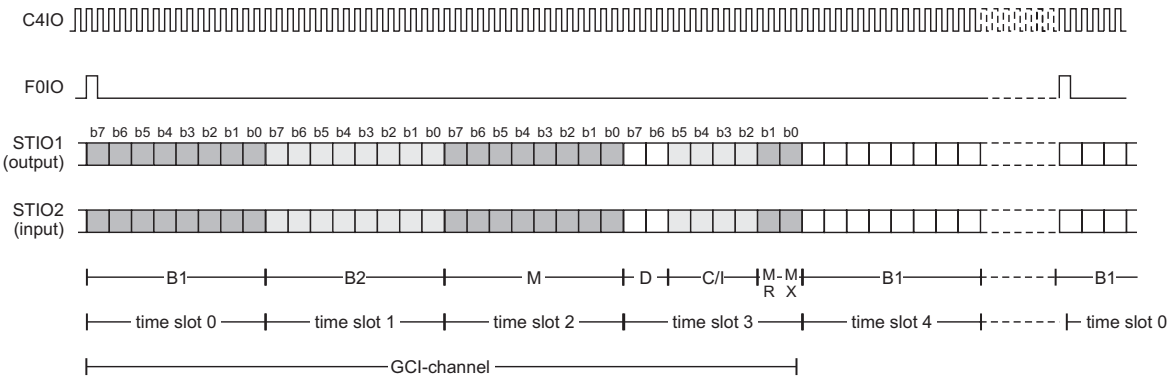


Figure 6.11: *Single channel GCI format*

Table 6.6: *Legend of figure 6.11*

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
MX	Handshake bit for the monitor channel

The PCM slot assigner can be used to allocate any HFC-channel to time slots 0 and 1. In case of an GCI device connected to the HFC-S mini, B1- and B2-channels should be assigned to these time slots because data channels are expected in the first two bytes of the CGI-frame.

6.5.3 GCI register programming

6.5.3.1 Enable CGI functionality

The GCI functionality is disabled after HFC-S mini 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_PCM_MD1. Then GCI occupies PCM time slots 2 and 3.

6.5.3.2 Monitor bytes

The HFC-S mini supports only 16-bit monitor messages [5].

Monitor data for the transmit direction must be written into the registers R_MON1_TX and R_MON2_TX. They are transmitted in time slot 2 with regard to the protocol described in Section 6.5.4. When the second byte is in transmit process, bit V_MON_TXR is set to '1' in register R_PCM_GCI_STA to indicate that the next two bytes can be written into the R_MON1_TX and R_MON2_TX registers.

If there are no further monitor bytes to be transmitted, the idle pattern 0xFF is transmitted.

Received monitor bytes can be read from the registers R_MON1_RX and R_MON2_RX. When two bytes have been received, an interrupt event is generated, i.e. bit V_MON_RX_IRQ is set to '1' in register R_MISC_IRQ. V_MON_RX_IRQ is set to '1' even if the interrupt mask bit V_MON_IRQMSK is not set.

When interrupt line signalling is desired, it must be enabled with V_MON_IRQMSK = '1' in register R_MISC_IRQMSK.

Transmitting and receiving monitor bytes is coordinated by the GCI controller of the HFC-S mini. This function is described in Section 6.5.4.

6.5.3.3 Command / indication channel (C/I)

Command / indication is used to transmit a command from the HFC-S mini to the connected GCI-device and to receive status information (indication).

C/I-bits are transmitted in every PCM frame continuously in both directions.

The HFC-S mini transmits the command that is written into the bitmap V_GCI_C of the register R_CI_TX.

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 can be read from V_CI_IRQ. This bit shows the interrupt condition even if the interrupt mask is not set.

The GCI controller does not interpret the C/I-bits. Indication bits must be processed from the host processor.

[5] CGI devices that can handle other message length are also available. They must be used with 16-bit monitor data transmissions if connected to the HFC-S mini.

6.5.3.4 Restrictions for used / unused GCI functionality

Due to the requirements of GCI conform devices, some restrictions must be considered when the GCI functionality of the HFC-S mini is used:

- PCM30 must be selected, because GCI conform devices expect a double bit clock of 4.096 MHz.
- The I/O behavior of the pins STIO1 and STIO2 is not programmable for time slots 2 and 3. STIO1 is always output and STIO2 is always input.
- Time slots 0 and 1 should be programmed to have the same I/O behavior as time slots 2 and 3.
- Time slot 0 is expected to transfer B1-channel data and time slot 1 is expected to transfer B2-channel data.

Table 6.7 shows the programming values to fulfill the requirements described above.

Table 6.7: *Typical programming values to operate with GCI conform devices*

Register	Bit	Value	Function
R_PCM_MD1	V_PCM_DR	'00'	PCM30, C4IO = 4.096 MHz
R_B1_TX_SL	V_B1_TX_SL	'0'	Time slot 0 for HFC-channel[B1,TX]
R_B1_TX_SL	V_B1_TX_ROUT	'10'	STIO1 is output
R_B2_TX_SL	V_B2_TX_SL	'1'	Time slot 1 for HFC-channel[B2,TX]
R_B2_TX_SL	V_B2_TX_ROUT	'10'	STIO1 is output
R_B1_RX_SL	V_B1_RX_SL	'0'	Time slot 0 for HFC-channel[B1,RX]
R_B1_RX_SL	V_B1_RX_ROUT	'10'	STIO2 is input
R_B2_RX_SL	V_B2_RX_SL	'1'	Time slot 1 for HFC-channel[B2,RX]
R_B2_RX_SL	V_B2_RX_ROUT	'10'	STIO2 is input

Important!

D-channel data to be transmitted from the PCM interface is automatically mapped to PCM time slot 3.

Due to a design error, D-channel data received on the PCM slot[3,RX] is not mapped to HFC-channel[AUX1,RX] when GCI function is enabled. The problem can be solved this way:

1. HFC-S mini assigns its HFC-channel[AUX2,RX] to PCM slot[3,RX].
2. The host processor can read D-channel data from the FIFO assigned to the HFC-channel[AUX2,RX].
3. Data received on HFC-channel[AUX1,RX] must be ignored.

The data flow has to be configured as follows:

Register setup:

A_CON_HDLC	: V_IFF	=	1	('1's as inter frame fill)
	: V_HDLC_TRP	=	0	(HDLC mode)
	: V_TRP_IRQ	=	1	(enable FIFO)
	: V_DATA_FLOW	=	'001'	(PCM → FIFO)
A_HDLC_PAR	: V_BIT_CNT	=	2	(Process 2 bits, 16 kbit/s)
	: V_START_BIT	=	0	(Start bit 0)

When CGI functionality is not used, received data on time slots 2 and 3 are extracted to the registers R_MON1_RX, R_MON2_RX and R_CI_RX nevertheless. For this reason, the interrupt masks V_CI_IRQMSK and V_MON_IRQMSK in register R_MISC_IRQMSK must stay in their reset state '0' to avoid senseless interrupts.

6.5.4 GCI protocol

Monitor bytes are transmitted and received under the control of the handshake bits MX and MR. The HFC-S mini supports only sequences of two monitor bytes. The handshake bits are automatically handled by the GCI controller.

Monitor data bytes must not be 0xFF because this value is used as *idle pattern*.

6.5.4.1 HFC-S mini transmit procedure

Figure 6.12 shows the transmit procedure of two monitor bytes.

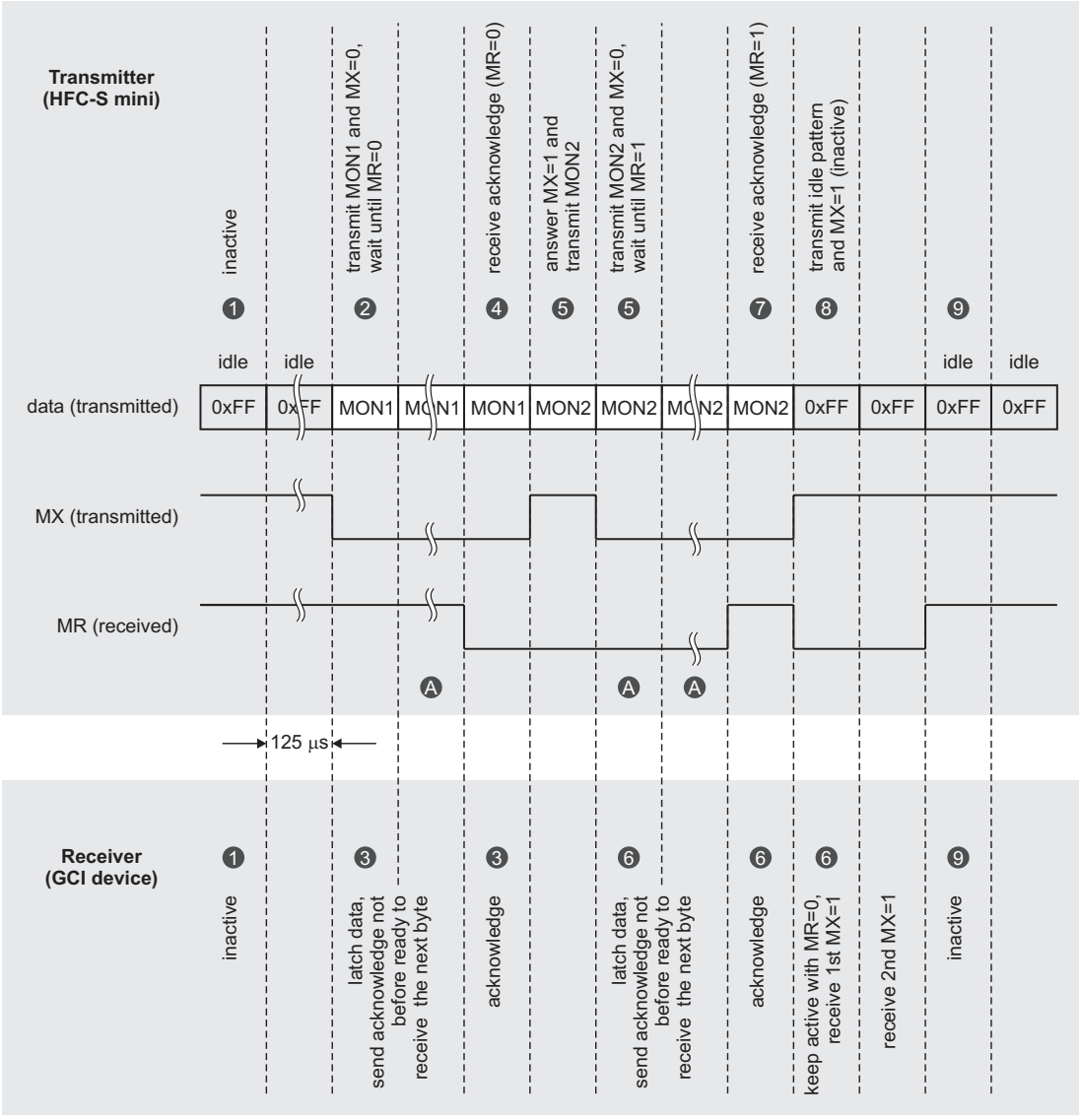


Figure 6.12: *GCI protocol for monitor bytes transmission*

❶ Beginning with idle state, no monitor byte is pending, MX and MR are both '1'. The monitor byte has the value 0xFF in this state which means 'idle pattern'.

❷ A write access to the register R_MON2_TX starts the transmit sequence. The first monitor byte must have been written into the register R_MON1_TX before.

The first monitor byte is transmitted within the next PCM time slot 2 and MX 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 = '0'.

❸ The receiver latches the first monitor byte. This can take an arbitrary number of PCM cycles. An acknowledge is send when the receiver is ready for the next byte.

❹ The transmitter gets acknowledge with MR = '0'.

❺ The transmitter answers the acknowledge signal with MX = '1' for one cycle and sets MX = '0' afterwards to keep the monitor channel active. The second monitor byte is transmitted at the same time as MX = '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 is send when the receiver is ready for the next byte.

❼ The transmitter receives acknowledge with MR = '1' for one cycle.

❽ As the HFC-S mini supports only 2-byte sequences on the monitor channel, the transmitter terminates the transmission with MX = '1'. Idle pattern 0xFF is send continuously. The monitor channel of the transmitter is in inactive state.

❾ When the receiver reads MX = '1' for two cycles, its monitor channel goes to inactive state with MR = '1'. Now the receiver is in invalid state, too. The monitor channel is in idle state.

Table 6.8 summarizes the rules for the handshake signals MX and MR when HFC-S mini transmits monitor bytes.

Table 6.8: *Rules for the handshake signals MX and MR when HFC-S mini transmits monitor bytes*

MX	MR	Monitor channel state
'1'	'1'	Idle
'0'	'0'	Transmitter and receiver are both active
'0'	'1' (once)	Acknowledge of the second monitor byte
'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

Please note !

The HFC-S mini handles all monitor bytes with value 0xFF to be idle pattern. So no monitor byte 0xFF can be transmitted. This must be taken into account when using the GCI function of HFC-S mini.

6.5.4.2 HFC-S mini receive procedure

The monitor channel is full duplex and operates similar in the opposite direction. The HFC-S mini recognizes idle state with the specified handshake signals and accepts every idle pattern.

Double last look criterion is implemented for the receive monitor channel.

Data bytes must not be 0xFF because this value is recognized to be *idle pattern*.

The HFC-S mini never aborts any message when it is receiver of the monitor channel. Every message has a sequence length of 2 bytes.

Figure 6.13 shows the receive procedure of two monitor bytes.

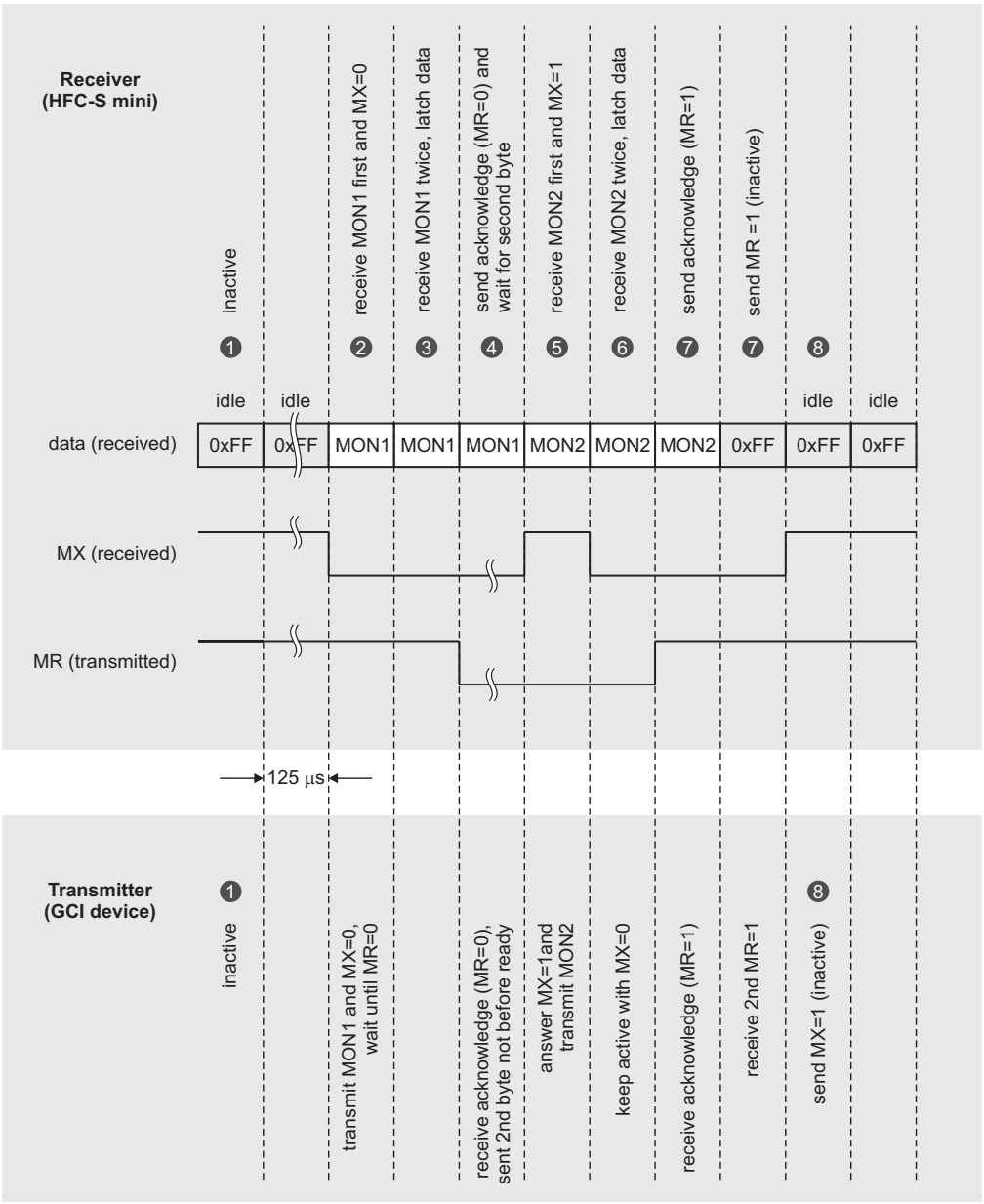


Figure 6.13: *GCI protocol for monitor bytes receiving*

❶ The monitor channel is idle when MX = '1' and MR = '1'. Any received data pattern is ignored.

❷ The first monitor byte is received if MX = '0'. Due to *Double last look criterion* it is not yet stored in register R_MON1_RX.

❸ The first monitor byte is received again (*Double last look criterion*) while MX remains at '0' level. The byte is latched in register R_MON1_RX now.

❹ Acknowledge is send in the next 125 μs cycle with MR = '0'.

❺ The second monitor byte is received for the first time when MX = '1' (answer to acknowledge) for one cycle.

❻ The second monitor byte is received again (*Double last look criterion*) while MX returned to '0' level. The byte is latched in register R_MON2_RX now.

❼ Acknowledge is send in the next 125 μs cycle with MR = '1'. The monitor channel of the HFC-S mini goes to inactive state with MR = '1' again in the next cycle.

❽ The transmitter gets MR = '1' within two consecutive cycles which forces the monitor channel to inactive state with MX = '1'. The monitor channel of the transmitter is in inactive state (idle state).

The rules for the handshake signals MX and MR when HFC-S mini receives monitor bytes are summarized in Table 6.9.

Table 6.9: *Rules for the handshake signals MX and MR when HFC-S mini receives monitor bytes*

MX	MR	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 acknowledge
'1' (repeated)	'0'	Impossible

Please note !

The HFC-S mini handles all monitor bytes with value 0xFF to be idle pattern. So no monitor byte 0xFF can be received. This must be taken into account when using the GCI function of HFC-S mini.

6.6 Register description

> **Please note !**
>
> Due to a thorough revision of the HFC-S mini data sheet, some registers had to be renamed. Please see remarks on page 15.

6.6.1 Write only registers

R_PCM_MD0 (formerly **MST_MODE0**)	**(w)**	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 in master mode** '0' = active for one C4IO clock (244 ns at 2 Mbit/s) '1' = active for two C4IO clocks (488 ns at 2 Mbit/s)

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
5..4	0	**V_SL_CODECA**	**Selection of the PCM time slot for the CODEC A shape signal on pin F1_A** The shape signal on pin F1_A (for CODEC A) can be moved to a PCM time slot which is assigned to any HFC-channel. Only receive HFC-channels are used for time slot selection. CODEC operation requires the same PCM time slot number for the corresponding transmit data direction. '00' = PCM time slot which is assigned to HFC-channel[B1,RX] (V_B1_RX_SL) '01' = PCM time slot which is assigned to HFC-channel[B2,RX] (V_B2_RX_SL) '10' = PCM time slot which is assigned to HFC-channel[AUX1,RX] (V_AUX1_RX_SL) '11' = no CODEC shape signal available, C2O clock is feeded on pin 34 instead
7..6	0	**V_SL_CODECB**	**Selection of the PCM time slot for the CODEC B shape signal on pin F1_B** The shape signal on pin F1_B (for CODEC A) can be moved to a PCM time slot which is assigned to any HFC-channel. Only receive HFC-channels are used for time slot selection. CODEC operation requires the same PCM time slot number for the corresponding transmit data direction. '00' = PCM time slot which is assigned to HFC-channel[B1,RX] (V_B1_RX_SL) '01' = PCM time slot which is assigned to HFC-channel[B2,RX] (V_B2_RX_SL) '10' = PCM time slot which is assigned to HFC-channel[AUX1,RX] (V_AUX1_RX_SL) '11' = PCM time slot which is assigned to HFC-channel[AUX2,RX] (V_AUX2_RX_SL)

R_PCM_MD1 (w) 0x15
(formerly **MST_MODE1**)

PCM mode, register 1

Bits	Reset value	Name	Description
0	0	**V_AUX1_MIR**	**AUX1 channel mirroring** '0' = disable AUX1 channel mirroring '1' = mirror AUX1 receive data to AUX1 transmit channel
1	0	**V_AUX2_MIR**	**AUX2 channel mirroring** '0' = disable AUX2 channel mirroring '1' = mirror AUX2 receive data to AUX2 transmit channel
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 half clock cycle of CLKI '01' = C4IO clock is adjusted in the last time slot of the PCM frame 3 times by one half clock cycle of CLKI '10' = C4IO clock is adjusted in the last time slot of the PCM frame twice by one half clock cycle of CLKI '11' = C4IO clock is adjusted in the last time slot of the PCM frame once by one half clock cycle of CLKI
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), long F0IO signal required (> 170 ns, bit 3 of R_PCM_MD0 must be set) '10' = 8 MBit/s (C4IO is 16.384 MHz, 128 time slots), only in PCM slave mode and with long F0IO signal (> 170 ns, bit V_F0_LEN of register R_PCM_MD0 must be set) '11' = unused Every time slot exists in receive and transmit 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.

(continued on next page)

(continued from previous page)

Bits	Reset value	Name	Description
7	0	**V_GCI_EN**	**Enable GCI transmit time slots** '0' = PCM time slots 2 and 3 can be used for normal data transmission '1' = GCI functionality is enabled, i.e. time slots 2 and 3 are used for monitor bytes, D-channel, C/I-bits and handshake bits **Note:** The I/O function of the pins STIO1 and STIO2 cannot be configured for time slots 2 and 3 when GCI functionality is enabled. STIO1 is always output and STIO2 is always input in this case.

R_PCM_MD2 (w) 0x16
(formerly **MST_MODE2**)

PCM mode, register 2

Bits	Reset value	Name	Description
0	0	**V_OKI_CODECA**	**CODEC enable signal on pin F1_A** '0' = default signal shape '1' = generate a signal shape compatible to OKI^{TM} CODECs
1	0	**V_OKI_CODECB**	**CODEC enable signal on pin F1_B** '0' = default signal shape '1' = generate a signal shape compatible to OKI^{TM} CODECs
2	0	**V_SYNC_SRC**	**PCM PLL synchronization source selection** '0' = received frame synchronization pulse from S/T interface '1' = SYNC_I input (8 kHz) **Note:** A frame synchronization pulse is only generated in TE mode and when a valid frame synchronization is achieved (states F6 or F7)
3	0	**V_SYNC_OUT**	**SYNC_O signal source selection** '0' = received frame synchronization pulse from S/T interface '1' = SYNC_I is connected to SYNC_O **Note:** A frame synchronization pulse is only generated in TE mode and when a valid frame synchronization is achieved (states F6 or F7)
5..4	0	**V_SL_BL**	**Block selection of PCM time slots** '00' = slots 0..31 accessable '01' = slots 32..63 accessable (only with 4 MBit/s or 8 MBit/s PCM data rate) '10' = slots 64..95 accessable (only with 8 MBit/s PCM data rate) '11' = slots 96..127 accessable (only with 8 MBit/s PCM data rate)
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 the bitmap V_PLL_ADJ of register R_PCM_MD1 '1' = PCM frame time is increased as selected by the bitmap V_PLL_ADJ of 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_OUT 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_B1_TX_SL (formerly **B1_SSL**) **(w)** 0x20

Slot assigner for HFC-channel[B1,TX] and PCM time slot configuration

Bits	Reset value	Name	Description
4..0	0	**V_B1_TX_SL**	**PCM time slot selection for HFC-channel[B1,TX]** The selected time slot in the range V_B1_TX_SL = 0..31 is added to the time slot block which is selected with bitmap V_SL_BL in register R_PCM_MD2.
5	0	(reserved)	Must be '0'.
7..6	0	**V_B1_TX_ROUT**	**PCM output buffer configuration** '00' = data transmission from HFC-channel[B1,TX] disabled, output buffer disabled '01' = not used '10' = output buffer for STIO1 enabled '11' = output buffer for STIO2 enabled

R_B2_TX_SL (w) 0x21
(formerly **B2_SSL**)

Slot assigner for HFC-channel[B2,TX] and PCM time slot configuration

Bits	Reset value	Name	Description
4..0	0	**V_B2_TX_SL**	**PCM time slot selection for HFC-channel[B2,TX]** The selected time slot in the range V_B2_TX_SL = 0..31 is added to the time slot block which is selected with bitmap V_SL_BL in register R_PCM_MD2.
5	0	(reserved)	Must be '0'.
7..6	0	**V_B2_TX_ROUT**	**PCM output buffer configuration** '00' = data transmission from HFC-channel[B2,TX] disabled, output buffer disabled '01' = not used '10' = output buffer for STIO1 enabled '11' = output buffer for STIO2 enabled

R_AUX1_TX_SL (formerly AUX1_SSL)	(w)	0x22

Slot assigner for HFC-channel[AUX1,TX] and PCM time slot configuration

Bits	Reset value	Name	Description
4..0	0	V_AUX1_TX_SL	**PCM time slot selection for HFC-channel[AUX1,TX]** The selected time slot in the range V_AUX1_TX_SL = 0..31 is added to the time slot block which is selected with bitmap V_SL_BL in register R_PCM_MD2.
5	0	(reserved)	Must be '0'.
7..6	0	V_AUX1_TX_ROUT	**PCM output buffer configuration** '00' = data transmission from HFC-channel[AUX1,TX] disabled, output buffer disabled '01' = not used '10' = output buffer for STIO1 enabled '11' = output buffer for STIO2 enabled

R_AUX2_TX_SL (w) 0x23
(formerly **AUX2_SSL**)

Slot assigner for HFC-channel[AUX2,TX] and PCM time slot configuration

Bits	Reset value	Name	Description
4..0	0	**V_AUX2_TX_SL**	**PCM time slot selection for HFC-channel[AUX2,TX]** The selected time slot in the range V_AUX2_TX_SL = 0..31 is added to the time slot block which is selected with bitmap V_SL_BL in register R_PCM_MD2.
5	0	(reserved)	Must be '0'.
7..6	0	**V_AUX2_TX_ROUT**	**PCM output buffer configuration** '00' = data transmission from HFC-channel[AUX2,TX] disabled, output buffer disabled '01' = not used '10' = output buffer for STIO1 enabled '11' = output buffer for STIO2 enabled

R_B1_RX_SL (w) 0x24
(formerly **B1_RSL**)

Slot assigner for HFC-channel[B1,RX] and PCM time slot configuration

Bits	Reset value	Name	Description
4..0	0	**V_B1_RX_SL**	**PCM time slot selection for HFC-channel[B1,RX]** The selected time slot in the range V_B1_RX_SL = 0..31 is added to the time slot block which is selected with bitmap V_SL_BL in register R_PCM_MD2.
5	0	(reserved)	Must be '0'.
7..6	0	**V_B1_RX_ROUT**	**PCM input path configuration** '00' = data transmission to HFC-channel[B1,RX] disabled, input data is ignored '01' = not used '10' = receive data from STIO2 '11' = receive data from STIO1

R_B2_RX_SL (w) 0x25
(formerly **B2_RSL**)

Slot assigner for HFC-channel[B2,RX] and PCM time slot configuration

Bits	Reset value	Name	Description
4..0	0	**V_B2_RX_SL**	**PCM time slot selection for HFC-channel[B2,RX]** The selected time slot in the range V_B2_RX_SL = 0..31 is added to the time slot block which is selected with bitmap V_SL_BL in register R_PCM_MD2.
5	0	(reserved)	Must be '0'.
7..6	0	**V_B2_RX_ROUT**	**PCM input path configuration** '00' = data transmission to HFC-channel[B2,RX] disabled, input data is ignored '01' = not used '10' = receive data from STIO2 '11' = receive data from STIO1

R_AUX1_RX_SL (w) 0x26
(formerly **AUX1_RSL**)

Slot assigner for HFC-channel[AUX1,RX] and PCM time slot configuration

Bits	Reset value	Name	Description
4..0	0	**V_AUX1_RX_SL**	**PCM time slot selection for HFC-channel[AUX1,RX]** The selected time slot in the range V_AUX1_RX_SL = 0..31 is added to the time slot block which is selected with bitmap V_SL_BL in register R_PCM_MD2.
5	0	(reserved)	Must be '0'.
7..6	0	**V_AUX1_RX_ROUT**	**PCM input path configuration** '00' = data transmission to HFC-channel[AUX1,RX] disabled, input data is ignored '01' = not used '10' = receive data from STIO2 '11' = receive data from STIO1

R_AUX2_RX_SL (w) 0x27
(formerly **AUX2_RSL**)

Slot assigner for HFC-channel[AUX2,RX] and PCM time slot configuration

Bits	Reset value	Name	Description
4..0	0	**V_AUX2_RX_SL**	**PCM time slot selection for HFC-channel[AUX2,RX]** The selected time slot in the range V_AUX2_RX_SL = 0..31 is added to the time slot block which is selected with bitmap V_SL_BL in register R_PCM_MD2.
5	0	(reserved)	Must be '0'.
7..6	0	**V_AUX2_RX_ROUT**	**PCM input path configuration** '00' = data transmission to HFC-channel[AUX2,RX] disabled, input data is ignored '01' = not used '10' = receive data from STIO2 '11' = receive data from STIO1

R_CI_TX (w) 0x28
(formerly **C/I**)

C/I channel of the GCI interface

Bits	Reset value	Name	Description
3..0	0	**V_GCI_C**	**Command bits of the C/I channel** These bits are continously send to the external GCI device.
7..4	0	(reserved)	Must be '0000'.

(See Figure 6.1 on page 136 for detailed information).

R_MON1_TX (w) 0x2A
(formerly **MON1_D**)

First monitor byte of the GCI frame

This byte must be written first. Transmission starts after writing register R_MON2_TX.

Bits	Reset value	Name	Description
7..0	0	**V_MON1_TX**	**First monitor data byte to be transmitted**

R_MON2_TX (w) 0x2B
(formerly **MON2_D**)

Second monitor byte of the GCI frame

Transmission starts after writing this register. R_MON1_TX must be written before.

Bits	Reset value	Name	Description
7..0	0	**V_MON2_TX**	**Second monitor data byte to be transmitted**

R_B1_TX (w) 0x2C
(formerly **B1_D**)

Transmit register for the PCM data from HFC-channel[B1,TX]

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.

Bits	Reset value	Name	Description
7..0		**V_B1_TX**	**PCM data byte** This data byte is written to the PCM time slot which is connected to HFC-channel[B1,TX].

R_B2_TX (w) 0x2D
(formerly **B2_D**)

Transmit register for the PCM data from HFC-channel[B2,TX]

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.

Bits	Reset value	Name	Description
7..0		**V_B2_TX**	**PCM data byte** This data byte is written to the PCM time slot which is connected to HFC-channel[B2,TX].

R_AUX1_TX (w) 0x2E
(formerly **AUX1_D**)

Transmit register for the PCM data from HFC-channel[AUX1,TX]

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.

Bits	Reset value	Name	Description
7..0		**V_AUX1_TX**	**PCM data byte** This data byte is written to the PCM time slot which is connected to HFC-channel[AUX1,TX].

R_AUX2_TX (w) 0x2F
(formerly **AUX2_D**)

Transmit register for the PCM data from HFC-channel[AUX2,TX]

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.

Bits	Reset value	Name	Description
7..0		**V_AUX2_TX**	**PCM data byte** This data byte is written to the PCM time slot which is connected to HFC-channel[AUX2,TX].

6.6.2 Read only registers

R_F0_CNTL **(r)** 0x18
(formerly **F0_CNT_L**)

F0IO pulse counter, low byte

This register is the low byte of a 16 bit ring counter. The high byte can be read from the register R_F0_CNTH. This register is incremented every 125 μ s.

Bits	Reset value	Name	Description
7..0	0	**V_F0_CNTL**	**Bits [7..0] of the F0IO counter**

R_F0_CNTH **(r)** 0x19
(formerly **F0_CNT_H**)

F0IO pulse counter, high byte

This register is the high byte of a 16 bit ring counter. The low byte can be read from the register R_F0_CNTL. This register is incremented every 32 ms.

Bits	Reset value	Name	Description
7..0	0	**V_F0_CNTH**	**Bits [15..8] of the F0IO counter**

R_CI_RX **(r)** 0x28
(formerly **C/I**)

C/I channel of the GCI interface

Bits	Reset value	Name	Description
3..0	0	**V_GCI_I**	**Indication bits of the C/I-channel** These indication bits are received from the external GCI device.
7..4	0	(reserved)	

R_PCM_GCI_STA (r) 0x29
(formerly **TRxR**)

Status register for STIO pins and GCI interface

Bits	Reset value	Name	Description
0	0	**V_MON_RXR**	**Monitor receiver ready** '1' = two monitor bytes have been received and can be read from registers R_MON1_RX and R_MON2_RX Reading R_MON2_RX resets this bit.
1	0	**V_MON_TXR**	**Monitor transmitter ready** '1' = two monitor bytes have been send and the next bytes can be written into registers R_MON1_TX and R_MON2_TX Writing on R_MON2_TX starts transmisssion and resets this bit.
5..2	0	(reserved)	
6	0	**V_STIO2_IN**	**STIO2 input** Value of the signal at pin STIO2
7	0	**V_STIO1_IN**	**STIO1 input** Value of the signal at pin STIO1

R_MON1_RX (r) 0x2A
(formerly **MON1_D**)

First monitor byte of the GCI frame

Bits	Reset value	Name	Description
7..0	0	**V_MON1_RX**	**First received monitor data byte**

R_MON2_RX (r) 0x2B
(formerly **MON2_D**)

Second monitor byte of the GCI frame

Bits	Reset value	Name	Description
7..0	0	**V_MON2_RX**	**Second received monitor data byte**

R_B1_RX (r) 0x2C
(formerly **B1_D**)

Receive register of the PCM data connected to HFC-channel[B1,RX]

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

Bits	Reset value	Name	Description
7..0		**V_B1_RX**	**PCM data byte** This data byte is received from the PCM time slot which is connected to HFC-channel[B1,RX].

R_B2_RX (r) 0x2D
(formerly **B2_D**)

Receive register of the PCM data connected to HFC-channel[B2,RX]

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

Bits	Reset value	Name	Description
7..0		**V_B2_RX**	**PCM data byte** This data byte is received from the PCM time slot which is connected to HFC-channel[B2,RX].

R_AUX1_RX (r) 0x2E
(formerly **AUX1_D**)

Receive register of the PCM data connected to HFC-channel[AUX1,RX]

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

Bits	Reset value	Name	Description
7..0		**V_AUX1_RX**	**PCM data byte** This data byte is received from the PCM time slot which is connected to HFC-channel[AUX1,RX].

R_AUX2_RX (r) 0x2F
(formerly **AUX2_D**)

Receive register of the PCM data connected to HFC-channel[AUX2,RX]

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

Bits	Reset value	Name	Description
7..0		**V_AUX2_RX**	**PCM data byte** This data byte is received from the PCM time slot which is connected to HFC-channel[AUX2,RX].

Chapter 7

Clock, reset, interrupt, timer and watchdog

Table 7.1: *Overview of the clock, reset and interrupt pins*

Number	Name	Description
26	CLKI	Oscillator input signal
27	CLKO	Oscillator output signal
44	/INT	Interrupt request
48	/RES	Reset

Table 7.2: *Overview of the HFC-S mini reset, timer and watchdog registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x1A	R_FIFO_IRQMSK	185	0x10	R_FIFO_IRQ	188
0x1B	R_MISC_IRQMSK	186	0x11	R_MISC_IRQ	189
0x1C	R_TI	187	0x1C	R_STATUS	190

7.1 Clock

7.1.1 Clock distribution

The HFC-S mini uses several internal clock frequencies f_{SYS}, f_{ST} and f_{PCM}. They are generated from one oscillator clock as shown in Figure 7.1.

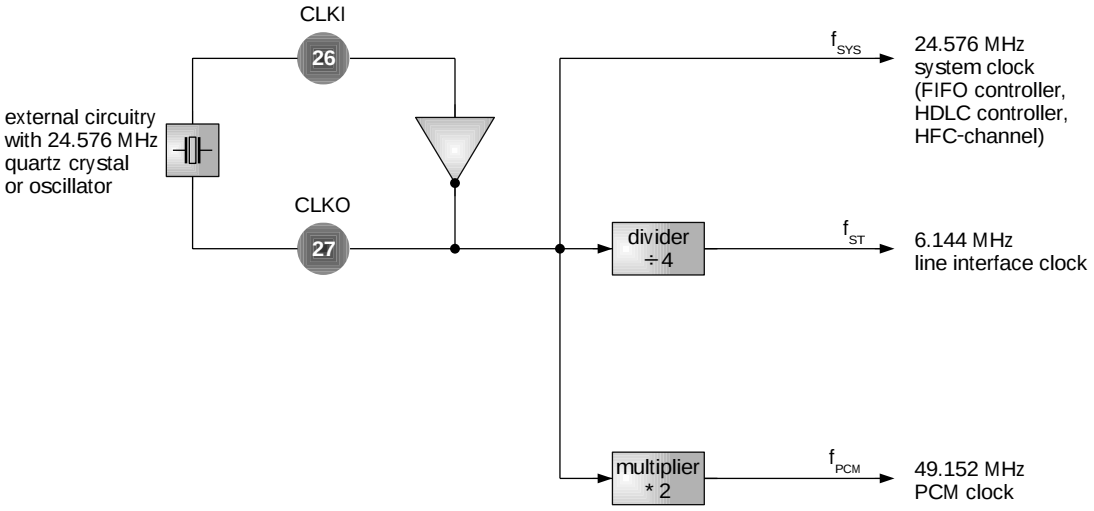


Figure 7.1: *Clock distribution*

7.1.2 Clock oscillator circuitry

There are different ways to provide the internal clocks of the HFC-S mini. This section describes the Pierce oscillator circuitry, crystal oscillator circuitry and, finally, shows how to connect the clock oscillator circuitry of several HFC-S mini in a cascade.

7.1.2.1 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.

7.1.2.2 Pierce oscillator

A typical clock oscillator circuitry using a 24.576 MHz crystal is shown in Figure 7.2. This Pierce oscillator is very popular for clock generation and is widely known from literature.

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

$$C_L = \frac{C1 \cdot C2}{C1 + C2} + C_S$$

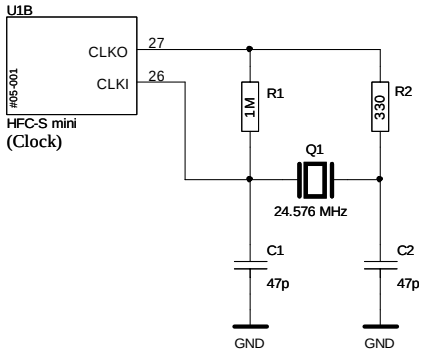


Figure 7.2: *Standard HFC-S mini quartz circuitry*

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.

$$\mathsf{R2} \sim \frac{1}{2\pi f_{\mathsf{Q1}} \cdot \mathsf{C2}}$$

R2 and C2 provide a low-pass filter that prevents the circuit from oscillating at a higher harmonic of the crystal frequency.

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 the HFC-S mini has a bufffered inverter between pins CLKI and CLKO the value of R2 can be increased. A factor of about 2 . . 3, e.g., is well.

The circuitry shown in Figure 7.2 is based on a crystal with $C_L = 30\,\mathrm{pF}$ and a stray capacitance of $C_S = 5\,\mathrm{pF}$. This leads to

$$\mathsf{C1} = \mathsf{C2} = 2 \cdot (C_L - C_S) = 59\,\mathrm{pF} \sim 47\,\mathrm{pF}$$

and

$$\mathsf{R2} = \frac{3}{2\pi f_{\mathsf{Q1}} \cdot \mathsf{C2}} = 413\,\Omega \sim 330\,\Omega \ .$$

7.1.2.3 Crystal oscillator circuitry

It is possible to feed the CLKI input of HFC-S mini with a standard 3.3 V crystal oscillator. The input switching level is close to $V_{DD}/2$ (CMOS level) and the HFC-S mini can accept at least a duty cycle of 45 % high/55 % low to 55 % high/45 % low.

7.1.2.4 HFC-S mini cascade

Figure 7.3 shows how to connect several HFC-S mini to only one quartz circuitry.

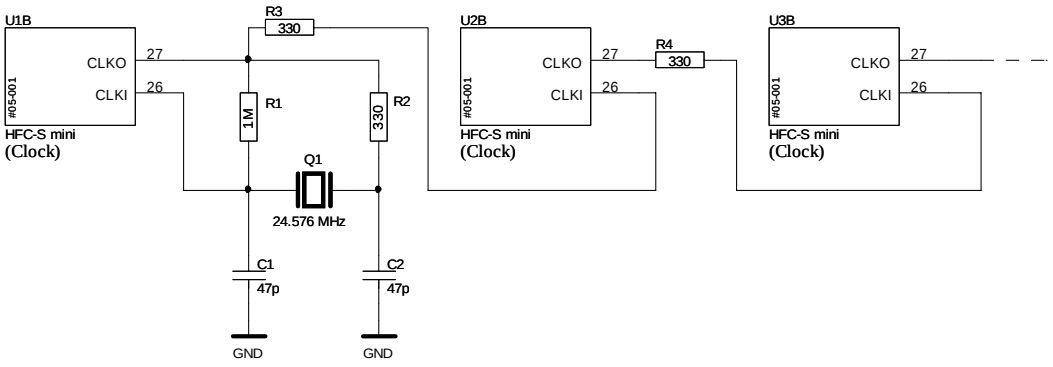


Figure 7.3: *Cascade-connected HFC-S mini with only one quartz circuitry*

7.2 Reset

HFC-S mini has a level sensitive reset input at pin 48 (active low). The reset pulse must not be shorter than $\frac{6}{f_{CLKI}}$.

After reset, HFC-S mini enters an initialization sequence. Its duration is 100 μs. When the initialization process is finished, the bit V_BUSY in register R_STATUS changes from '1' to '0'.

The HFC-S mini has a soft reset which resets all registers except R_CIRM and R_ST_CLK_DLY. Soft reset is initiated with V_SRES = '1' in register R_CIRM. The bit must be reset to '0' from the host processor.

7.3 Interrupt

7.3.1 Overview

The HFC-S mini is equipped with a maskable interrupt engine. Every FIFO has its own interrupt capability. Five additional interrupts are available to indicate other interrupt events.

FIFO interrupts can be read from the register R_FIFO_IRQ while the other interrupts are stored in R_MISC_IRQ. Reading an interrupt status register resets the bits. New interrupts may occur during the read access. These interrupts are reported at the next read of R_FIFO_IRQ or R_MISC_IRQ.

All interrupt bits are reported regardless of the mask registers settings (R_FIFO_IRQMSK and R_MISC_IRQMSK). The mask register settings only influence the interrupt output condition.

The interrupt output goes inactive during the read of R_FIFO_IRQ or R_MISC_IRQ. If interrupts occur during this read the interrupt line goes active immediately after the read is finished. So processors with level or transition triggered interrupt inputs can be connected.

Pin 44 is the interrupt output line. After reset, all interrupts are disabled. The interrupt line must be enabled with V_IRQ_EN set to '1' in register R_MISC_IRQMSK. The polarity of the interrupt signals can be changed by an optional external transistor from *active low* to *active high*. This must be configured with the bitmap V_IRQ_REV in the same register.

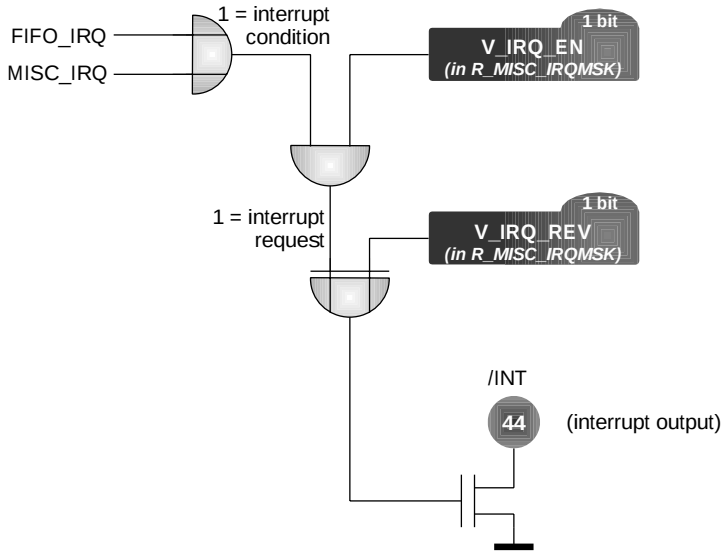


Figure 7.4: *Interrupt output*

7.3.2 FIFO interrupt

The interrupt capability of every FIFO can be enabled or disabled by setting its bit V_FIFO0_TX_IRQMSK . . V_FIFO3_RX_IRQMSK in register R_FIFO_IRQMSK.

Interrupt events can be read from register R_FIFO_IRQ even if the mask bits are not set. Reading R_FIFO_IRQ clears the register. All FIFO interrupts are OR-ed to bit V_FIFO_IRQSTA in register R_STATUS. This bit reports only those FIFO interrupts with enabled mask bit.

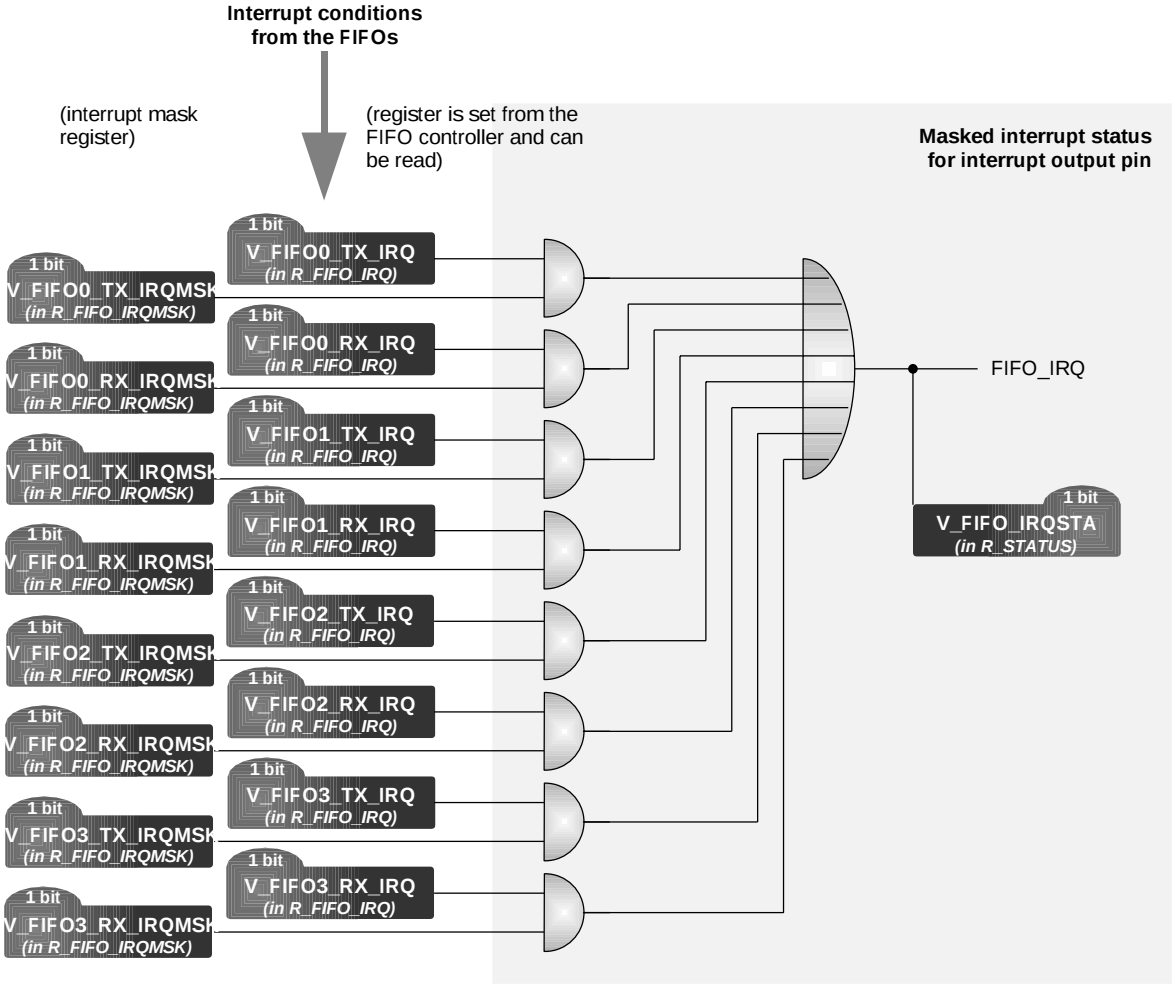


Figure 7.5: *FIFO interrupt*

7.3.3 Miscelleanous interrupts

Miscelleanous interrupt events can be read from register R_MISC_IRQ even if the mask bits are not set. Reading R_MISC_IRQ clears the register. All miscelleanous interrupts are OR-ed to bit V_MISC_IRQSTA in register R_STATUS. This bit reports only those miscelleanous interrupts with enabled mask bit.

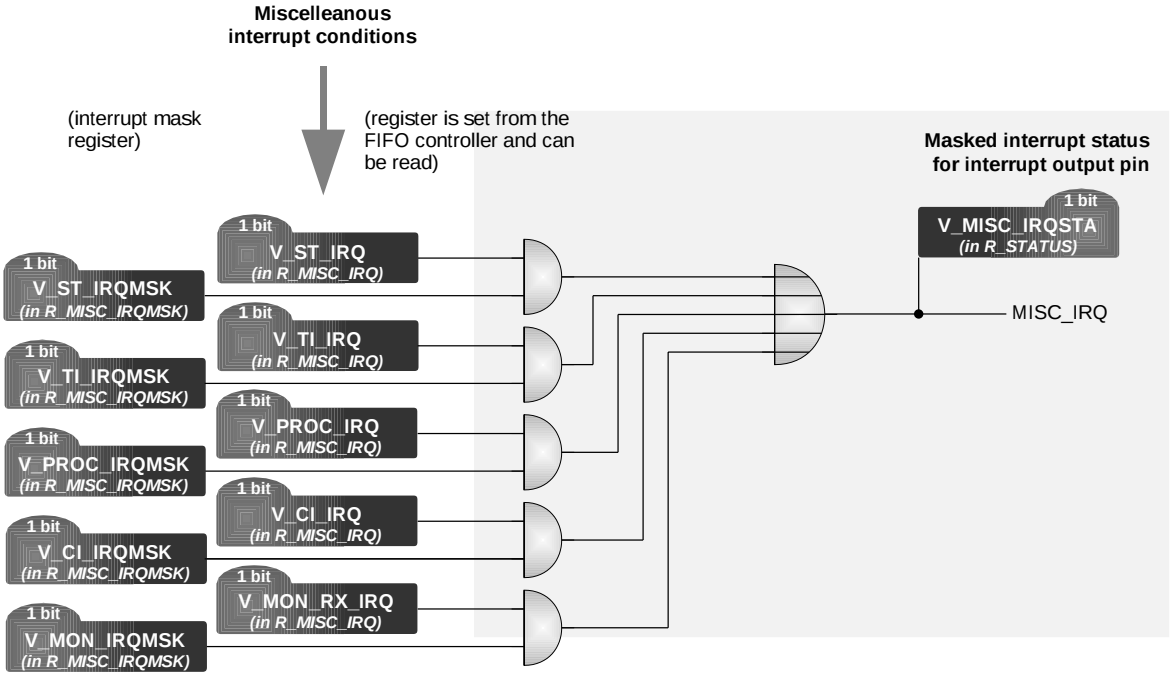


Figure 7.6: *Miscelleanous interrupts*

7.3.3.1 Line interface interrupt

The line interface has interrupt capability to indicate a state change condition. The interrupt mask can to be programmed with the bit V_ST_IRQMSK in register R_MISC_IRQMSK.

The register R_MISC_IRQ contains the state change condition even if the interrupt mask is not set.

7.3.3.2 Timer interrupt

HFC-S mini 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. This leads to a timer event frequency from 250 μs to 8.192 s.

7.3.3.3 125 µs interrupt

HFC-S mini changes every 125 µs from non processing into processing state. This event can be reported with an interrupt. The bit V_PROC_IRQMSK in register R_MISC_IRQMSK must be set to '1' to enable this interrupt capability. In case of an interrupt, the bit V_PROC_IRQ in register R_MISC_IRQ has the value '1'.

7.3.3.4 GCI interrupt

Received indication bits of the GCI interface are stored in the bitmap V_GCI_I of register R_CI_RX. Whenever these bits change, an interrupt can be generated. For this, the mask bit V_CI_IRQMSK in register R_MISC_IRQMSK must be set to '1'.

The interrupt event is reported in V_CI_IRQ in register R_MISC_IRQMSK even if the mask bit is not set.

7.3.3.5 Monitor bytes received interrupt

The bitmap V_MON_RX_IRQ of register R_MISC_IRQ indicates when two monitor bytes are received. For this, the mask bit V_MON_IRQMSK in register R_MISC_IRQMSK must be set to '1'.

The interrupt event is reported in V_MON_RX_IRQ even if the mask bit is not set.

7.4 Register description

> **Please note !**
>
> Due to a thorough revision of the HFC-S mini data sheet, some registers had to be renamed. Please see remarks on page 15.

7.4.1 Write only registers

R_FIFO_IRQMSK (formerly **INT_M1**)	**(w)**	0x1A

FIFO interrupt masks

Interrupt capability is enabled for every FIFO separately. For mask bits of this register a '1' enables and a '0' disables the interrupt.

Bits	Reset value	Name	Description
0	0	**V_FIFO0_TX_IRQMSK**	**Interrupt mask for FIFO[0,TX]**
1	0	**V_FIFO0_RX_IRQMSK**	**Interrupt mask for FIFO[0,RX]**
2	0	**V_FIFO1_TX_IRQMSK**	**Interrupt mask for FIFO[1,TX]**
3	0	**V_FIFO1_RX_IRQMSK**	**Interrupt mask for FIFO[1,RX]**
4	0	**V_FIFO2_TX_IRQMSK**	**Interrupt mask for FIFO[2,TX]**
5	0	**V_FIFO2_RX_IRQMSK**	**Interrupt mask for FIFO[2,RX]**
6	0	**V_FIFO3_TX_IRQMSK**	**Interrupt mask for FIFO[3,TX]**
7	0	**V_FIFO3_RX_IRQMSK**	**Interrupt mask for FIFO[3,RX]**

R_MISC_IRQMSK (formerly INT_M2)	(w)	0x1B

Interrupt mask register

For mask bits of this register the value '1' enables and '0' disables the interrupt.

Bits	Reset value	Name	Description
0	0	**V_ST_IRQMSK**	**State change interrupt mask bit of the line interface**
1	0	**V_TI_IRQMSK**	**Timer elapsed interrupt mask bit**
2	0	**V_PROC_IRQMSK**	**Processing / non processing transition interrupt mask bit** (every 125 μs)
3	0	**V_CI_IRQMSK**	**Command / indication interrupt mask bit**
4	0	**V_MON_IRQMSK**	**Receive monitor byte interrupt mask bit**
5	0	(reserved)	Must be '0'.
6	0	**V_IRQ_REV**	**Reverse interrupt output** '0' = interrupt signal is indicated with low level on pin /INT '1' = interrupt signal is indicated with open drain on pin /INT
7	0	**V_IRQ_EN**	**Enable interrupt output** '0' = interrupt pin /INT disabled (always open drain) '1' = interrupt pin /INT enabled

R_TI (w) 0x1C
(formerly **TIME_SEL**)

Periodic timer interrupt configuration register

Bits	Reset value	Name	Description
3..0	0	**V_EV_TS**	**Timer interrupt frequency** A timer interrupt can be enabled. The interrupt event occurs after every $2^n \cdot 250\,\mu s$ 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	(reserved)	Must be '0000'.

7.4.2 Read only registers

R_FIFO_IRQ (formerly **INT_S1**)	**(r)**	0x10

FIFO interrupt status register

A FIFO interrupt condition occurs when a complete HDLC frame has been send or received in HDLC mode (increment of *F*-counter). In transparent mode, the interrupt event can be enabled when the specified number of bytes are transmitted or received (see register A_CON_HDLC).

'0' = no interrupt condition has occured
'1' = an interrupt condition has occured

Every FIFO interrupt can be enabled and disabled with its mask bit in register R_FIFO_IRQMSK. R_FIFO_IRQ reports interrupt conditions even when the mask bits are not set.

Bits	Reset value	Name	Description
0	0	**V_FIFO0_TX_IRQ**	**Interrupt event of FIFO[0,TX]**
1	0	**V_FIFO0_RX_IRQ**	**Interrupt event of FIFO[0,RX]**
2	0	**V_FIFO1_TX_IRQ**	**Interrupt event of FIFO[1,TX]**
3	0	**V_FIFO1_RX_IRQ**	**Interrupt event of FIFO[1,RX]**
4	0	**V_FIFO2_TX_IRQ**	**Interrupt event of FIFO[2,TX]**
5	0	**V_FIFO2_RX_IRQ**	**Interrupt event of FIFO[2,RX]**
6	0	**V_FIFO3_TX_IRQ**	**Interrupt event of FIFO[3,TX]**
7	0	**V_FIFO3_RX_IRQ**	**Interrupt event of FIFO[3,RX]**

R_MISC_IRQ (r) 0x11
(formerly **INT_S2**)

Miscellaneous interrupt status register

Bits	Reset value	Name	Description
0	0	**V_ST_IRQ**	**TE/NT state machine interrupt status** '1' = state changed of the S/T state machine
1	0	**V_TI_IRQ**	**Timer interrupt status** '1' = timer elapsed
2	0	**V_PROC_IRQ**	**Processing / non processing transition interrupt status** '1' = the HFC-S mini has changed from processing to nonprocessing phase (every 125 μ s).
3	0	**V_CI_IRQ**	**Command / indication interrupt status** '1' = received indication bits have changed
4	0	**V_MON_RX_IRQ**	**Receive monitor bytes interrupt status** '1' = two monitor bytes have been received
7..5	0	(reserved)	

R_STATUS	(r)	0x1C

HFC-S mini status register

Bits	Reset value	Name	Description
0		V_BUSY	**Busy / not busy status** '0' = the HFC-S mini is not busy, all accesses are allowed '1' = the HFC-S mini is BUSY during initializing, reset FIFO, increment *F*-counter or change FIFO **Note:** Accesses to FIFO registers are not allowed during busy period.
1	0	V_PROC	**Processing / non processing status** '0' = HFC-S mini has finished the processing phase during the 125 μs cycls '1' = HFC-S mini is in processing phase (oncy every 125 μs)
2	0	(reserved)	
3		V_AWAKE_IN	**Wake-up input** Value of the AWAKE input signal
4		V_SYNC_IN	**Synchronization input** Value of the SYNC_I input signal
5	0	(reserved)	
6	0	V_MISC_IRQSTA	**Any miscellaneous interrupt** All enabled miscellaneous interrupts in register R_MISC_IRQ are 'ored'.
7	0	V_FIFO_IRQSTA	**Any FIFO interrupt** All enabled FIFO interrupts in register R_FIFO_IRQ are 'ored'.

Reading the R_STATUS register clears no bit.

Chapter 8

Electrical characteristics

Absolute maximum ratings

Parameter	Symbol	Min.	Max.
Power supply	V_{DD}	-0.3 V	+7.0 V
Input voltage	V_I	-0.3 V	$V_{DD}+0.3$ V
Output voltage	V_O	-0.3 V	$V_{DD}+0.3$ V
Operating temperature	T_{opr}	-10 °C	+85 °C
Storage temperature	T_{stg}	-40 °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	$V_{DD} = 3.3$ V nominal
		4.75 V	5 V	5.25 V	$V_{DD} = 5$ V nominal
Operating temperature	T_{opr}	0 °C		+70 °C	
Supply current					$f_{CLK} = 24.576$ MHz
Normal	I_{DD}		12 mA		$V_{DD} = 3.3$ V, running oscillator
			24 mA		$V_{DD} = 5$ V, running oscillator
Power down				1 mA	$V_{DD} = 3.3$ V, oscillator stopped, no pin floating
				2 mA	$V_{DD} = 5$ V, oscillator stopped, no pin floating

Electrical characteristics for 3.3 V power supply (TTL level)

$V_{DD} = 3.0\,\text{V}\ to\ 3.6\,\text{V}, T_{opr} = 0\,^{\circ}\text{C}\ to\ +70\,^{\circ}\text{C}$

Parameter	Symbol	Min.	Typ.	Max
Low input voltage	V_{IL}			0.5 V
High input voltage	V_{IH}	1.5 V		
Low output voltage	V_{OL}	0 V		0.4 V
High output voltage	V_{OH}	2.4 V		V_{DD}
Schmitt trigger, positive-going threshold	VT+			1.3 V
Schmitt trigger, negative-going threshold	VT-	0.5 V		

Electrical characteristics for 3.3 V power supply (CMOS level)

$V_{DD} = 3.0\,\text{V}\ to\ 3.6\,\text{V}, T_{opr} = 0\,^{\circ}\text{C}\ to\ +70\,^{\circ}\text{C}$

Parameter	Symbol	Min.	Typ.	Max
Low input voltage	V_{IL}			1.0 V
High input voltage	V_{IH}	2.0 V		
Low output voltage	V_{OL}	0 V		0.4 V
High output voltage	V_{OH}	2.4 V		V_{DD}
Schmitt trigger, positive-going threshold	VT+			2.0 V
Schmitt trigger, negative-going threshold	VT-	1.0 V		

Electrical characteristics for 5 V power supply (TTL level)

$V_{DD} = 4.75\,\text{V}$ *to* $5.25\,\text{V}, T_{opr} = 0\,^{\circ}\text{C}$ *to* $+70\,^{\circ}\text{C}$

Parameter	Symbol	Min.	Typ.	Max
Low input voltage	V_{IL}			0.8 V
High input voltage	V_{IH}	2.0 V		
Low output voltage	V_{OL}	0 V		0.4 V
High output voltage	V_{OH}	2.4 V		V_{DD}
Schmitt trigger, positive-going threshold	VT+			2.0 V
Schmitt trigger, negative-going threshold	VT-	0.8 V		

Electrical characteristics for 5 V power supply (CMOS level)

$V_{DD} = 4.75\,\text{V}$ *to* $5.25\,\text{V}, T_{opr} = 0\,^{\circ}\text{C}$ *to* $+70\,^{\circ}\text{C}$

Parameter	Symbol	Min.	Typ.	Max
Low input voltage	V_{IL}			1.5 V
High input voltage	V_{IH}	3.5 V		
Low output voltage	V_{OL}	0 V		0.4 V
High output voltage	V_{OH}	2.4 V		V_{DD}
Schmitt trigger, positive-going threshold	VT+			4.0 V
Schmitt trigger, negative-going threshold	VT-	1.0 V		

Table 8.1: *I/O characteristics*

Input	Interface level
/DS	CMOS
R/W	CMOS
/CS	CMOS, internal pull-up resistor
ALE	CMOS, internal pull-up resistor
A0	CMOS
D7 . . D0	CMOS
CLKI	CMOS
AWAKE	CMOS
C4IO	TTL Schmitt Trigger, internal pull-up resistor
F0IO	CMOS, internal pull-up resistor
STIO1 , STIO2	CMOS, internal pull-up resistor
/RES	CMOS Schmitt Trigger, internal pull-up resistor

Table 8.2: *Driver Capability*

Output	Low 0.4 V	High $V_{DD} - 0.8$ V
D7 . . D0	4 mA	2 mA
C4IO	8 mA	4 mA
F0IO	8 mA	4 mA
STIO1 , STIO2	8 mA	4 mA
F1_A , F1_B	4 mA	2 mA
/INT	4 mA	

Chapter 9

HFC-S mini
package dimensions

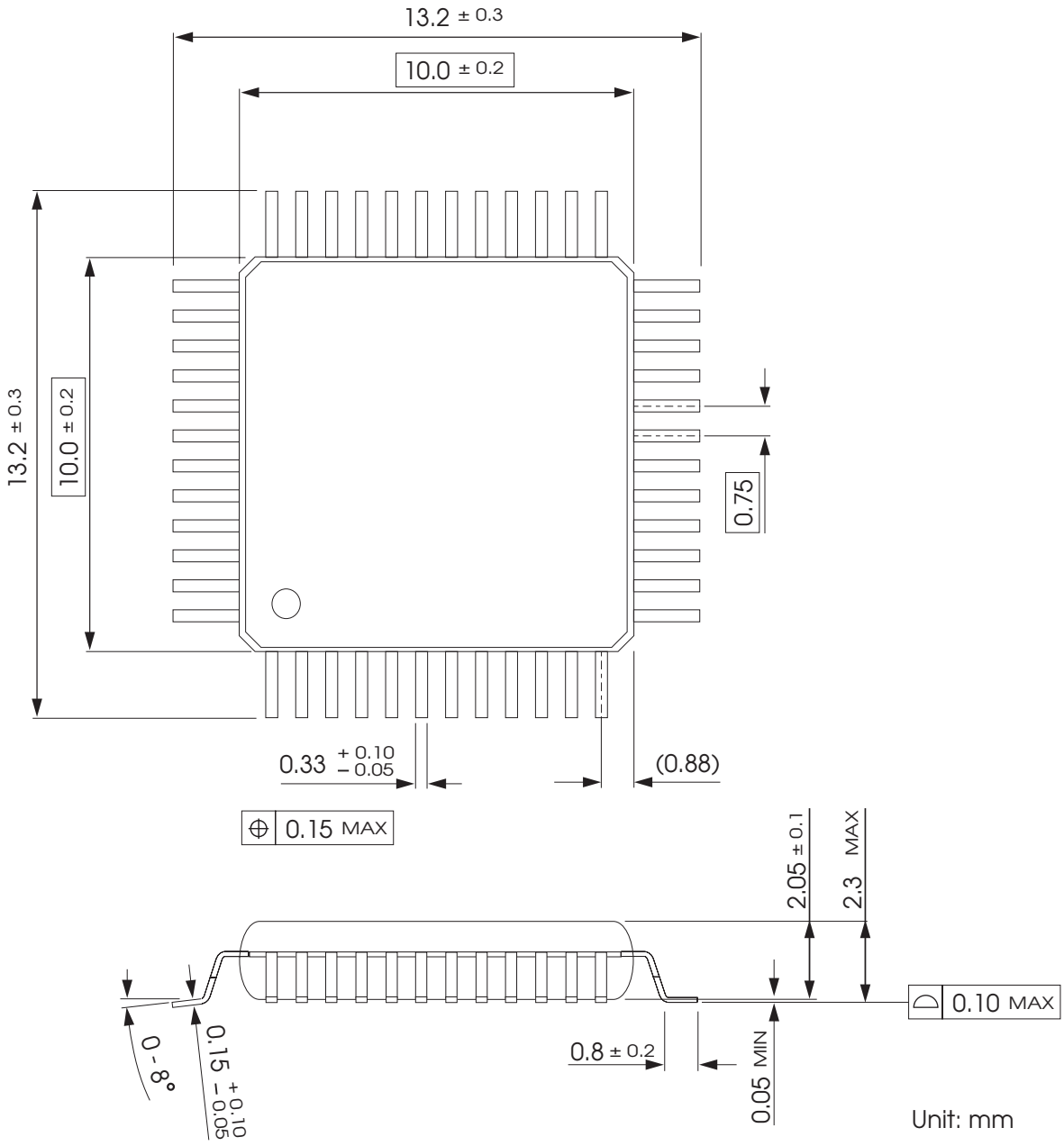


Figure 9.1: *HFC-S mini package dimensions*

References

[1] 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*.

[2] Siemens AG. *ICs for Communication. IOM®-2 Interface Reference Guide*, 3 1991.

[3] 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.

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.

96KHZ	96 kHz
ACT	active, activation
ADDR	address
ADJ	adjust
AUX1	AUX1-channel
AUX2	AUX2-channel
AWAKE	awake
B1	B1-channel
B12	B1- and B2-channels
B2	B2-channel
BIT	bit
BL	block
BUSY	busy
C	command bits (of the C/I channel)
C4	C4IO clock (PCM double bit clock)
CH	HFC-channel
CHANNEL	HFC-channel
CHIP	microchip
CI	Command / Indication (C/I channel of the GCI interface)
CIRM	configuration, interrupt and reset
CLK	clock
CNT	counter
CNTH	counter, high byte
CNTL	counter, low byte
CODECA	CODEC A
CODECB	CODEC B
CON	connection settings
CSM	Channel Select Mode
CTRL	control
D	D-channel
DATA	data
DF	data flow
DIR	direction
DLY	delay
DR	data rate
E	E-channel
EN	enable
EV	event
EXP	expired
F	*F*-counter
F0	frame synchronization signal
F1	*F*1-counter
F2	*F*2-counter
FIFO	FIFO
FILL	fill level
FLOW	flow
FO	force
FR	frame
G2	G2 state
G3	G3 state
GCI	General Circuit Interface
HDLC	high-level data link control
I	indication bits (of the C/I channel)
ICR	increase
ID	identifier
IFF	inter frame fill
IGNO	ignore
IN	input
INC	increment
INFO0	INFO 0 line condition (no signal)
INV	invert, inversion
IRQ	interrupt
IRQMSK	interrupt mask
IRQSTA	interrupt status
LD	load
LEN	length
LI	line
LO	low
LOOP	loop
MAN	manual
MD	mode
MF	multiframe
MIR	mirror, mirroring
MISC	miscellaneous
MON	monitor channel of the GCI interface
MON1	first monitor byte

MON2	second monitor byte
MSK	mask
NEG	negative
NOINC	no increment
NUM	number
OKI	OKI™
OUT	output
PAR	parameter
PCM	pulse code modulation
PLL	phase locked loop
POL	polarity
PRIO	priority
PROC	processing
RAM	RAM
RD	read
RDY	ready
RES	reset
REV	reverse
ROUT	routing
RX	receive
RXR	receiver ready
SET	setup
SL	time slot
SMPL	sample
SQ	S/Q-bits
SRC	source
SRES	soft reset
ST	S/T interface
STA	state, status
START	start
STATUS	status
STIO1	PCM data bus 1
STIO2	PCM data bus 2
STOP	stop
SWAP	swap
SYNC	synchronize, synchronization
T2	S/T timer T2
THRES	threshold
TI	timer
TRP	transparent
TS	time step
TX	transmit
TXR	transmitter ready
USAGE	usage
USB	Universal Serial Bus
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 HFC-S mini