Cologne Chip

HFC - E1

ISDN HDLC FIFO controller

with

Primary Rate Interface (E1)

Data Sheet ✦ March 2003 (rev. A)

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

General Remarks to Notations

1. Numerical values have different notations for various number systems, e.g. the hexadecimal value 0xC9 is in binary '11001001' and in decimal notation 201.
2. The first letter of register names indicates the type: 'R_...' is a register, 'A_...' is an array-register.
3. The first letter of register's bit and bitmap names indicates the type: 'V_...' is a bit or bitmap value and 'M_...' is its bitmap mask, i.e. all bits of the bitmap are set to '1'.

List of Figures

List of Tables

List of Registers (sorted by name)

Please note !

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

It must be distinguished between *registers*, *array registers* and *multi-registers*.

Array registers have multiple instances and are indexed by a number. This index is either the FIFO number (R_FIFO with 13 indexed registers) or the PCM time slot number (R_SLOT with 2 indexed registers). Array registers have equal name, bitmap structure and meaning for every instance.

Multi-registers have multiple instances, too, but they are selected by a bitmap value. With this value, different registers can be selected with the same address. Multi-register addresses are 0x15 (14 instances selected by R_PCM_MD0) and 0x0F (2 instances selected by R_FIFO_MD) for HFC-E1. Multi-registers have different names, bitmap structure and meaning for each instance.

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

Write only registers:

Address	Name	Reset group	Page
0xF4	A_CH_MSK	0, 1	124
0xFC	A_CHANNEL	0, 1	127
0xFA	A_CON_HDLC	0, 1	125
0xD1	A_CONF	–	205
0xFD	A_FIFO_SEQ	0, 1	127
0x0E	R_INC_RES_FIFO	–	138
0xFF	A_IRQ_MSK	0, 1	239
0xD0	A_SL_CFG	0, 3	123
0xFB	A_SUBCH_CFG	0, 1	126
0x1B	R_BERT_WD_MD	0, 1	215
0x45	R_BRG_CTRL	0	227
0x47	R_BRG_MD	0	228
0x02	R_BRG_PCM_CFG	H	226
0x4C	R_BRG_TIM_SEL01	0	231
0x4D	R_BRG_TIM_SEL23	0	231
0x4E	R_BRG_TIM_SEL45	0	232
0x4F	R_BRG_TIM_SEL67	0	232
0x48	R_BRG_TIM0	0	229
0x49	R_BRG_TIM1	0	229
0x4A	R_BRG_TIM2	0	230
0x4B	R_BRG_TIM3	0	230
0x00	R_CIRM	H	86
0x18	R_CONF_EN	0, 2	204
0x01	R_CTRL	H	87

Address	Name	Reset group	Page
0x1C	R_DTMF0	0	211
0x1D	R_DTMF1	0	212
0x20	R_E1_WR_STA	0, 1, 3	155
0x0D	R_FIFO_MD	H	120
0x0F	R_FIFO	0, 1	121
0x0B	R_FIRST_FIFO	0, 1	119
0x0F	R_FSM_IDX	0, 1	121
0x42	R_GPIO_EN0	0	256
0x43	R_GPIO_EN1	0	257
0x40	R_GPIO_OUT0	0	254
0x41	R_GPIO_OUT1	0	255
0x44	R_GPIO_SEL	0	258
0x13	R_IRQ_CTRL	0	237
0x11	R_IRQMSK_MISC	H	236
0x22	R_LOS0	0, 1, 3	155
0x23	R_LOS1	0, 1, 3	156
0x14	R_PCM_MD0	0, 2	183
0x15	R_PCM_MD1	0, 2	189
0x15	R_PCM_MD2	0, 2	190
0x46	R_PWM_MD	0	198
0x38	R_PWM0	0, 1, 3	197
0x39	R_PWM1	0, 1, 3	197
0x08	R_RAM_ADDR0	0	87
0x09	R_RAM_ADDR1	0	88
0x0A	R_RAM_ADDR2	0	88
0x0C	R_RAM_MISC	H	89

Address	Name	Reset group	Page
0x25	R_RX_FR0	0, 1, 3	158
0x26	R_RX_FR1	0, 1, 3	159
0x30	R_RX_OFF	0, 1, 3	165
0x24	R_RX0	0, 1, 3	157
0x15	R_SH0H	0, 2	191
0x15	R_SH0L	0, 2	191
0x15	R_SH1H	0, 2	192
0x15	R_SH1L	0, 2	191
0x15	R_SL_SEL0	0, 2	184
0x15	R_SL_SEL1	0, 2	185
0x15	R_SL_SEL2	0, 2	186
0x15	R_SL_SEL3	0, 2	186
0x15	R_SL_SEL4	0, 2	187
0x15	R_SL_SEL5	0, 2	187
0x15	R_SL_SEL6	0, 2	188
0x15	R_SL_SEL7	0, 2	188
0x10	R_SLOT	0, 2	122
0x35	R_SYNC_CTRL	0, 1, 3	168
0x31	R_SYNC_OUT	0, 1, 3	166
0x1A	R_TI_WD	0, 1	238
0x2C	R_TX_FR0	0, 1, 3	162
0x2D	R_TX_FR1	0, 1, 3	163
0x2E	R_TX_FR2	0, 1, 3	164
0x34	R_TX_OFF	0, 1, 4	167
0x28	R_TX0	0, 1, 3	160
0x29	R_TX1	0, 1, 3	161

Read only registers:

Address	Name	Reset group	Page
0x0C	A_F1	0, 1	141
0x0C	A_F12	0, 1	142
0x0D	A_F2	0, 1	141
0x04	A_Z1	0, 1	139
0x04	A_Z12	0, 1	141
0x05	A_Z1H	0, 1	139
0x04	A_Z1L	0, 1	139
0x06	A_Z2	0, 1	140
0x07	A_Z2H	0, 1	140
0x06	A_Z2L	0, 1	140
0x1B	R_BERT_ECH	0, 1	217
0x1A	R_BERT_ECL	0, 1	216
0x17	R_BERT_STA	0, 1	216
0x16	R_CHIP_ID	H	91
0x1F	R_CHIP_RV	–	91
0x14	R_CONF_OFLOW	0, 1	206
0x35	R_CRC_ECH	0, 3	174
0x34	R_CRC_ECL	0, 3	174
0x37	R_E_ECH	0, 3	175
0x36	R_E_ECL	0, 3	175

Address	Name	Reset group	Page
0x20	R_STATE	0, 3	169
0x19	R_F0_CNTH	0, 1	193
0x18	R_F0_CNTL	0, 1	193
0x31	R_FAS_ECH	0, 3	173
0x30	R_FAS_ECL	0, 3	173
0x44	R_GPI_IN0	–	261
0x45	R_GPI_IN1	–	262
0x46	R_GPI_IN2	–	263
0x47	R_GPI_IN3	–	264
0x40	R_GPIO_IN0	–	259
0x41	R_GPIO_IN1	–	260
0x88	R_INT_DATA	–	142
0xC8	R_IRQ_FIFO_BL0	0, 1	243
0xC9	R_IRQ_FIFO_BL1	0, 1	244
0xCA	R_IRQ_FIFO_BL2	0, 1	245
0xCB	R_IRQ_FIFO_BL3	0, 1	246
0xCC	R_IRQ_FIFO_BL4	0, 1	247
0xCD	R_IRQ_FIFO_BL5	0, 1	248
0xCE	R_IRQ_FIFO_BL6	0, 1	249
0xCF	R_IRQ_FIFO_BL7	0, 1	250
0x11	R_IRQ_MISC	0, 1	241
0x10	R_IRQ_OVIEW	0, 1	240
0x15	R_RAM_USE	0, 1	90
0x24	R_RX_STA0	0, 3	170
0x25	R_RX_STA1	0, 3	171
0x26	R_RX_STA2	0, 3	171
0x27	R_RX_STA3	0, 3	172
0x39	R_SA6_SA13_ECH	0, 3	176
0x38	R_SA6_SA13_ECL	0, 3	175
0x3B	R_SA6_SA23_ECH	0, 3	176
0x3A	R_SA6_SA23_ECL	0, 3	176
0x2C	R_SLIP	0, 3	172
0x1C	R_STATUS	–	242
0x33	R_VIO_ECH	0, 3	174
0x32	R_VIO_ECL	0, 3	173

Read / Write registers:

Address	Name	Reset group	Page
0x84	A_FIFO_DATA0_NOINC	–	144
0x80	A_FIFO_DATA0	–	143
0x84	A_FIFO_DATA1_NOINC	–	145
0x80	A_FIFO_DATA1	–	143
0x84	A_FIFO_DATA2_NOINC	–	145
0x80	A_FIFO_DATA2	–	144
0xC0	R_RAM_DATA	–	90

Note: See table 12.4 on page 235 for ‘Reset group’ explanation.

List of Registers (sorted by address)

> **Please note !**
>
> See explanation of register types on page 14.

Write only registers:

Address	Name	Reset group	Page
0x00	R_CIRM	H	86
0x01	R_CTRL	H	87
0x02	R_BRG_PCM_CFG	H	226
0x08	R_RAM_ADDR0	0	87
0x09	R_RAM_ADDR1	0	88
0x0A	R_RAM_ADDR2	0	88
0x0B	R_FIRST_FIFO	0, 1	119
0x0C	R_RAM_MISC	H	89
0x0D	R_FIFO_MD	H	120
0x0E	R_INC_RES_FIFO	–	138
0x0F	R_FSM_IDX	0, 1	121
0x0F	R_FIFO	0, 1	121
0x10	R_SLOT	0, 2	122
0x11	R_IRQMSK_MISC	H	236
0x13	R_IRQ_CTRL	0	237
0x14	R_PCM_MD0	0, 2	183
0x15	R_PCM_MD1	0, 2	189
0x15	R_PCM_MD2	0, 2	190
0x15	R_SH0H	0, 2	191
0x15	R_SH1H	0, 2	192
0x15	R_SH0L	0, 2	191
0x15	R_SH1L	0, 2	191
0x15	R_SL_SEL0	0, 2	184
0x15	R_SL_SEL1	0, 2	185
0x15	R_SL_SEL2	0, 2	186
0x15	R_SL_SEL3	0, 2	186
0x15	R_SL_SEL4	0, 2	187
0x15	R_SL_SEL5	0, 2	187
0x15	R_SL_SEL6	0, 2	188
0x15	R_SL_SEL7	0, 2	188
0x18	R_CONF_EN	0, 2	204
0x1A	R_TI_WD	0, 1	238
0x1B	R_BERT_WD_MD	0, 1	215
0x1C	R_DTMF0	0	211
0x1D	R_DTMF1	0	212
0x20	R_E1_WR_STA	0, 1, 3	155
0x22	R_LOS0	0, 1, 3	155
0x23	R_LOS1	0, 1, 3	156
0x24	R_RX0	0, 1, 3	157
0x25	R_RX_FR0	0, 1, 3	158
0x26	R_RX_FR1	0, 1, 3	159
0x28	R_TX0	0, 1, 3	160
0x29	R_TX1	0, 1, 3	161
0x2C	R_TX_FR0	0, 1, 3	162
0x2D	R_TX_FR1	0, 1, 3	163
0x2E	R_TX_FR2	0, 1, 3	164
0x30	R_RX_OFF	0, 1, 3	165
0x31	R_SYNC_OUT	0, 1, 3	166
0x34	R_TX_OFF	0, 1, 4	167
0x35	R_SYNC_CTRL	0, 1, 3	168
0x38	R_PWM0	0, 1, 3	197
0x39	R_PWM1	0, 1, 3	197
0x40	R_GPIO_OUT0	0	254
0x41	R_GPIO_OUT1	0	255
0x42	R_GPIO_EN0	0	256
0x43	R_GPIO_EN1	0	257
0x44	R_GPIO_SEL	0	258
0x45	R_BRG_CTRL	0	227
0x46	R_PWM_MD	0	198
0x47	R_BRG_MD	0	228
0x48	R_BRG_TIM0	0	229
0x49	R_BRG_TIM1	0	229
0x4A	R_BRG_TIM2	0	230
0x4B	R_BRG_TIM3	0	230
0x4C	R_BRG_TIM_SEL01	0	231
0x4D	R_BRG_TIM_SEL23	0	231
0x4E	R_BRG_TIM_SEL45	0	232
0x4F	R_BRG_TIM_SEL67	0	232
0xD0	A_SL_CFG	0, 3	123
0xD1	A_CONF	–	205
0xF4	A_CH_MSK	0, 1	124
0xFA	A_CON_HDLC	0, 1	125
0xFB	A_SUBCH_CFG	0, 1	126
0xFC	A_CHANNEL	0, 1	127
0xFD	A_FIFO_SEQ	0, 1	127
0xFF	A_IRQ_MSK	0, 1	239

Read only registers:

Address	Name	Reset group	Page
0x04	A_Z12	0, 1	141
0x04	A_Z1L	0, 1	139
0x04	A_Z1	0, 1	139
0x05	A_Z1H	0, 1	139
0x06	A_Z2L	0, 1	140
0x06	A_Z2	0, 1	140
0x07	A_Z2H	0, 1	140
0x0C	A_F1	0, 1	141
0x0C	A_F12	0, 1	142
0x0D	A_F2	0, 1	141
0x10	R_IRQ_OVIEW	0, 1	240
0x11	R_IRQ_MISC	0, 1	241
0x14	R_CONF_OFLOW	0, 1	206
0x15	R_RAM_USE	0, 1	90
0x16	R_CHIP_ID	H	91
0x17	R_BERT_STA	0, 1	216
0x18	R_F0_CNTL	0, 1	193
0x19	R_F0_CNTH	0, 1	193
0x1A	R_BERT_ECL	0, 1	216
0x1B	R_BERT_ECH	0, 1	217
0x1C	R_STATUS	–	242
0x1F	R_CHIP_RV	–	91
0x20	R_STATE	0, 3	169
0x24	R_RX_STA0	0, 3	170
0x25	R_RX_STA1	0, 3	171
0x26	R_RX_STA2	0, 3	171
0x27	R_RX_STA3	0, 3	172
0x2C	R_SLIP	0, 3	172
0x30	R_FAS_ECL	0, 3	173
0x31	R_FAS_ECH	0, 3	173
0x32	R_VIO_ECL	0, 3	173
0x33	R_VIO_ECH	0, 3	174
0x34	R_CRC_ECL	0, 3	174
0x35	R_CRC_ECH	0, 3	174
0x36	R_E_ECL	0, 3	175
0x37	R_E_ECH	0, 3	175
0x38	R_SA6_SA13_ECL	0, 3	175
0x39	R_SA6_SA13_ECH	0, 3	176
0x3A	R_SA6_SA23_ECL	0, 3	176
0x3B	R_SA6_SA23_ECH	0, 3	176
0x40	R_GPIO_IN0	–	259
0x41	R_GPIO_IN1	–	260
0x44	R_GPI_IN0	–	261
0x45	R_GPI_IN1	–	262
0x46	R_GPI_IN2	–	263
0x47	R_GPI_IN3	–	264
0x88	R_INT_DATA	–	142
0xC8	R_IRQ_FIFO_BL0	0, 1	243
0xC9	R_IRQ_FIFO_BL1	0, 1	244
0xCA	R_IRQ_FIFO_BL2	0, 1	245
0xCB	R_IRQ_FIFO_BL3	0, 1	246
0xCC	R_IRQ_FIFO_BL4	0, 1	247
0xCD	R_IRQ_FIFO_BL5	0, 1	248
0xCE	R_IRQ_FIFO_BL6	0, 1	249
0xCF	R_IRQ_FIFO_BL7	0, 1	250

Read / Write registers:

Address	Name	Reset group	Page
0x80	A_FIFO_DATA2	–	144
0x80	A_FIFO_DATA0	–	143
0x80	A_FIFO_DATA1	–	143
0x84	A_FIFO_DATA2_NOINC	–	145
0x84	A_FIFO_DATA0_NOINC	–	144
0x84	A_FIFO_DATA1_NOINC	–	145
0xC0	R_RAM_DATA	–	90

Note: See table 12.4 on page 235 for ‘Reset group’ explanation.

Chapter 1

General description

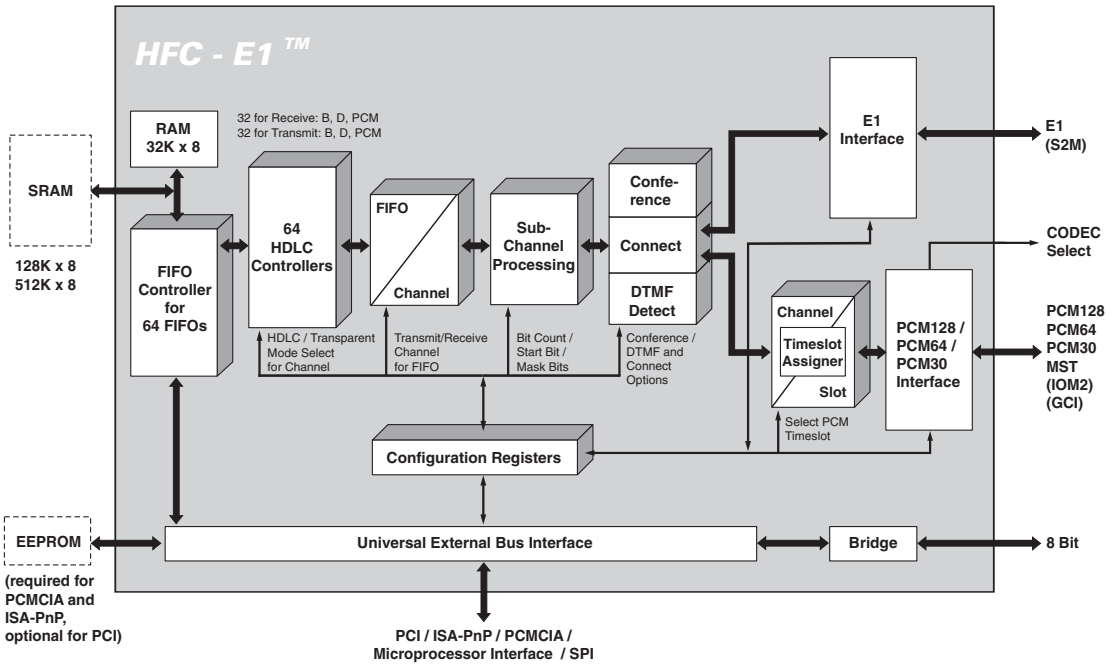

Figure 1.1: *HFC-E1 block diagram*

1.1 System overview

The HFC-E1 is an ISDN E1 HDLC primary rate controller for all kinds of PRI equipment, such as

- high performance ISDN PC cards
- ISDN PRI terminal adapters
- ISDN PABX for PRI
- VoIP gateways
- Integrated Access Devices (IAD)
- ISDN LAN routers for PRI
- ISDN least cost routers for PRI
- ISDN test equipment for PRI

The integrated universal bus interface of the HFC-E1 can be configured to PCI, ISA Plug and Play, PCMCIA, microprocessor interface or SPI. A PCM128 / PCM64 / PCM30 interface for CODEC or inter chip connection is also integrated. The very deep FIFOs of the HFC-E1 is realized with an internal or external SRAM.

1.2 Features

- integrated E1 interface
- single chip ISDN-E1 controller with HDLC support for all B- and D-channels
- full I.431 ITU E1 ISDN support in TE , NT and LT mode
- 32 independent read and write HDLC channels for e.g. 30 ISDN B-channels, 1 ISDN D-channel
- B-channel transparent mode independently selectable
- up to 32 FIFOs for transmit and for receive data, FIFO sizes are configurable
- each FIFO can be assigned to an arbitrary HFC-channel, moreover each HFC-channel can be assigned to a time slot of the E1 interface or to a time slot of the PCM interface
- max. 31 HDLC frames (with 128 kByte or 512 kByte external RAM) or 15 HDLC frames (with 32 kByte build-in RAM) per FIFO
- 1 . . . 8 bit processing for subchannels selectable
- B-channels for higher data rate can be combined up to 256 bit
- PCM128 / PCM64 / PCM30 interface configurable to interface MST™(MVIP™) [1] or Siemens IOM2™and Motorola GCI™(no monitor or C/I-channel support) for inter chip connection or external CODECs [2]
- Switch matrix for PCM included
- H.100 data rate supported
- integrated ISA Plug and Play interface with buffers for ISA-databus
- integrated PCMCIA interface
- integrated PCI bus interface (Spec. 2.2) for 3.3 V and 5 V signal environment
- microprocessor interface compatible to Motorala bus and Siemens / Intel bus
- Serial processor interface (SPI)
- multiparty audio conferences switchable
- DTMF detection on all 32 channels
- Timer and watchdog with interrupt capability
- CMOS technology 3.3 V (5 V tolerant on nearly all inputs [3])
- PQFP 208 package

[1] Mitel Serial Telecom bus
[2] All ™ marked names are registered trademarks of the appropriate organizations.
[3] Never connect the power supply of the HFC-E1 to 5 V!

1.3 Pin description

1.3.1 Pinout diagram

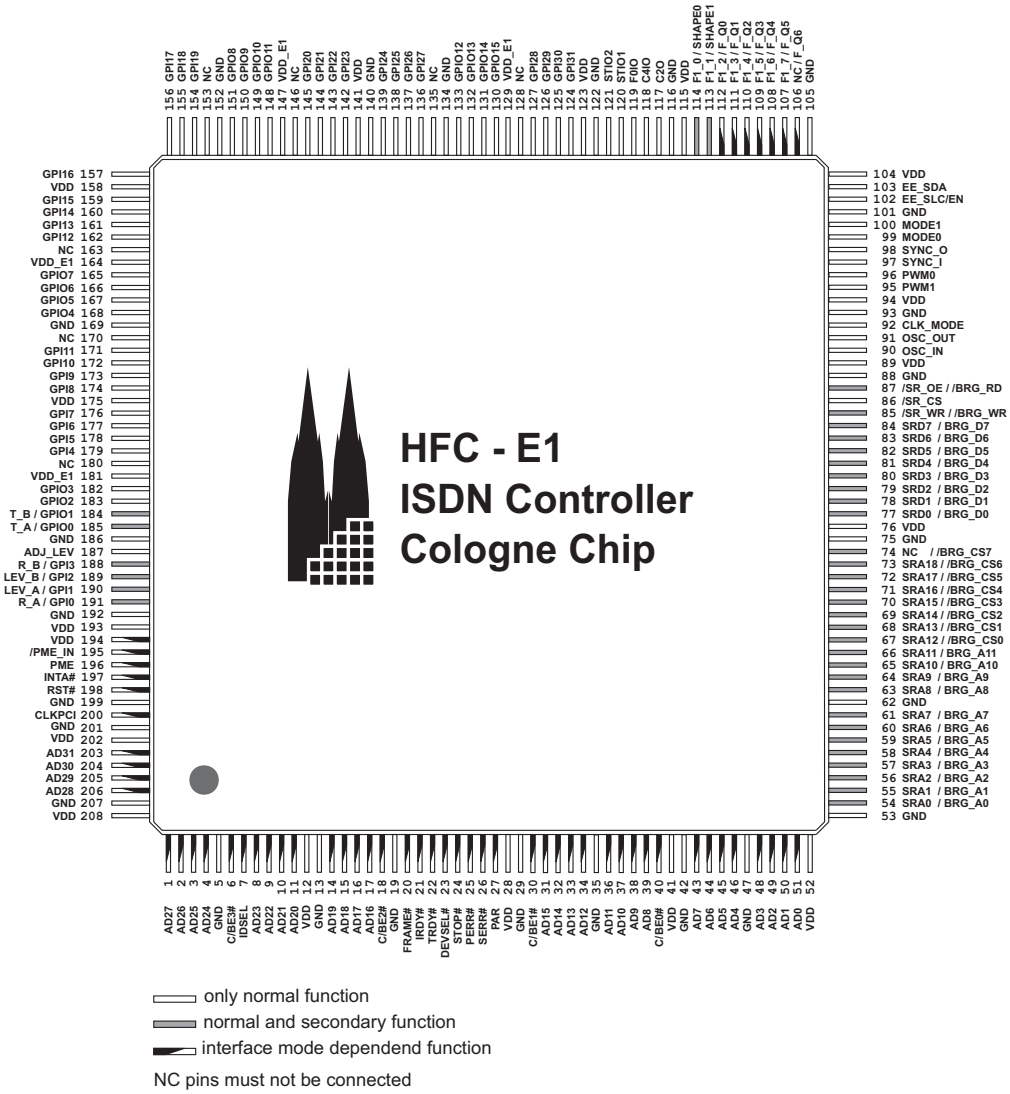

Figure 1.2: *HFC-E1 pinout in PCI mode*

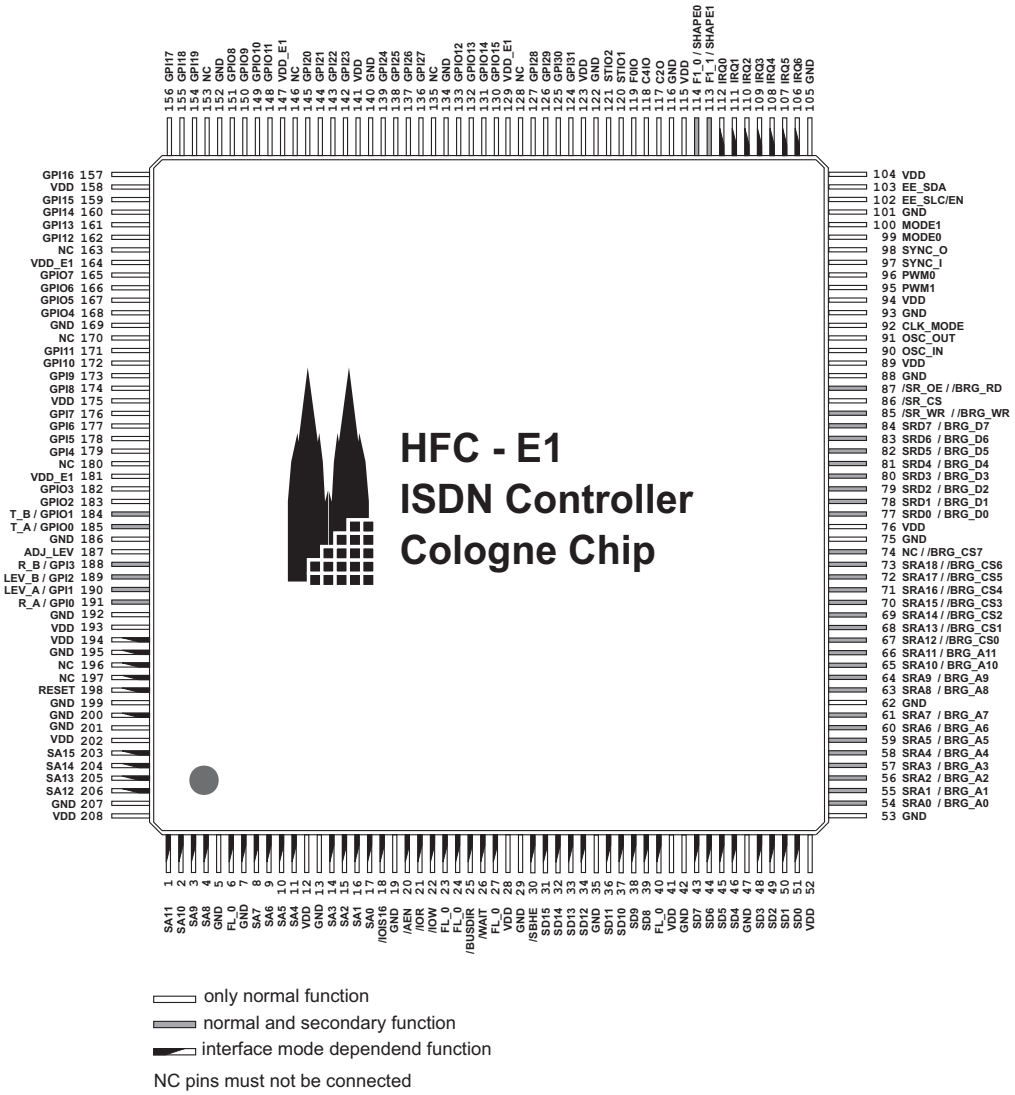

Figure 1.3: *HFC-E1 pinout in ISA PnP mode*

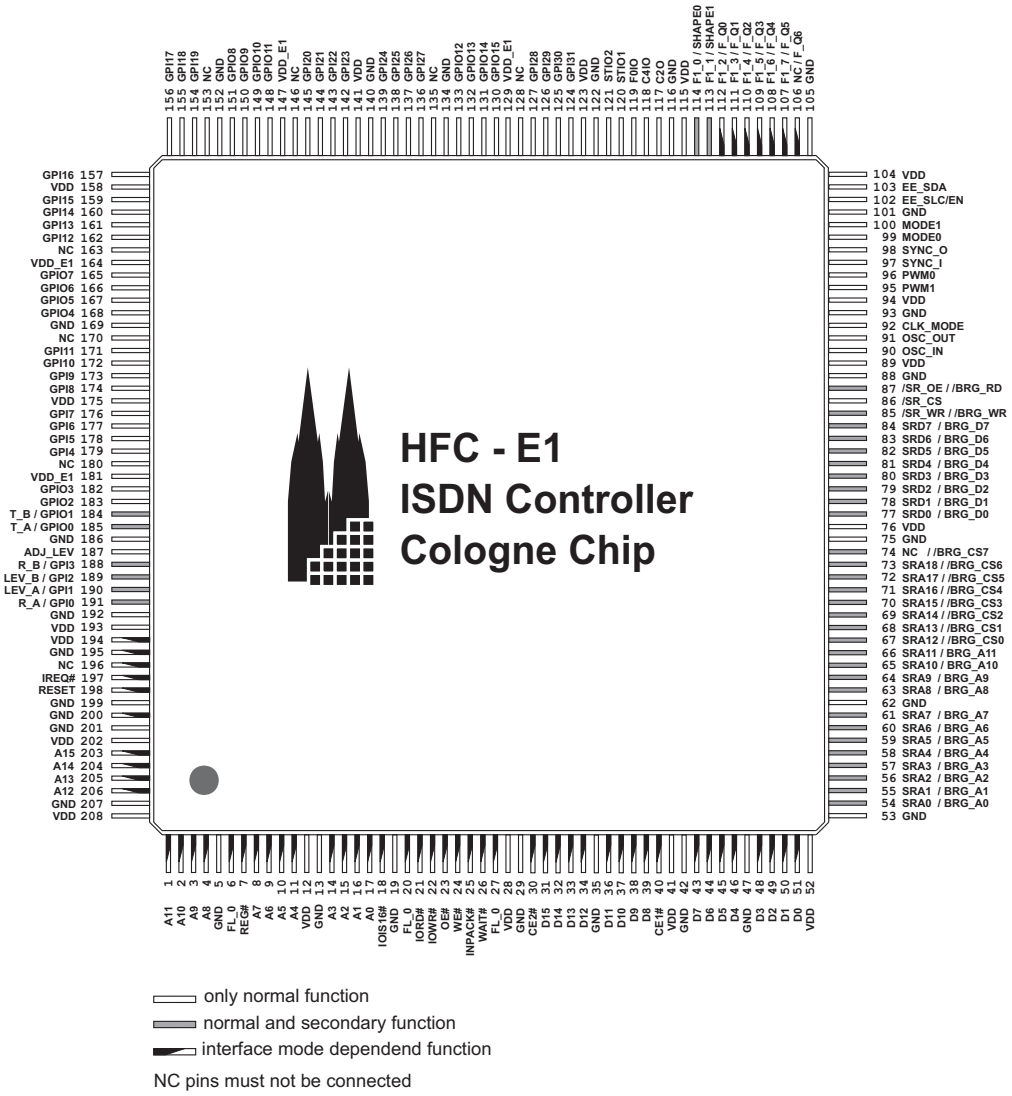

Figure 1.4: *HFC-E1 pinout in PCMCIA mode*

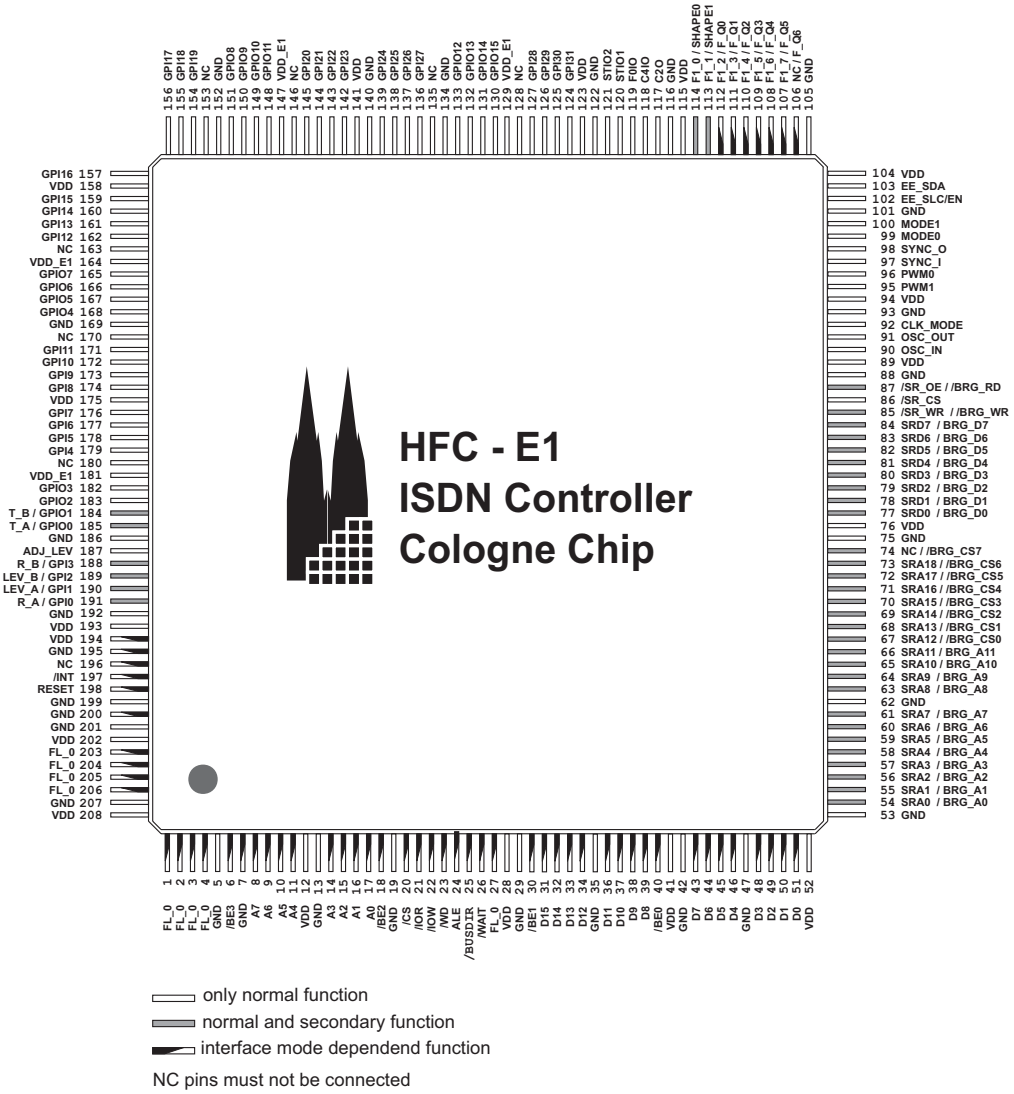

Figure 1.5: *HFC-E1 pinout in processor mode*

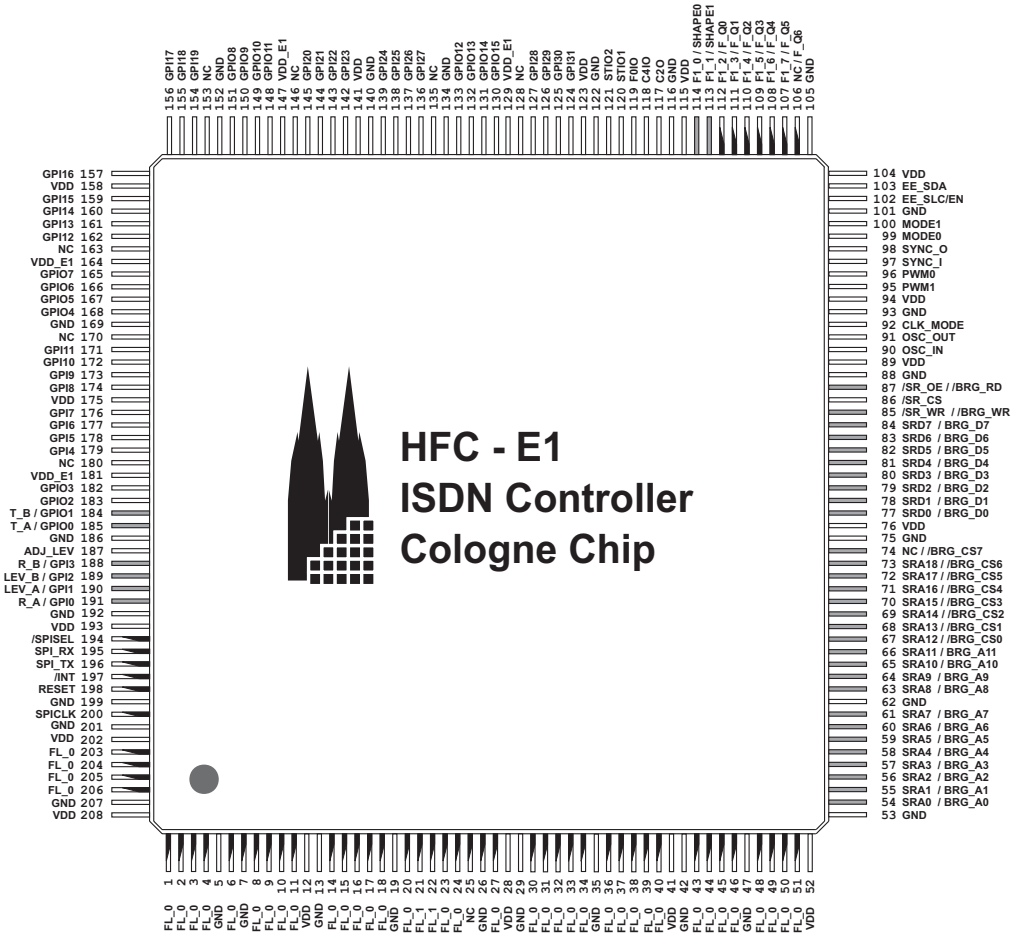

Figure 1.6: *HFC-E1 pinout in SPI mode*

1.3.2 Pin list

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
				Universal bus interface		
1	PCI	AD27	IO	Address / Data bit 27	LVCMOS	8
	ISA PnP	SA11	I	Address bit 11	LVCMOS	
	PCMCIA	A11	I	Address bit 11	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
2	PCI	AD26	IO	Address / Data bit 26	LVCMOS	8
	ISA PnP	SA10	I	Address bit 10	LVCMOS	
	PCMCIA	A10	I	Address bit 10	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
3	PCI	AD25	IO	Address / Data bit 25	LVCMOS	8
	ISA PnP	SA9	I	Address bit 9	LVCMOS	
	PCMCIA	A9	I	Address bit 9	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
4	PCI	AD24	IO	Address / Data bit 24	LVCMOS	8
	ISA PnP	SA8	I	Address bit 8	LVCMOS	
	PCMCIA	A8	I	Address bit 8	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
5		GND		Ground		
6	PCI	C/BE3#	I	Bus command and Byte Enable 3	LVCMOS	
	ISA PnP	FL1	I	Fixed level (high), connect to power supply via ext. pull-up	LVCMOS	
	PCMCIA	FL1	I	Fixed level (high), connect to power supply via ext. pull-up	LVCMOS	
	Processor	/BE3	I	Byte Enable 3	LVCMOS	
	SPI	FL1	I	Fixed level (high), connect to power supply via ext. pull-up	LVCMOS	
7	PCI	IDSEL	I	Initialisation Device Select	LVCMOS	
	ISA PnP	GND	I	Ground	LVCMOS	
	PCMCIA	REG#	I	PCMCIA Register and Attr. Mem. Select	LVCMOS	
	Processor	GND	I	Ground	LVCMOS	
	SPI	GND	I	Ground	LVCMOS	

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
8	PCI	AD23	IO	Address / Data bit 23	LVCMOS	8
	ISA PnP	SA7	I	Address bit 7	LVCMOS	
	PCMCIA	A7	I	Address bit 7	LVCMOS	
	Processor	A7	I	Address bit 7	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
9	PCI	AD22	IO	Address / Data bit 22	LVCMOS	8
	ISA PnP	SA6	I	Address bit 6	LVCMOS	
	PCMCIA	A6	I	Address bit 6	LVCMOS	
	Processor	A6	I	Address bit 6	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
10	PCI	AD21	IO	Address / Data bit 21	LVCMOS	8
	ISA PnP	SA5	I	Address bit 5	LVCMOS	
	PCMCIA	A5	I	Address bit 5	LVCMOS	
	Processor	A5	I	Address bit 5	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
11	PCI	AD20	IO	Address / Data bit 20	LVCMOS	8
	ISA PnP	SA4	I	Address bit 4	LVCMOS	
	PCMCIA	A4	I	Address bit 4	LVCMOS	
	Processor	A4	I	Address bit 4	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
12		VDD		+3.3 V power supply		
13		GND		Ground		
14	PCI	AD19	IO	Address / Data bit 19	LVCMOS	8
	ISA PnP	SA3	I	Address bit 3	LVCMOS	
	PCMCIA	A3	I	Address bit 3	LVCMOS	
	Processor	A3	I	Address bit 3	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
15	PCI	AD18	IO	Address / Data bit 18	LVCMOS	8
	ISA PnP	SA2	I	Address bit 2	LVCMOS	
	PCMCIA	A2	I	Address bit 2	LVCMOS	
	Processor	A2	I	Address bit 2	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
16	PCI	AD17	IO	Address / Data bit 17	LVCMOS	8
	ISA PnP	SA1	I	Address bit 1	LVCMOS	
	PCMCIA	A1	I	Address bit 1	LVCMOS	
	Processor	A1	I	Address bit 1	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
17	PCI	AD16	IO	Address / Data bit 16	LVCMOS	8
	ISA PnP	SA0	I	Address bit 0	LVCMOS	
	PCMCIA	A0	I	Address bit 0	LVCMOS	
	Processor	A0	I	Address bit 0	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
18	PCI	C/BE2#	I	Bus command and Byte Enable 2	LVCMOS	
	ISA PnP	/IOIS16	Ood	16 bit access enable		8
	PCMCIA	IOIS16#	O	16 bit access enable		8
	Processor	/BE2	I	Byte Enable 2	LVCMOS	
	SPI	FL1	I	Fixed level (high), connect to power supply via ext. pull-up	LVCMOS	
19		GND		Ground		
20	PCI	FRAME#	I	Cycle Frame	LVCMOS	
	ISA PnP	/AEN	I	Address Enable	LVCMOS	
	PCMCIA	GND		Ground		
	Processor	/CS	I	Chip Select	LVCMOS	
	SPI	VDD		+3.3 V power supply		
21	PCI	IRDY#	I	Initiator Ready	LVCMOS	
	ISA PnP	/IOR	I	Read Enable	LVCMOS	
	PCMCIA	IORD#	I	Read Enable	LVCMOS	
	Processor	/IOR	I	Read Enable	LVCMOS	
	SPI	VDD		+3.3 V power supply		
22	PCI	TRDY#	O	Target Ready		8
	ISA PnP	/IOW	I	Write Enable	LVCMOS	
	PCMCIA	IOWR#	I	Write Enable	LVCMOS	
	Processor	/IOW	I	Write Enable	LVCMOS	
	SPI	FL1	I	Fixed level (high), connect to power supply via ext. pull-up	LVCMOS	
23	PCI	DEVSEL#	O	Device Select		8
	ISA PnP	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	PCMCIA	OE#	I	PCMCIA Output Enable for Attr. Mem. Read	LVCMOS	
	Processor	/WD	Ood	Watch Dog Output		8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
24	PCI	STOP#	O	Stop		8
	ISA PnP	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	PCMCIA	WE#	I	PCMCIA Write Enable for Conf. Reg. Write	LVCMOS	
	Processor	ALE	I	Address Latch Enable	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
25	PCI	PERR#	IO	Parity Error	LVCMOS	8
	ISA PnP	/BUSDIR	O	Bus Direction		8
	PCMCIA	INPACK#	O	Read access		8
	Processor	/BUSDIR	O	Bus Direction		8
	SPI	NC				
26	PCI	SERR#	Ood	System Error		8
	ISA PnP	NC				
	PCMCIA	NC				
	Processor	NC				
	SPI	NC				
27	PCI	PAR	IO	Parity Bit	LVCMOS	8
	ISA PnP	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	PCMCIA	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
28		VDD		+3.3 V power supply		
29		GND		Ground		
30	PCI	C/BE1#	I	Bus command and Byte Enable 1	LVCMOS	
	ISA PnP	/SBHE	I	High byte enable	LVCMOS	
	PCMCIA	CE2#	I	High byte enable	LVCMOS	
	Processor	/BE1	I	Byte Enable 1	LVCMOS	
	SPI	FL1	I	Fixed level (high), connect to power supply via ext. pull-up	LVCMOS	
31	PCI	AD15	IO	Address / Data bit 15	LVCMOS	8
	ISA PnP	SD15	IO	ISA Data Bus Bit 15	LVCMOS	8
	PCMCIA	D15	IO	PCMCIA Data Bus Bit 15	LVCMOS	8
	Processor	D15	IO	Data bit 15	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
32	PCI	AD14	IO	Address / Data bit 14	LVCMOS	8
	ISA PnP	SD14	IO	ISA Data Bus Bit 14	LVCMOS	8
	PCMCIA	D14	IO	PCMCIA Data Bus Bit 14	LVCMOS	8
	Processor	D14	IO	Data bit 14	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
33	PCI	AD13	IO	Address / Data bit 13	LVCMOS	8
	ISA PnP	SD13	IO	ISA Data Bus Bit 13	LVCMOS	8
	PCMCIA	D13	IO	PCMCIA Data Bus Bit 13	LVCMOS	8
	Processor	D13	IO	Data bit 13	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
34	PCI	AD12	IO	Address / Data bit 12	LVCMOS	8
	ISA PnP	SD12	IO	ISA Data Bus Bit 12	LVCMOS	8
	PCMCIA	D12	IO	PCMCIA Data Bus Bit 12	LVCMOS	8
	Processor	D12	IO	Data bit 12	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
35		GND		Ground		
36	PCI	AD11	IO	Address / Data bit 11	LVCMOS	8
	ISA PnP	SD11	IO	ISA Data Bus Bit 11	LVCMOS	8
	PCMCIA	D11	IO	PCMCIA Data Bus Bit 11	LVCMOS	8
	Processor	D11	IO	Data bit 11	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
37	PCI	AD10	IO	Address / Data bit 10	LVCMOS	8
	ISA PnP	SD10	IO	ISA Data Bus Bit 10	LVCMOS	8
	PCMCIA	D10	IO	PCMCIA Data Bus Bit 10	LVCMOS	8
	Processor	D10	IO	Data bit 10	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
38	PCI	AD9	IO	Address / Data bit 9	LVCMOS	8
	ISA PnP	SD9	IO	ISA Data Bus Bit 9	LVCMOS	8
	PCMCIA	D9	IO	PCMCIA Data Bus Bit 9	LVCMOS	8
	Processor	D9	IO	Data bit 9	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
39	PCI	AD8	IO	Address / Data bit 8	LVCMOS	8
	ISA PnP	SD8	IO	ISA Data Bus Bit 8	LVCMOS	8
	PCMCIA	D8	IO	PCMCIA Data Bus Bit 8	LVCMOS	8
	Processor	D8	IO	Data bit 8	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
40	PCI	C/BE0#	I	Bus command and Byte Enable 0	LVCMOS	
	ISA PnP	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
	PCMCIA	CE1#	I	Low byte enable	LVCMOS	
	Processor	/BE0	I	Byte Enable 0	LVCMOS	
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
41		VDD		+3.3 V power supply		
42		GND		Ground		
43	PCI	AD7	IO	Address / Data bit 7	LVCMOS	8
	ISA PnP	SD7	IO	ISA Data Bus Bit 7	LVCMOS	8
	PCMCIA	D7	IO	PCMCIA Data Bus Bit 7	LVCMOS	8
	Processor	D7	IO	Data bit 7	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
44	PCI	AD6	IO	Address / Data bit 6	LVCMOS	8
	ISA PnP	SD6	IO	ISA Data Bus Bit 6	LVCMOS	8
	PCMCIA	D6	IO	PCMCIA Data Bus Bit 6	LVCMOS	8
	Processor	D6	IO	Data bit 6	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
45	PCI	AD5	IO	Address / Data bit 5	LVCMOS	8
	ISA PnP	SD5	IO	ISA Data Bus Bit 5	LVCMOS	8
	PCMCIA	D5	IO	PCMCIA Data Bus Bit 5	LVCMOS	8
	Processor	D5	IO	Data bit 5	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
46	PCI	AD4	IO	Address / Data bit 4	LVCMOS	8
	ISA PnP	SD4	IO	ISA Data Bus Bit 4	LVCMOS	8
	PCMCIA	D4	IO	PCMCIA Data Bus Bit 4	LVCMOS	8
	Processor	D4	IO	Data bit 4	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
47		GND		Ground		
48	PCI	AD3	IO	Address / Data bit 3	LVCMOS	8
	ISA PnP	SD3	IO	ISA Data Bus Bit 3	LVCMOS	8
	PCMCIA	D3	IO	PCMCIA Data Bus Bit 3	LVCMOS	8
	Processor	D3	IO	Data bit 3	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
49	PCI	AD2	IO	Address / Data bit 2	LVCMOS	8
	ISA PnP	SD2	IO	ISA Data Bus Bit 2	LVCMOS	8
	PCMCIA	D2	IO	PCMCIA Data Bus Bit 2	LVCMOS	8
	Processor	D2	IO	Data bit 2	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
50	PCI	AD1	IO	Address / Data bit 1	LVCMOS	8
	ISA PnP	SD1	IO	ISA Data Bus Bit 1	LVCMOS	8
	PCMCIA	D1	IO	PCMCIA Data Bus Bit 1	LVCMOS	8
	Processor	D1	IO	Data bit 1	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
51	PCI	AD0	IO	Address / Data bit 0	LVCMOS	8
	ISA PnP	SD0	IO	ISA Data Bus Bit 0	LVCMOS	8
	PCMCIA	D0	IO	PCMCIA Data Bus Bit 0	LVCMOS	8
	Processor	D0	IO	Data bit 0	LVCMOS	8
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down	LVCMOS	
52		VDD		+3.3 V power supply		
53		GND		Ground		

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
				SRAM / Auxiliary interface		
54	1st function	SRA0	O	Address bit 0 for external SRAM		2
	2nd function	BRG_A0	O	Bridge Address bit 0		2
55	1st function	SRA1	O	Address bit 1 for external SRAM		2
	2nd function	BRG_A1	O	Bridge Address bit 1		2
56	1st function	SRA2	O	Address bit 2 for external SRAM		2
	2nd function	BRG_A2	O	Bridge Address bit 2		2
57	1st function	SRA3	O	Address bit 3 for external SRAM		2
	2nd function	BRG_A3	O	Bridge Address bit 3		2
58	1st function	SRA4	O	Address bit 4 for external SRAM		2
	2nd function	BRG_A4	O	Bridge Address bit 4		2
59	1st function	SRA5	O	Address bit 5 for external SRAM		2
	2nd function	BRG_A5	O	Bridge Address bit 5		2
60	1st function	SRA6	O	Address bit 6 for external SRAM		2
	2nd function	BRG_A6	O	Bridge Address bit 6		2
61	1st function	SRA7	O	Address bit 7 for external SRAM		2
	2nd function	BRG_A7	O	Bridge Address bit 7		2
62		GND		Ground		
63	1st function	SRA8	O	Address bit 8 for external SRAM		2
	2nd function	BRG_A8	O	Bridge Address bit 8		2
64	1st function	SRA9	O	Address bit 9 for external SRAM		2
	2nd function	BRG_A9	O	Bridge Address bit 9		2
65	1st function	SRA10	O	Address bit 10 for external SRAM		2
	2nd function	BRG_A10	O	Bridge Address bit 10		2
66	1st function	SRA11	O	Address bit 11 for external SRAM		2
	2nd function	BRG_A11	O	Bridge Address bit 11		2
67	1st function	SRA12	O	Address bit 12 for external SRAM		2
	2nd function	/BRG_CS0	O	Bridge Chip Select 0		2
68	1st function	SRA13	O	Address bit 13 for external SRAM		2
	2nd function	/BRG_CS1	O	Bridge Chip Select 1		2
69	1st function	SRA14	O	Address bit 14 for external SRAM		2
	2nd function	/BRG_CS2	O	Bridge Chip Select 2		2
70	1st function	SRA15	O	Address bit 15 for external SRAM		2
	2nd function	/BRG_CS3	O	Bridge Chip Select 3		2
71	1st function	SRA16	O	Address bit 16 for external SRAM		2
	2nd function	/BRG_CS4	O	Bridge Chip Select 4		2
72	1st function	SRA17	O	Address bit 17 for external SRAM		2
	2nd function	/BRG_CS5	O	Bridge Chip Select 5		2

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
73	1st function	SRA18	O	Address bit 18 for external SRAM		2
	2nd function	/BRG_CS6	O	Bridge Chip Select 6		2
74	1st function	NC				
	2nd function	/BRG_CS7	O	Bridge Chip Select 7		2
75		GND		Ground		
76		VDD		+3.3 V power supply		
77	1st function	SRD0	IO	Data bit 0 for external SRAM	LVCMOS	8
	2nd function	BRG_D0	IO	Bridge Data bit 0	LVCMOS	8
78	1st function	SRD1	IO	Data bit 1 for external SRAM	LVCMOS	8
	2nd function	BRG_D1	IO	Bridge Data bit 1	LVCMOS	8
79	1st function	SRD2	IO	Data bit 2 for external SRAM	LVCMOS	8
	2nd function	BRG_D2	IO	Bridge Data bit 2	LVCMOS	8
80	1st function	SRD3	IO	Data bit 3 for external SRAM	LVCMOS	8
	2nd function	BRG_D3	IO	Bridge Data bit 3	LVCMOS	8
81	1st function	SRD4	IO	Data bit 4 for external SRAM	LVCMOS	8
	2nd function	BRG_D4	IO	Bridge Data bit 4	LVCMOS	8
82	1st function	SRD5	IO	Data bit 5 for external SRAM	LVCMOS	8
	2nd function	BRG_D5	IO	Bridge Data bit 5	LVCMOS	8
83	1st function	SRD6	IO	Data bit 6 for external SRAM	LVCMOS	8
	2nd function	BRG_D6	IO	Bridge Data bit 6	LVCMOS	8
84	1st function	SRD7	IO	Data bit 7 for external SRAM	LVCMOS	8
	2nd function	BRG_D7	IO	Bridge Data bit 7	LVCMOS	8
85	1st function	/SR_WR	O	Write enable for external SRAM		4
	2nd function	/BRG_WR	O	Bridge Write enable / RD/WR		4
86		/SR_CS	O	Chip Select for external SRAM		4
87	1st function	/SR_OE	O	Output enable for external SRAM		4
	2nd function	/BRG_RD	O	Bridge Read enable / /DS		4
88		GND		Ground		
89		VDD		+3.3 V power supply		
				Clock		
90		OSC_IN	I	Oscillator Input Signal		
91		OSC_OUT	O	Oscillator Output Signal		
92		CLK_MODE	I	Clock Mode	LVCMOS	
93		GND		Ground		
94		VDD		+3.3 V power supply		
				Miscellaneous		

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
95		PWM1	O	Pulse Width Modulator Output 1		8
96		PWM0	O	Pulse Width Modulator Output 0		8
97		SYNC_I	I	Synchronization Input	LVCMOS	
98		SYNC_O	O	Synchronization Output		4
99		MODE0	I	Interface Mode pin 0	LVCMOS	
100		MODE1	I	Interface Mode pin 1	LVCMOS	
101		GND		Ground		
				EEPROM		
102		EE_SCL/EN	IO	EEPROM clock / EEPROM enable	LVCMOS	1
103		EE_SDA	IO	EEPROM data I/O	LVCMOS	1
104		VDD		+3.3 V power supply		
105		GND		Ground		
				PCM		
106	1st function	NC				
	2nd function	F_Q6	O	PCM time slot count 6		6
	ISA PnP	IRQ6	O	ISA Interrupt Request 6		6
107	1st function	F1_7	O	PCM CODEC enable 7		6
	2nd function	F_Q5	O	PCM time slot count 5		6
	ISA PnP	IRQ5	O	ISA Interrupt Request 5		6
108	1st function	F1_6	O	PCM CODEC enable 6		6
	2nd function	F_Q4	O	PCM time slot count 4		6
	ISA PnP	IRQ4	O	ISA Interrupt Request 4		6
109	1st function	F1_5	O	PCM CODEC enable 5		6
	2nd function	F_Q3	O	PCM time slot count 3		6
	ISA PnP	IRQ3	O	ISA Interrupt Request 3		6
110	1st function	F1_4	O	PCM CODEC enable 4		6
	2nd function	F_Q2	O	PCM time slot count 2		6
	ISA PnP	IRQ2	O	ISA Interrupt Request 2		6
111	1st function	F1_3	O	PCM CODEC enable 3		6
	2nd function	F_Q1	O	PCM time slot count 1		6
	ISA PnP	IRQ1	O	ISA Interrupt Request 1		6
112	1st function	F1_2	O	PCM CODEC enable 2		6
	2nd function	F_Q0	O	PCM time slot count 0		6
	ISA PnP	IRQ0	O	ISA Interrupt Request 0		6
113	1st function	F1_1	O	PCM CODEC enable 1		6
	2nd function	SHAPE1	O	PCM CODEC enable shape signal 1		6

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
114	1st function 2nd function	F1_0 SHAPE0	O O	PCM CODEC enable 0 PCM CODEC enable shape signal 0		6 6
115		VDD		+3.3 V power supply		
116		GND		Ground		
117		C2O	O	PCM bit clock output		8
118		C4IO	IOpu	PCM double bit clock I/O	LVCMOS	8
119		F0IO	IOpu	PCM frame clock I/O (8 kHz)	LVCMOS	8
120		STIO1	IOpu	PCM data bus 1, I or O per time slot	LVCMOS	8
121		STIO2	IOpu	PCM data bus 2, I or O per time slot	LVCMOS	8
122		GND		Ground		
123		VDD		+3.3 V power supply		
				GPIO		
124		GPI31	I	General Purpose Input pin 31	LVCMOS	
125		GPI30	I	General Purpose Input pin 30	LVCMOS	
126		GPI29	I	General Purpose Input pin 29	LVCMOS	
127		GPI28	I	General Purpose Input pin 28	LVCMOS	
128		NC				
129		VDD_E1		app. +2.8 V power supply (depends on the E1 transmit amplitude)		
130		GPIO15	IO	General Purpose I/O pin 15	LVCMOS	16
131		GPIO14	IO	General Purpose I/O pin 14	LVCMOS	16
132		GPIO13	IO	General Purpose I/O pin 13	LVCMOS	16
133		GPIO12	IO	General Purpose I/O pin 12	LVCMOS	16
134		GND		Ground		
135		NC				
136		GPI27	I	General Purpose Input pin 27	LVCMOS	
137		GPI26	I	General Purpose Input pin 26	LVCMOS	
138		GPI25	I	General Purpose Input pin 25	LVCMOS	
139		GPI24	I	General Purpose Input pin 24	LVCMOS	
140		GND		Ground		

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
141		VDD		+3.3 V power supply		
142		GPI23	I	General Purpose Input pin 23	LVCMOS	
143		GPI22	I	General Purpose Input pin 22	LVCMOS	
144		GPI21	I	General Purpose Input pin 21	LVCMOS	
145		GPI20	I	General Purpose Input pin 20	LVCMOS	
146		NC				
147		VDD_E1		app. +2.8 V power supply (depends on the E1 transmit amplitude)		
148		GPIO11	IO	General Purpose I/O pin 11	LVCMOS	16
149		GPIO10	IO	General Purpose I/O pin 10	LVCMOS	16
150		GPIO9	IO	General Purpose I/O pin 9	LVCMOS	16
151		GPIO8	IO	General Purpose I/O pin 8	LVCMOS	16
152		GND		Ground		
153		NC				
154		GPI19	I	General Purpose Input pin 19	LVCMOS	
155		GPI18	I	General Purpose Input pin 18	LVCMOS	
156		GPI17	I	General Purpose Input pin 17	LVCMOS	
157		GPI16	I	General Purpose Input pin 16	LVCMOS	
158		VDD		+3.3 V power supply		
159		GPI15	I	General Purpose Input pin 15	LVCMOS	
160		GPI14	I	General Purpose Input pin 14	LVCMOS	
161		GPI13	I	General Purpose Input pin 13	LVCMOS	
162		GPI12	I	General Purpose Input pin 12	LVCMOS	
163		NC				
164		VDD_E1		app. +2.8 V power supply (depends on the E1 transmit amplitude)		
165		GPIO7	IO	General Purpose I/O pin 7	LVCMOS	16
166		GPIO6	IO	General Purpose I/O pin 6	LVCMOS	16
167		GPIO5	IO	General Purpose I/O pin 5	LVCMOS	16
168		GPIO4	IO	General Purpose I/O pin 4	LVCMOS	16
169		GND		Ground		

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
170		NC				
171		GPI11	I	General Purpose Input pin 11	LVCMOS	
172		GPI10	I	General Purpose Input pin 10	LVCMOS	
173		GPI9	I	General Purpose Input pin 9	LVCMOS	
174		GPI8	I	General Purpose Input pin 8	LVCMOS	
175		VDD		+3.3 V power supply		
176		GPI7	I	General Purpose Input pin 7	LVCMOS	
177		GPI6	I	General Purpose Input pin 6	LVCMOS	
178		GPI5	I	General Purpose Input pin 5	LVCMOS	
179		GPI4	I	General Purpose Input pin 4	LVCMOS	
180		NC				
181		VDD_E1		app. +2.8 V power supply (depends on the E1 transmit amplitude)		
182		GPIO3	IO	General Purpose I/O pin 3	LVCMOS	16
183		GPIO2	IO	General Purpose I/O pin 2	LVCMOS	16
				E1 interface		
184	1st function	T_B	O	E1 interface transmit data B		16
	2nd function	GPIO1	IO	General Purpose I/O pin 1	LVCMOS	16
185	1st function	T_A	O	E1 interface transmit data A		16
	2nd function	GPIO0	IO	General Purpose I/O pin 0	LVCMOS	16
186		GND		Ground		
187		ADJ_LEV	Ood	E1 interface level generator		
188	1st function	R_B	I	E1 interface receive input B	E1	
	2nd function	GPI3	I	General Purpose Input pin 3	LVCMOS	
189	1st function	LEV_B	I	E1 interface level detect B	E1	
	2nd function	GPI2	I	General Purpose Input pin 2	LVCMOS	
190	1st function	LEV_A	I	E1 interface level detect A	E1	
	2nd function	GPI1	I	General Purpose Input pin 1	LVCMOS	
191	1st function	R_A	I	E1 interface receive input A	E1	
	2nd function	GPI0	I	General Purpose Input pin 0	LVCMOS	
192		GND		Ground		
193		VDD		+3.3 V power supply		
				Universal bus interface		

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
194	PCI	VDD	I	+3.3 V power supply	LVCMOS	
	ISA PnP	VDD	I	+3.3 V power supply	LVCMOS	
	PCMCIA	VDD	I	+3.3 V power supply	LVCMOS	
	Processor	VDD	I	+3.3 V power supply	LVCMOS	
	SPI	/SPISEL	I	SPI device select low active	LVCMOS	
195	PCI	PME_IN	I	Power Management Event Input	LVCMOS	
	ISA PnP	GND		Ground		
	PCMCIA	GND		Ground		
	Processor	GND		Ground		
	SPI	SPI_RX	I	SPI receive data input	LVCMOS	
196	PCI	PME	O	Power Management Event output		4
	ISA PnP	NC				
	PCMCIA	NC				
	Processor	NC				
	SPI	SPI_TX	O	SPI transmit data output		4
197	PCI	INTA#	Ood	Interrupt request		4
	ISA PnP	NC				
	PCMCIA	IREQ#	Ood	Interrupt request		4
	Processor	/INT	Ood	Interrupt request		4
	SPI	/INT	Ood	Interrupt request		4
198	PCI	RST#	I	Reset low active	LVCMOS	
	ISA PnP	RESET	I	Reset high active	LVCMOS	
	PCMCIA	RESET	I	Reset high active	LVCMOS	
	Processor	RESET	I	Reset high active	LVCMOS	
	SPI	RESET	I	Reset high active	LVCMOS	
199		GND		Ground		
200	PCI	PCICLK	I	PCI Clock Input	LVCMOS	
	ISA PnP	GND		Ground		
	PCMCIA	GND		Ground		
	Processor	GND		Ground		
	SPI	SPICLK	I	SPI clock input	LVCMOS	
201		GND		Ground		
202		VDD		+3.3 V power supply		
203	PCI	AD31	IO	Address / Data bit 31	LVCMOS	8
	ISA PnP	SA15	I	Address bit 15	LVCMOS	
	PCMCIA	A15	I	Address bit 15	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down		
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down		

(continued on next page)

(continued from previous page)

Pin	Interface	Name	I/O	Description	U_{in} / V	I_{out} / mA
204	PCI	AD30	IO	Address / Data bit 30	LVCMOS	8
	ISA PnP	SA14	I	Address bit 14	LVCMOS	
	PCMCIA	A14	I	Address bit 14	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down		
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down		
205	PCI	AD29	IO	Address / Data bit 29	LVCMOS	8
	ISA PnP	SA13	I	Address bit 13	LVCMOS	
	PCMCIA	A13	I	Address bit 13	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down		
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down		
206	PCI	AD28	IO	Address / Data bit 28	LVCMOS	8
	ISA PnP	SA12	I	Address bit 12	LVCMOS	
	PCMCIA	A12	I	Address bit 12	LVCMOS	
	Processor	FL0	I	Fixed level (low), connect to ground via ext. pull-down		
	SPI	FL0	I	Fixed level (low), connect to ground via ext. pull-down		
207		GND		Ground		
208		VDD		+3.3 V power supply		

Legend:

I	Input pin
O	Output pin
IO	Bidirectional pin
Ood	Output pin with open drain
IOpu	Bidirectional pin with internal pull-up resistor of app. 100 kΩ to VDD
NC	Not connected
FL0	Fixed level (low), must be connected to ground via external pull-down (e.g. 1 MΩ)
VDD	Fixed level (high), must be connected to power supply via external external pull-up (e.g. 1 MΩ)

Unused input pins should be tied to ground. Unused I/O pins should be tied via a 1 MΩ resistor to ground.

Important !

FL0 and VDD pins might be driven as chip output during power-on. To prevent a short circuit these pins must either be connected via a resistor (e.g. 1 MΩ) to ground resp. power supply or they can directly be tied to ground resp. power supply, if RESET is always active during power-on.

Chapter 2

Universal external bus interface

(Overview tables of the HFC-E1 bus interface pins can be found at the beginning of the sections 2.2 . . . 2.6.)

Table 2.1: *Overview of the HFC-E1 bus interface registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x00	R_CIRM	86	0x15	R_RAM_USE	90
0x01	R_CTRL	87	0x16	R_CHIP_ID	91
0x08	R_RAM_ADDR0	87	0x1C	R_STATUS	242
0x09	R_RAM_ADDR1	88	0x1F	R_CHIP_RV	91
0x0A	R_RAM_ADDR2	88			
0x0C	R_RAM_MISC	89			

The HFC-E1 has an integrated universal external bus interface which can be configured as PCI, ISA PnP, PCMCIA, microprocessor interface and SPI. Table 2.2 shows how to select the bus mode via the two pins MODE0 and MODE1.

Table 2.2: *Access types*

Bus mode	MODE1	MODE0	8 bit	16 bit	32 bit	Page
PCI	0	0				47
PCI memory mapped mode			✓	✓	✓	
PCI I/O mapped mode			✓	✓	✓	
ISA Plug and Play	1	0	✓	✓	✗	54
PCMCIA	1	1	✓	✓	✗	60
Processor Interface	0	1				63
Mode 2: Motorola			✓	✓	✗	
Mode 3: Intel, non-multiplexed			✓	✓	✗	
Mode 4: Intel, multiplexed			✓	✓	✓	
SPI *	0	1	✓	✗	✗	83

(*: SPI mode is selected by using processor interface mode and connecting pin 200 to SPI clock.)

The external bus interface supports 8 bit, 16 bit and 32 bit accesses. The available access types depend on the selected bus mode like shown in Table 2.2.

The sections 2.2 to 2.6 explain how to use the HFC-E1 in the different bus modes.

2.1 Common features of all interface modes

Table 2.3: *Overview of common bus interface pins* [1]

Number	Name	Description
99	MODE0	Interface Mode pin 0
100	MODE1	Interface Mode pin 1
102	EE_SCL/EN	EEPROM clock / EEPROM enable
103	EE_SDA	EEPROM data I/O

2.1.1 EEPROM programming

The ISA PnP and PCMCIA interfaces require an external EEPROM. For the PCI bus and the processor interface mode, this EEPROM is optional. The EEPROM programming specification is only available on special request from Cologne Chip to avoid destruction of configuration information by not authorized programs or software viruses.

The EEPROM is used to store the configuration data for PCMCIA, PCI or ISA PnP. After a reset (hardware reset or EEPROM load with V_RLD_EPR = 1 of the register R_CIRM) the HFC-E1 copies a constant number of bytes from the EEPROM to the SRAM. The bytes which are not used by the configuration data can be filled with vendor defined data. This data (and the configuration data as well) can be read by RAM accesses to the HFC-E1. Tables 2.4 and 2.5 show how many bytes are copied in the different modes and which start address is used for different SRAM sizes.

Table 2.4: *EEPROM load size*

Mode	Number of bytes copied
ISA PnP mode	512
PCMCIA mode	512
PCI mode	128
parallel processor mode	512

Table 2.5: *SRAM start address*

SRAM size	Start address in SRAM
32k x 8	0x1A00
128k x 8	0x2A00
512k x 8	0x2A00

2.1.2 EEPROM circuitry

Figure 2.1 shows the connection of an EEPROM (e.g. 24C04 type) to the HFC-E1 pins EE_SCL/EN and EE_SDA.

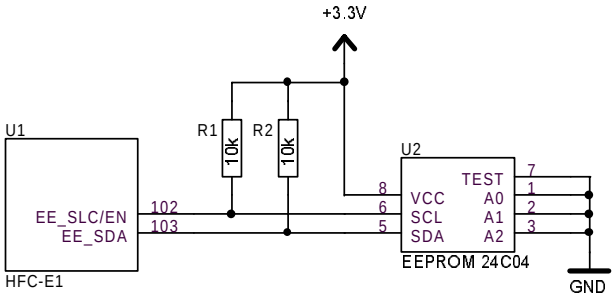

Figure 2.1: *EEPROM connection circuitry*

If no EEPROM is used, pin EE_SCL/EN must be connected to ground while EE_SDA must remain open as shown in Figure 2.2.

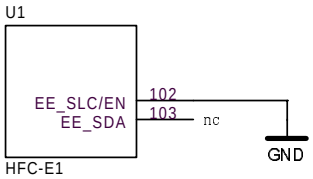

Figure 2.2: *EE_SCL/EN and EE_SDA connection without EEPROM*

2.1.3 Register access

In PCI I/O mapped mode, ISA PnP, PCMCIA mode and SPI mode all registers are selected by writing the register address into the *Control Internal Pointer* (CIP) register. This is done by writing the CIP on the higher I/O addresses (AD2, SA2, A2, $A/\bar{D} = 1$). The CIP register can also be read with AD2, SA2, A2, $A/\bar{D} = 1$.

All consecutive read or write data accesses (AD2, SA2, A2, $A/\bar{D} = 0$) are done with the selected register until the CIP register is changed.

In processor interface mode all internal registers can be directly accessed. The registers are selected by A0 . . . A7.

In PCI mode internal A0 and A1 are generated from the byte enable lines.

2.1.4 RAM access

The SRAM of the HFC-E1 can be accessed by the host. For doing so the desired RAM address has to be written in the R_RAM_ADDR0 . . . R_RAM_ADDR2 registers first. Then data can be read / written by reading / writing the register R_RAM_DATA. An automatic increment function can be set in the register R_RAM_ADDR2.

[1] See sections 2.2 to 2.6 for overview tables of the interface specific pins.

2.2 PCI interface

Table 2.6: *Overview of the PCI interface pins*

Number	Name	Description
203 . . . 206, 1 . . . 4	AD31 . . . AD24	Address / Data byte 3
8 . . . 17	AD23 . . . AD16	Address / Data byte 2
31 . . . 39	AD15 . . . AD8	Address / Data byte 1
43 . . . 51	AD7 . . . AD0	Address / Data byte 0
6, 18, 30, 40	C/BE3# . . . C/BE0#	Bus command and Byte Enable 3 . . . 0
7	IDSEL	Initialisation Device Select
20	FRAME#	Cycle Frame
21	IRDY#	Initiator Ready
22	TRDY#	Target Ready
23	DEVSEL#	Device Select
24	STOP#	Stop
25	PERR#	Parity Error
26	SERR#	System Error
27	PAR	Parity Bit
195	PME_IN	Power Management Event Input
196	PME	Power Management Event output
197	INTA#	Interrupt request
198	RST#	Reset low active
200	PCICLK	PCI Clock Input

The PCI mode is selected by MODE0 $= 0$ and MODE1 $= 0$. Only PCI target mode accesses are supported by the HFC-E1.

5 V PCI bus signaling environment is supported with 3.3 V supply voltage of the HFC-E1. Never connect the power supply of the HFC-E1 to 5 V!

The PCI interface is build according to the PCI Specification 2.2.

2.2.1 PCI command types

Table 2.7 shows the supported PCI commands of the HFC-E1.

Memory Read Line and Memory Read Multiple commands are aliased to Memory Read. Memory Write and Invalidate is aliased to Memory Write.

Byte 3	Byte 2	Byte 1	Byte 0	Hex Address
Device ID		Vendor ID		00h
Status Register		Command Register		04h
Class Code			Revision ID	08h
BIST	Header Type	Latency Timer	Cache Line Size	0Ch
I/O Base Address				10h
Memory Base Address				14h
Base Address 2				18h
Base Address 3				1Ch
Base Address 4				20h
Base Address 5				24h
CardBus CIS Pointer				28h
Subsystem ID		Subsystem Vendor ID		2Ch
Expansion ROM Base Address				30h
Reserved			Cap_Ptr	34h
Reserved				38h
Max_Lat	Min_Gnt	Interrupt Pin	Interrupt Line	3Ch
PMC		Next Item Ptr	Cap_ID	40h
Data	PMCSR BSE	PMCSR		44h

- Register is implemented, value can be set by EEPROM
- Register is implemented
- Register is not implemented and returns all 0's when read

Figure 2.3: *PCI configuration registers*

Table 2.7: *PCI command types*

C/BE3#	C/BE2#	C/BE1#	C/BE0#	nibble value	Command type
0	0	1	0	2	I/O Read
0	1	1	0	6	Memory Read
1	1	0	0	0xC	Memory Read Multiple
1	1	1	0	0xE	Memory Read Line
1	0	1	0	0xA	Configuration Read
0	0	1	1	3	I/O Write
0	1	1	1	7	Memory Write
1	1	1	1	0xF	Memory Write and Invalidate
1	0	1	1	0xB	Configuration Write

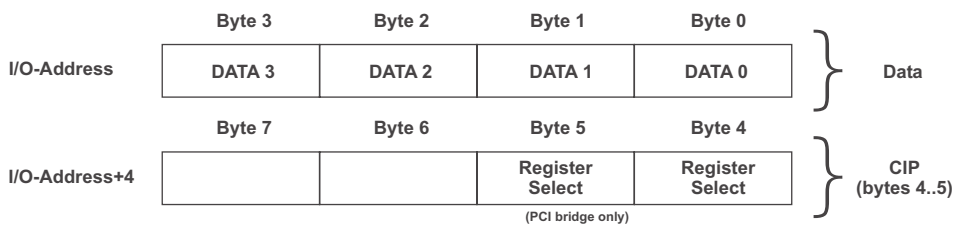

Figure 2.4: *PCI access in PCI I/O mapped mode*

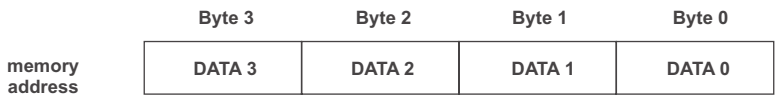

Figure 2.5: *PCI access in PCI memory mapped mode*

2.2.2 PCI access description

Two modes exist for register access:

1. If HFC-E1 is used in *PCI memory mapped mode* all registers can directly be accessed by adding their CIP address to the configured Memory Base Address.
2. In *PCI I/O mapped mode* HFC-E1 only occupies 8 bytes in the I/O address space.

In PCI I/O mapped mode all registers are selected by writing the register address into the *Control Internal Pointer* (CIP) register. This is done by writing the HFC-E1 on the higher I/O addresses (AD2 $= 1$). If the auxiliary interface is used (see Chapter 11) the CIP write access must have a width of 16 bit.

All consecutive read or write data accesses (AD2 $= 0$) use the selected register until the CIP register is changed.

2.2.3 PCI configuration registers

The PCI configuration space is defined by the configuration register set which is illustrated in Figure 2.3. In the configuration address space 0x00 ... 0x47 the PCI configuration register values are either

- set by the HFC-E1 default settings of the configuration values or
- they can be written to upper configuration registers or
- they are read from the external EEPROM.

The external EEPROM is optional. If no EEPROM is available, the pin EE_SCL/EN has to be connected to GND and the pin EE_SDA has to be left open. Without EEPROM the PCI configuration registers will be loaded with the default values shown in Table 2.8.

All configuration registers which can be set by the EEPROM can also be written by configuration write accesses to the upper addresses of the configuration register space (from 0xC0 upwards). The addresses for configuration writes are shown in Table 2.8. Unimplemented registers return all '0's when read.

Table 2.8: PCI configuration registers

Register Name	Address	Width	Default Value	Remarks
Vendor ID	0x00	Word	0x1397	Value can be set by EEPROM. Base address for configuration write is 0xC0.
Device ID	0x02	Word	0x30B1	Value can be set by EEPROM. Base address for configuration write is 0xC0.
Command Register	0x04	Word	0x0000	Bits / Function 0 Enables / disables I/O space accesses 1 Enables / disables memory space accesses 5..2 fixed to 0 6 PERR# enable / disable 7 fixed to '0' 8 SERR# enable / disable 15..9 fixed to 0

(continued on next page)

Table 2.8: PCI configuration registers

(continued from previous page)

Register Name	Address	Width	Default Value	Remarks
Status Register	0x06	Word	0x0210	Bits 0 … 7 can be set by EEPROM. Base address for configuration write is 0xC4. Bits — Function 3..0 — reserved 4 — '1' = *Capabilities List* exists, fixed to '1' 5 — '0' = 33 MHz capable (default) '1' = 66 MHz capable 6 — reserved 7 — '0' = fast Back-to-Back not capable (default) '1' = fast Back-to-Back capable 8 — fixed to '0' 10..9 — fixed to '01': timing of DEVSEL# is medium 11 — fixed to '0' 13..12 — fixed to '00' 14 — system error (address parity error) 15 — any detected data or system parity error
Revision ID	0x08	Byte	0x01	HFC-E1 Revision 01
Class Code	0x09	3 Bytes	0x020400	Class code for 'ISDN controller'.Value can be set by EEPROM. Base address for configuration write is 0xC8.
Header Type	0x0E	Byte	0x00	Header type 0
BIST	0x0F	Byte	0x00	No build in self test supported.
I/O Base Address	0x10	DWord		Bits 3 … 31 are r/w by configuration accesses. 8 Byte address space is used.
Memory Base Address	0x14	DWord		Bits 12 … 31 are r/w by configuration accesses. 4 kByte address space is used.
Subsystem Vendor ID	0x2C	Word	0x1397	Value can be set by EEPROM. Base address for configuration write is 0xEC.
Subsystem ID	0x2E	Word	0x30B1	Value can be set by EEPROM. Base address for configuration write is 0xEC.
Cap_Ptr	0x34	Byte	0x40	Offset to Power Management register block.
Interrupt Line	0x3C	Byte	0xFF	This register must be configured by configuration write.
Interrupt Pin	0x3D	Byte	0x01	INTA# supported
Cap_ID	0x40	Byte	0x01	Capability ID. 0x01 identifies the linked list item as PCI Power Management registers.
Next Item Ptr	0x41	Byte	0x00	There are no next items in the linked list.

(continued on next page)

Table 2.8: PCI configuration registers

(continued from previous page)

Register Name	Address	Width	Default Value	Remarks
PMC *1	0x42	Word	0x7E22	Power Management Capabilities, see also 'PCI Bus Power Management Interface Specification Rev. 1.1'.This register's value can be set by EEPROM. Base address for configuration write is 0xE0. Bits — Function 0..2: '010' = PCI Power Management Spec. Version 1.1. 3: '0' = The HFC-E1 does not require PCI-clock to generate PME. 4: Fixed to '0'. 5: '1' = Device specific initialisation is required. 8..6: '000' = No D3_cold support *1. 9: '1' = Supports D1 Power Management State *2. 10: '1' = Supports D2 Power Management State *2. 15..11: PME can be asserted from D0, D1, D2 and D3_hot.
PMCSR	0x44	Word	0x0000	Power Management Control/Status Bits — Function 1..0: **PowerState**: These bits are used both to determine the current power state of a function and to set the function into a new power state *2. '00': D0 '01': D1 '10': D2 '11': D3_hot 7..2: fixed to '0' 8: **PME_En**: '1' enables the function to assert PME. '0' = PME assertion is disabled. 14..9: fixed to 0 15: **PME_Status**: This bit is set when the function would normally assert the PME signal independent of the state of the **PME_En** bit. Writing a '1' to this bit will clear it and cause the function to stop asserting a PME (if enabled). Writing a '0' has no effect.

*1: D3_cold support is implemented but must be set in the EEPROM configuration data.

*2: Changing the power management does not change the power dissipation. It is only implemented for PCI specification compatibility.

2.2.4 PCI connection circuitry

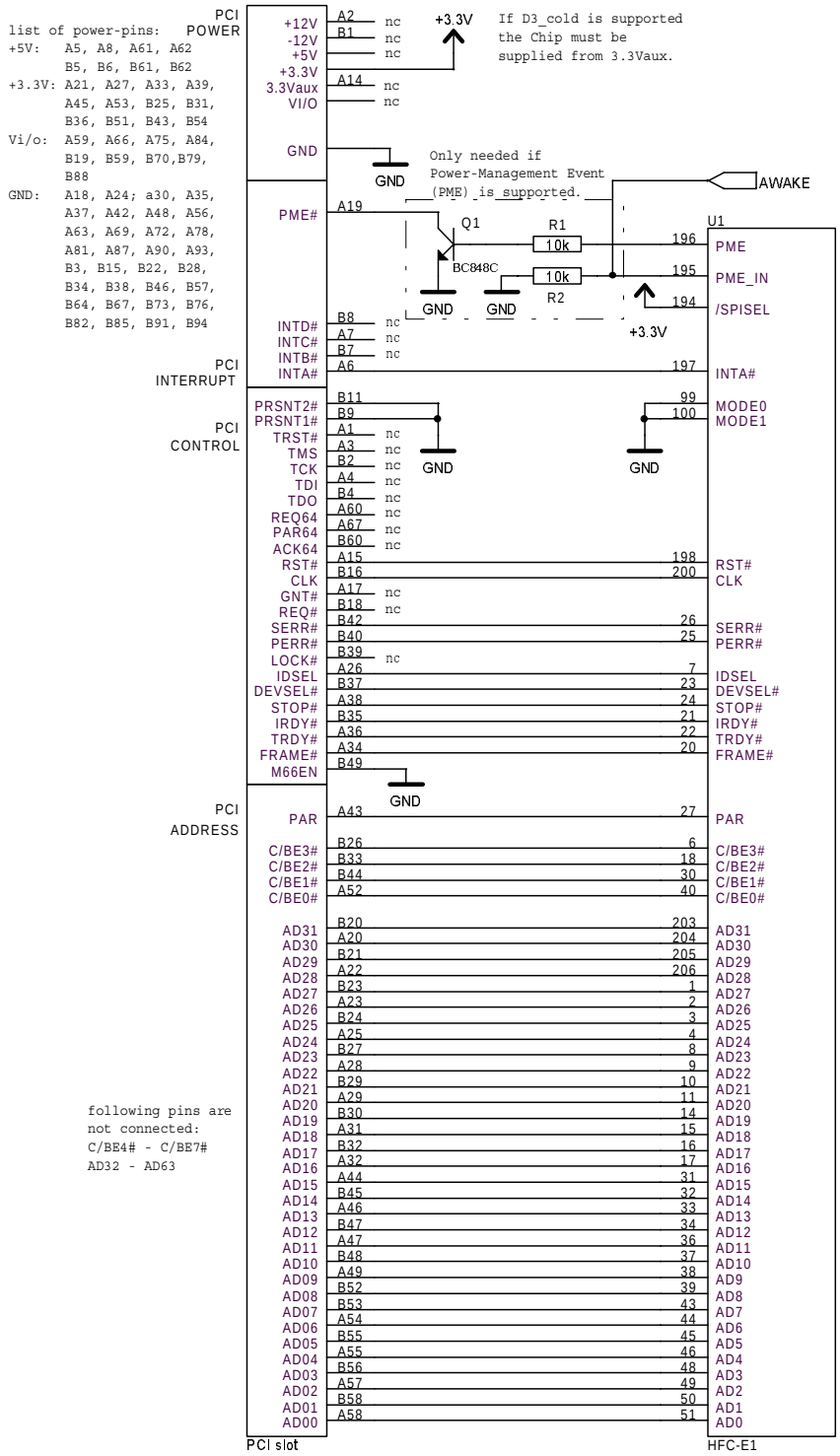

Figure 2.6: *PCI connection circuitry*

2.3 ISA Plug and Play interface

Table 2.9: *Overview of the ISA PnP interface pins*

Number	Name	Description
203 . . . 206,1 . . . 4	SA15 . . . SA8	Address byte 1
8 . . . 17	SA7 . . . SA0	Address byte 0
31 . . . 39	SD15 . . . SD8	Data byte 1
43 . . . 51	SD7 . . . SD0	Data byte 0
106 . . . 112	IRQ6 . . . IRQ0	ISA Interrupt Request 6 . . . 0
18	/IOIS16	16 bit access enable
20	/AEN	Address Enable
21	/IOR	Read Enable
22	/IOW	Write Enable
25	/BUSDIR	Bus Direction
30	/SBHE	High byte enable
198	RESET	Reset high active

ISA Plug and Play mode is selected by MODE0 $= 0$ and MODE1 $= 1$. The HFC-E1 needs eight consecutive addresses in the I/O map of a PC for operation. Usually also one out of several ISA IRQ lines is used. Section 2.3.1 describes how to configure the interrupt lines of the HFC-E1.

The port address is selected by the lines SA0 . . . SA15. The address with SA2 = '1' is used for register selection via the CIP (Control Internal Pointer) and the address with SA2 = '0' is used for data read / write like shown in Table 2.10. The bits SA3 . . . SA15 are decoded by the address decoder to match the PnP configuration address.

Table 2.10: *ISA address decoding (X = don't care)*

SA2	/IOR	/IOW	/AEN	Operation
X	X	X	1	no access
X	1	1	X	no access
0	0	1	0	read data
0	1	0	0	write data
1	0	1	0	read CIP
1	1	0	0	write CIP

The HFC-E1 has no memory or DMA access to any component on the ISA PC bus. Because of its characteristic power drive no external driver for the ISA PC bus data lines is needed.

If necessary (e.g. due to an old ISA specification which requires 24 mA output current) an external bus driver can be added. In this case the output signal /BUSDIR determines the driver direction.

/BUSDIR = 0 means that the HFC-E1 is read and data is driven to the external bus.

/BUSDIR = 1 means that data is driven (written) into the HFC-E1.

2.3.1 IRQ assignment

The IRQ lines are tristated after a hardware reset.

The IRQ assigned by the PnP BIOS can be read from the bitmap V_PNP_IRQ of the register R_CHIP_ID. The bitmap V_IRQ_SEL of the register R_CIRM has to be set according to the IRQ wiring between HFC-E1 and the ISA slot on the PCB. Thus the IRQ number assigned by the PnP BIOS is connected to the right IRQ line on the ISA bus.

2.3.2 ISA Plug and Play registers

Table 2.11: ISA Plug and Play registers

Card level control register address	Read / write Mode	Accessable in state	Description
0x00	w	Isolation state, Config state *1	**Set read data port address register**. Bits 0 . . . 7 become bits 2 . . . 9 of the port's I/O address. Bits 10 and 11 are hardwired to '00' and bits 0 and 1 are hardwired to '11'.
0x01	r	Isolation state	**Serial isolation register**. Used to read the serial identifier during the card isolation process.
0x02	w	Sleep state, Isolation state, Config state	**Configuration control register**. Bits — Function 0 — **Reset Bit**. The value '1' resets all of the card's configuration registers to their default state. The CSN is not affected. 1 — **Return to wait for key state**. When set to one, all cards return to wait for key state. Their CSNs and configuration registers are not affected. This command is issued after all cards have been configured and activated. 2 — **Reset CSN to zero**. When set to one, all cards reset their CSN to zero. All bits are automatically cleared by the hardware. 7..3 — Reserved, must be zero

(continued on next page)

Table 2.11: ISA Plug and Play registers

(continued from previous page)

Card level control register address	Read / write Mode	Accessable in state	Description
0x03	w	Sleep state, Isolation state, Config state	**Wake command register.** Writing a CSN to this register has the following effects: • If the value written is 0x00, all cards in the sleep state with a CSN = 0x00 go to the isolation state. All cards in configure state (CSN not 0x00) go to the sleep state. • If the value written is not 0x00, all cards in the sleep state with a matching CSN go to the configure state. All cards in the isolation state go to the sleep state. Every write to a card's wake command register with a match on its CSN causes the pointer to the serial identifier / resource data to be reset to the first byte of the serial identifier.
0x04	r	Config state	**Resource data register.** This register is used to read the device's recource data. Each time when a read is performed from this register a byte of the resource data is returned and the resource data pointer is incremented. Prior to reading each byte, the programmer must read from the status register to determine if the next byte is available for reading from the resource data register. The card's serial identifier and checksum must be read prior to accessing the resource requirement list via this register.
0x05	r	Config state	**Status register.** Prior to reading the next byte of the device's resource data, the programmer must read from this register and check bit 0 for a '1'. This is the resource data byte available bit. Bits 1 . . . 7 are reserved.
0x06	r/w	Isolation state *2 Config state	**Card select number (CSN) register.** The configuration software uses the CSN register to assign a unique ID to the card. The CSN is then used to wake up the card's configuration logic whenever the configuration program must access its configuration registers.
0x07	r	Config state	**Logical device number register.** The number in this register points to the logical device the next commands will operate on. The HFC-E1 only supports one logical device. This register is hardwired to all zeros.

(continued on next page)

Table 2.11: ISA Plug and Play registers

(continued from previous page)

Card level control register address	Read / write Mode	Accessable in state	Description
0x30	r/w	Config state	**Activate register.** Setting bit 0 to '1' activates the card on the ISA bus. When cleared, the card cannot respond to any ISA bus transactions (other than accesses to its Plug and Play configuration ports). Reset clears bit 0. Bits 1 . . . 7 are reserved and return zeros when read. The HFC-E1 only supports one logical device, so it is not necessary to write the logical device number into the card's logical device number register prior to writing to this register.
0x31	r/w	Config state	**I/O range check register.** Bits — Function 0 — When set, the logical device returns 0x55 in response to any read from the logical device's assigned I/O space. When cleared, 0xAA is returned. 1 — When set to one, enables I/O range checking and disables it when cleared to zero. When enabled, bit 0 is used to select a pattern for the logical device to return. **This bit is only valid if the logical device is deactivated (see *Activate register*)**. 7..2 — Reserved, return zero when read
0x60	r/w	Config state	**I/O decoder 0 base address upper byte.** I/O port base address bits 8 . . . 15.
0x61	r/w	Config state	**I/O decoder 0 base address lower byte.** I/O port base address bits 0 . . . 7.
0x70	r/w	Config state	**IRQ select configuration register 0.** Bits 0 . . . 3 specify the selected IRQ number. Bits 4 . . . 7 are reserved.
0x71	r/w	Config state	**IRQ type configuration register 0.** Bits 0 and 1 are ignored. Bits 2 . . . 7 are reserved.
0x74	r	Config state	**DMA configuration register 0.** Bits — Function 2..0 — Select which DMA channel (0 . . . 7) is used for DMA 0. DMA channel 4, the cascade channel, indicates no DMA channel is active. 7..3 — Reserved. Because no DMA is used this register is hardwired to 0x04.

(continued on next page)

Table 2.11: ISA Plug and Play registers

(continued from previous page)

Card level control register address	Read / write Mode	Accessable in state	Description
0x75	r	Config state	**DMA configuration register 1.** Bits — Function 2..0 — Select which DMA channel (0 … 7) is used for DMA 1. DMA channel 4, the cascade channel, indicates no DMA channel is active. 7..3 — Reserved. Because no DMA is used this register is hardwired to 0x04.

*1: This is an extension to the Plug and Play Specification.

*2: Only when the isolation process is finished. The last card remains in isolation state until a CSN is assigned.

Important !

All ISA registers not implemented return 0x00 when read except the DMA configuration registers 0x74 and 0x75. These two registers return 0x04 when read. This means no DMA channel has been selected.

2.3.3 ISA connection circuitry

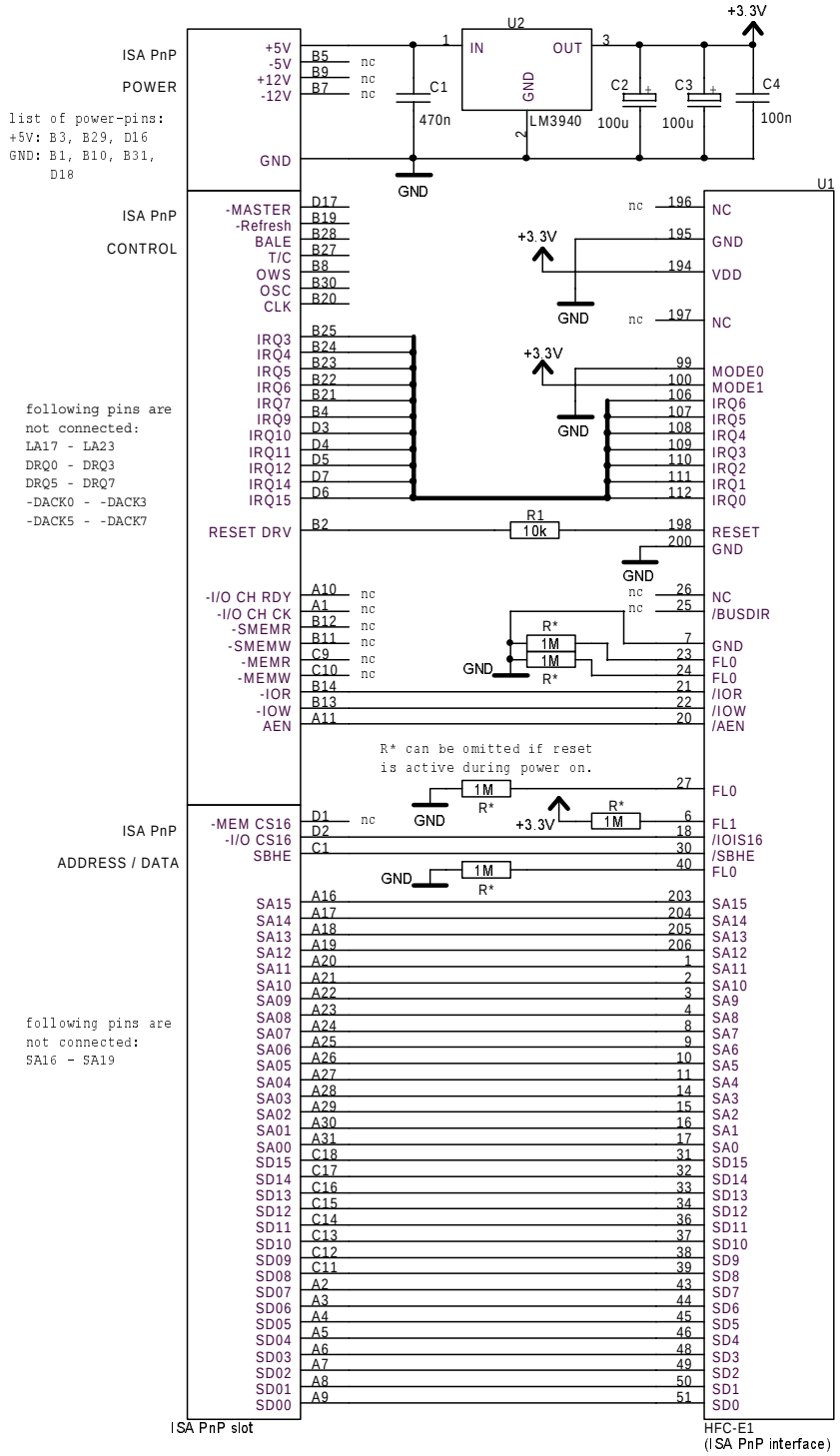

Figure 2.7: *ISA PnP circuitry*

2.4 PCMCIA interface

Table 2.12: *Overview of the PCMCIA interface pins*

Number	Name	Description
203 . . . 206, 1 . . . 4	A15 . . . A8	Address byte 1
8 . . . 17	A7 . . . A0	Address byte 0
31 . . . 39	D15 . . . D8	Data byte 1
43 . . . 51	D7 . . . D0	Data byte 0
7	REG#	PCMCIA Register and Attr. Mem. Select
18	IOIS16#	16 bit access enable
21	IORD#	Read Enable
22	IOWR#	Write Enable
23	OE#	PCMCIA Output Enable for Attr. Mem. Read
24	WE#	PCMCIA Write Enable for Conf. Reg. Write
25	INPACK#	Read access
30	CE2#	High byte enable
40	CE1#	Low byte enable
197	IREQ#	Interrupt request
198	RESET	Reset high active

The PCMCIA mode is selected by MODE0 $= 1$ and MODE1 $= 1$. The HFC-E1 occupies eight consecutive addresses in the I/O map.

The base I/O address must be 8 byte aligned. The lines A3 . . . A15 are don't care for I/O accesses.

The address with A2 $= 1$ is used for register selection via CIP. The address with A2 $= 0$ is used for data read / write.

2.4.1 Attribute memory

After a hardware reset the card's information structure (CIS) is copied from the EEPROM to the SRAM, starting with the address shown in Table 2.5. The CIS is located on even numbered addresses from 0 to 0x3FE in the attribute memory space. The CIS occupies 512 byte. To avoid accesses in this copy phase the signal IREQ# of the HFC-E1 is active. This is interpreted as 'wait' by the PCMCIA host controller after card insertion.

2.4.2 PCMCIA registers

Table 2.13: *PCMCIA registers*

Register Name	Address *	Width	Remarks			
Configuration Option Register (COR)	0x400	Byte	**Bit**	**Name**	**Reset value**	**Function**
			5..0	Configuration Index	0x00	Bit 0 must be set to '1' to enable accesses to the HFC-E1.
			6	LevIREQ	1	This bit is not implemented and returns always '1' when read to indicate usage of level mode interrupts.
			7	SRESET		SRESET card. Setting this bit to '1' places the card in the reset state. This bit must be cleared to zero for normal operation.
Card Configuration and Status Register (CSR)	0x402	Byte	**Bit**	**Name**	**Reset value**	**Function**
			0	Rsvd	0	
			1	Intr	0	Internal state of interrupt request (IREQ#).
			2	PwrDwn	0	Unimplemented, returns '0' when read.
			3	Audio	0	Unimplemented, returns '0' when read.
			4	Rsvd	0	Unimplemented, returns '0' when read.
			5	IOis8	0	Returns '0' when read to indicate an 16 bit data path.
			6	SigChg	0	Unimplemented, returns '0' when read.
			7	Changed	0	Unimplemented, returns '0' when read.

(*: Register address in attribute memory)

2.4.3 PCMCIA connection circuitry

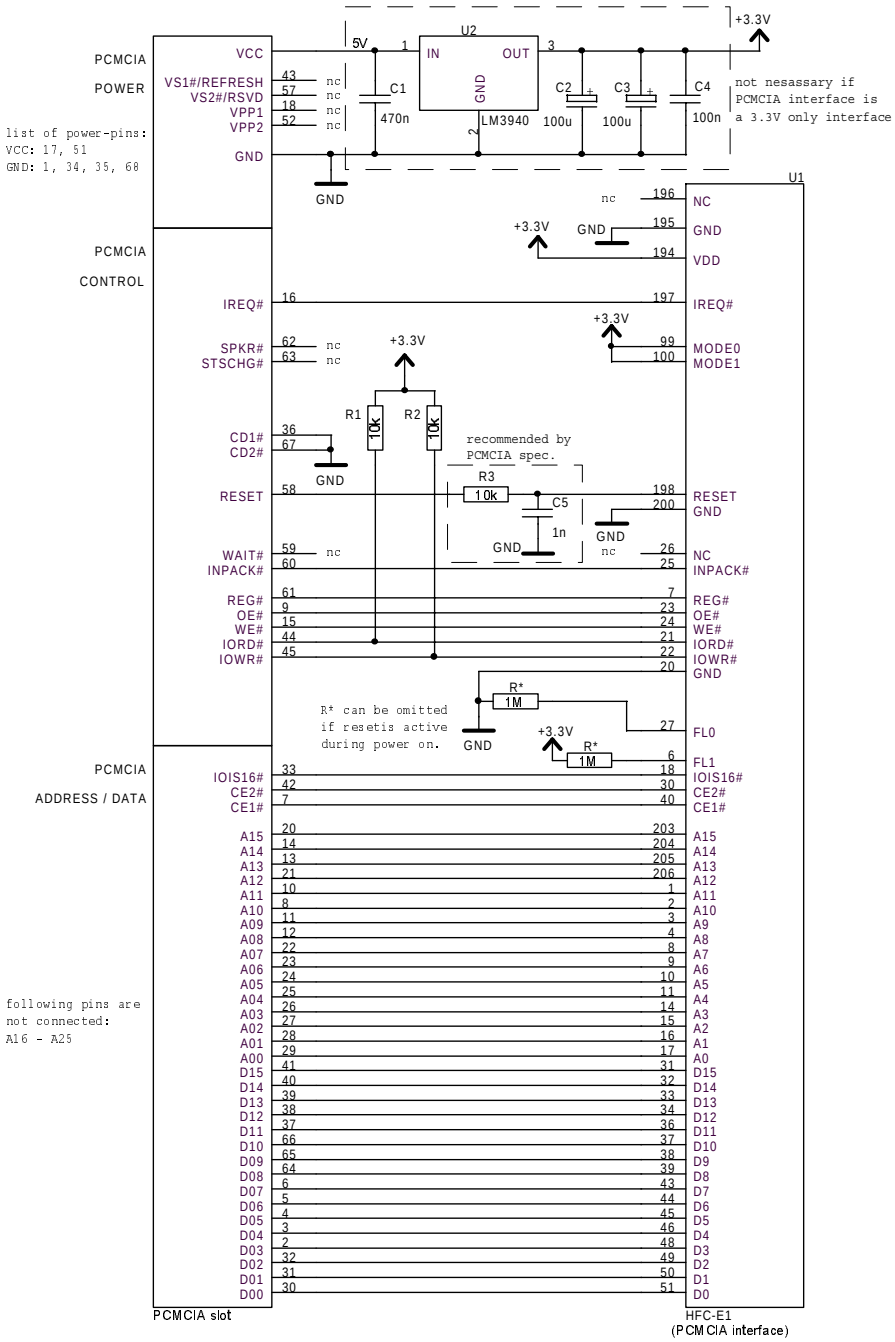

Figure 2.8: *PCMCIA circuitry*

2.5 Parallel processor interface

Table 2.14: *Overview of the parallel processor interface pins in mode 2 and 3*

Number	Name	Description
8 . . . 17	A7 . . . A0	Address byte
43 . . . 51	D7 . . . D0	Data byte 0
31 . . . 39	D15 . . . D8	Data byte 1
6, 18, 30, 40	/BE3 . . . /BE0	Byte Enable 3 . . . 0
20	/CS	Chip Select
21	/IOR	Read Enable
22	/IOW	Write Enable
23	/WD	Watch Dog Output
24	ALE	Address Latch Enable
25	/BUSDIR	Bus Direction
197	/INT	Interrupt request
198	RESET	Reset high active

Table 2.15: *Overview of the processor interface pins in mode 4*

Number	Name	Description
43 . . . 51	AD7 . . . AD0	Address / Data byte 0
31 . . . 39	AD15 . . . AD8	Address / Data byte 1
8 . . . 17	AD23 . . . AD16	Address / Data byte 2
203 . . . 206, 1 . . . 4	AD31 . . . AD24	Address / Data byte 3
6, 18, 30, 40	/BE3 . . . /BE0	Byte Enable 3 . . . 0
20	/CS	Chip Select
21	/IOR	Read Enable
22	/IOW	Write Enable
23	/WD	Watch Dog Output
24	ALE	Address Latch Enable
25	/BUSDIR	Bus Direction
197	/INT	Interrupt request
198	RESET	Reset high active

The processor interface mode is selected by MODE0 = 1 and MODE1 = 0. Then 256 I/O addresses (A0 ... A7) are used for addressing the internal registers of the HFC-E1 directly by their address.

In processor interface mode some user data can be stored in the EEPROM (see Section 2.1.1 for details).

2.5.1 Parallel processor interface modes

The HFC-E1 has 3 different parallel processor interface modes. Due to name compatibility with other chips of the HFC series the processor interface modes are numbered 2 ... 4 like shown in Table 2.16.

Table 2.16: *Pins and signal names of the HFC-E1 processor interface modes*

HFC-E1 pins		Signal names		
Number	Name	Mode 2 (Motorola) Non-multiplexed	Mode 3 (Intel) Non-multiplexed	Mode 4 (Intel) Multiplexed
20	/CS	/CS	/CS	/CS
21	/IOR	/DS	/RD	/RD
22	/IOW	R/W	/WR	/WR
24	ALE	'1'	'0'	ALE

Processor interface modes 2 and 3 use separate lines for address and data. These two modes are selected by ALE. This pin must have a fixed level and should be directly connected to ground or power supply. Mode 4 has multiplexed address / data lines. The address is latched from lines D7 ... D0 with the falling edge of ALE.

The processor interface mode is determined during hardware reset time (pin RESET). For modes 2 and 3 the ALE pin must have the appropriate level. Mode 4 is selected after reset with the first rising edge of ALE. The HFC-E1 then switches permanently from mode 2 or mode 3 into mode 4. The HFC-E1 cannot switch to mode 4 until end of reset time. Rising and falling edges of ALE are ignored during reset time.

ALE must be stable after reset except in processor interface mode 4.

2.5.2 Signal and timing characteristics

Table 2.17 shows the interface signal levels for the different processor interface modes. Timing characteristics are shown in Figures 2.9 to 2.12 for mode 2 and mode 3. Figures 2.13 to 2.18 show mode 4 timing characteristics. Please see Table 2.18 for a quick timing and symbol list finding.

In processor interface mode 4 it is possible to access byte, word or double word on the lines AD31 ... AD0. Due to the multiplexed lines the PCI pin names are used in this case. In

Table 2.17: *Overview of read and write accesses in processor interface mode (X = don't care)*

/CS	/IOR (/DS, /RD)	/IOW (R/W, /WR)	ALE	Operation	Processor interface mode
1	X	X	X	no access	all
X	1	1	X	no access	all
0	0	1	1	read data	mode 2
0	0	0	1	write data	mode 2
0	0	1	0	read data	mode 3
0	1	0	0	write data	mode 3
0	0	1	0 *	read data	mode 4
0	1	0	0 *	write data	mode 4

(*: 1-pulse latches register address)

Table 2.18: *Timing diagrams of the parallel processor interface*

Mode	Processor	Access type		Timing Figure	Timing on page	Timing values table	Timing values on page
2 & 3	8 bit	8 bit	read	2.9	66	2.20	70
2 & 3	8 bit	8 bit	write	2.10	68	2.21	72
2 & 3	16 bit	16 bit & 8 bit	read	2.11	69	2.20	70
2 & 3	16 bit	16 bit & 8 bit	write	2.12	71	2.21	72
4	8 bit	8 bit	read	2.13	73	2.23	78
4	8 bit	8 bit	write	2.14	74	2.24	80
4	16 bit	16 bit	read	2.15	75	2.23	78
4	16 bit	16 bit	write	2.16	76	2.24	80
4	32 bit	32 bit	read	2.17	77	2.23	78
4	32 bit	32 bit	write	2.18	79	2.24	80

processor interface mode 2 and mode 3 the pins AD31 . . . AD24 are not available.

Unused byte enable pins should be connected to power supply via pull-up resistors. In mode 4 unused bus lines AD[31..] should be connected to ground via pull-down resistors to avoid floating inputs.

Important !

/BE2 and /BE3 must always be '1' in mode 2 and mode 3.

2.5.2.1 8 bit processors in mode 2 (Motorola) and mode 3 (Intel)

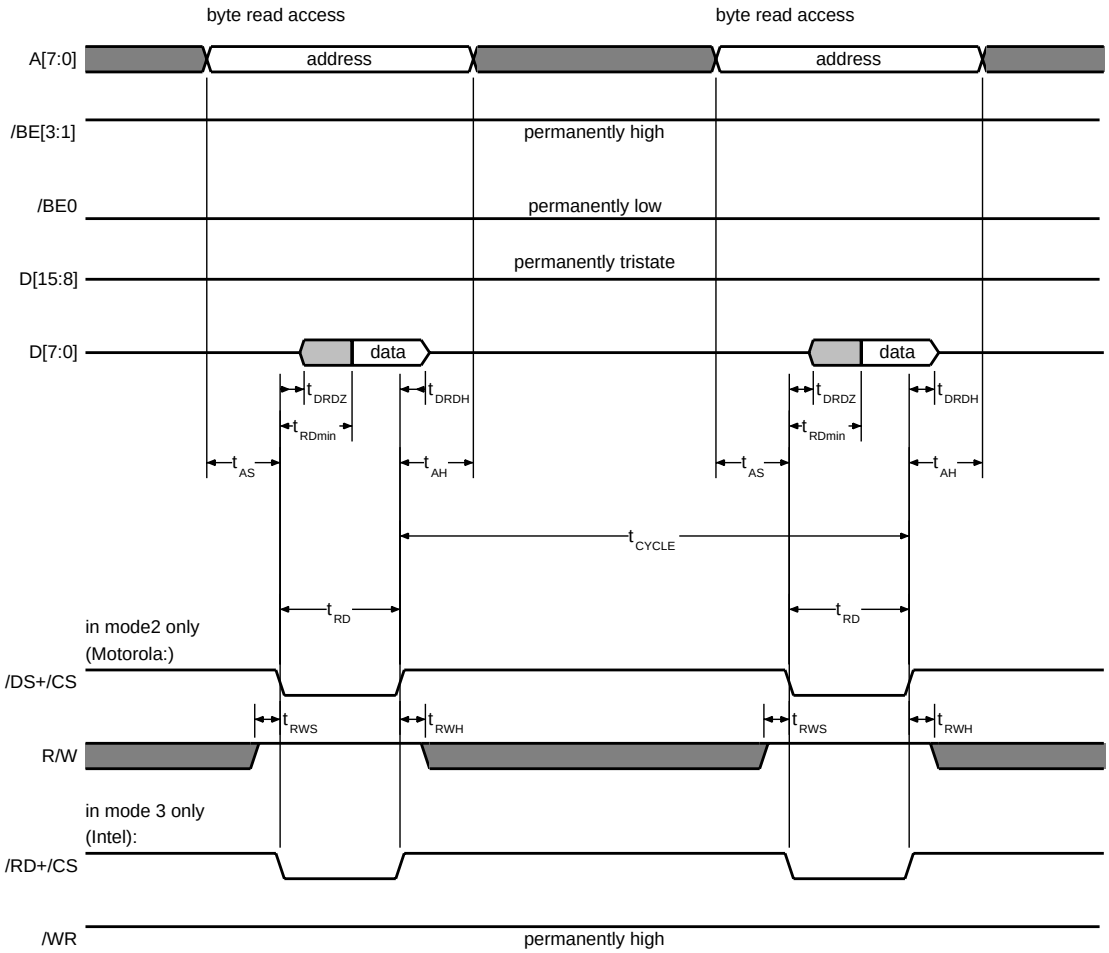

Figure 2.9: *Read access from 8 bit processors in mode 2 (Motorola) and mode 3 (Intel)*

8 bit processors read data like shown in Figure 2.9. Timing values are listed in Table 2.20.

/BE3 … /BE1 must always be '1'. /BE0 can be fixed to '0' or must be low during access to switch the data bus D7 … D0 from tristate into data driven state.

Data can be read in mode 2 (Motorola) with [2]

$$/\mathrm{BE0} = \text{'0'} \quad \text{and} \quad (/\mathrm{DS} + /\mathrm{CS}) = \text{'0'} \quad \text{and} \quad \mathrm{R/W} = \text{'1'} \ .$$

In mode 3 (Intel, non-multiplexed) the states

$$/\mathrm{BE0} = \text{'0'} \quad \text{and} \quad (/\mathrm{RD} + /\mathrm{CS}) = \text{'0'} \quad \text{and} \quad /\mathrm{WR} = \text{'1'}$$

must be fulfilled to drive data out. The data bus is stable after t_{RDmin} and returns into tristate after t_{DRDH}.

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

Address and /BE0 (if not fixed to low) require a setup time t_{AS} which starts when all address and byte enable signals are valid. The hold time of these lines is t_{AH}.

Short read method

In some applications it may be difficult to implement a long read access ($t_{RD} \geq 5 \cdot t_{CLKI}$) for only some registers (here called *target register*).

For this reason there is an alternative method with two register read accesses with $t_{RD} \geq 20\,\mathrm{ns}$ each:

1. The read access to the target register initiates a data transmission from the RAM to the target register. This job is always done correctly with long and short t_{RD}, but after a short t_{RD} the data is not yet 'arrived' at the target register. Thus the data which is read with a short t_{RD} must be ignored ...
2. ...but the data byte is already internally buffered and can be read from the register R_INT_DATA. This second register read access can also be executed with a short $t_{RD} \geq 20\,\mathrm{ns}$. For the time from the first access to the second one t_{CYCLE} must be met, of course.

The short read method is practical for all read registers in the address range 0xC0 ...0xFF, these target registers are R_IRQ_FIFO_BL0 ...R_IRQ_FIFO_BL7 and R_RAM_DATA.

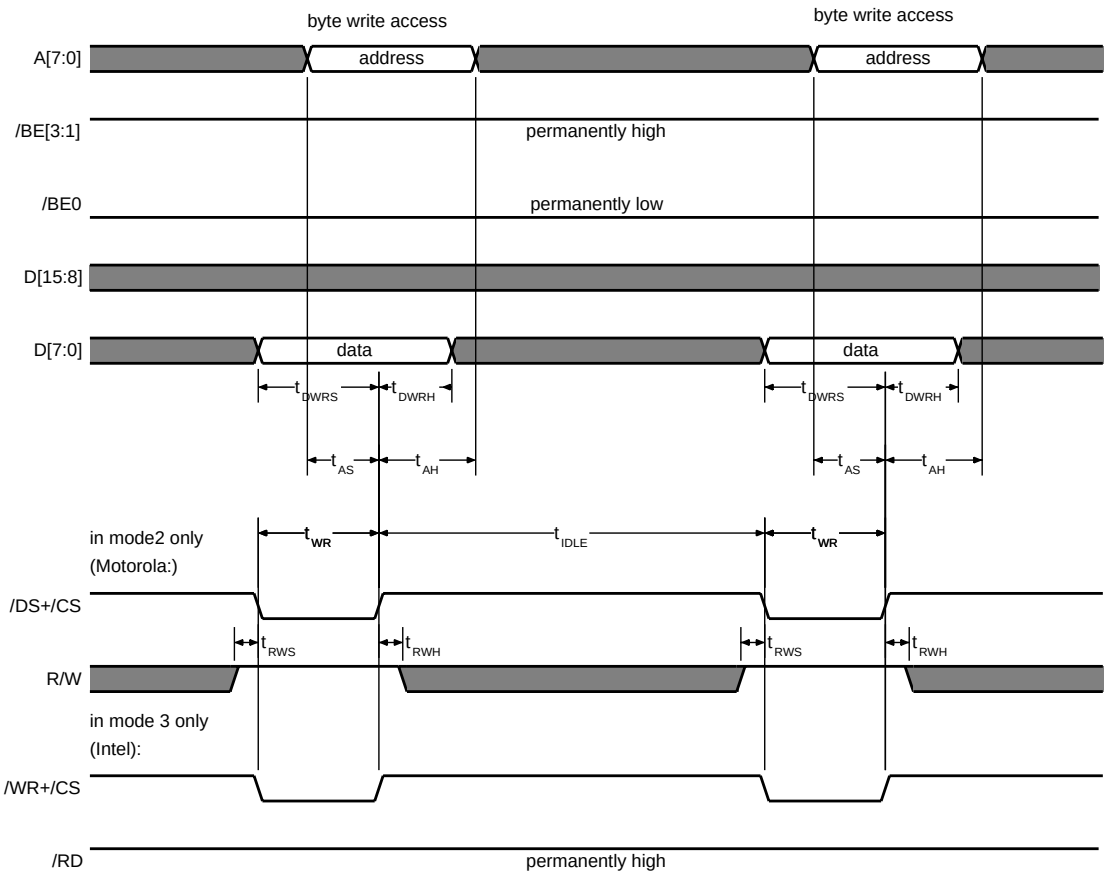

Figure 2.10: *Write access from 8 bit processors in mode 2 (Motorola) and mode 3 (Intel)*

8 bit processors write data like shown in Figure 2.10. Timing values are listed in Table 2.21.

/BE3 ... /BE1 must always be '1'. /BE0 controls the data bus D7 ... D0 and can be fixed to '0'.

Data is written with ⌋of (/DS + /CS) in mode 2 (Motorola) respective (/WR + /CS) in mode 3 (Intel, non-multiplexed). The HFC-E1 requires a data setup time t_{DWRS} and a data hold time t_{DWRH}.

Address and /BE0 (if not fixed to low) require a setup time t_{AS} which starts when all address and byte enable signals are valid. The hold time of these lines is t_{AH}.

2.5.2.2 16 bit processors in mode 2 (Motorola) and mode 3 (Intel)

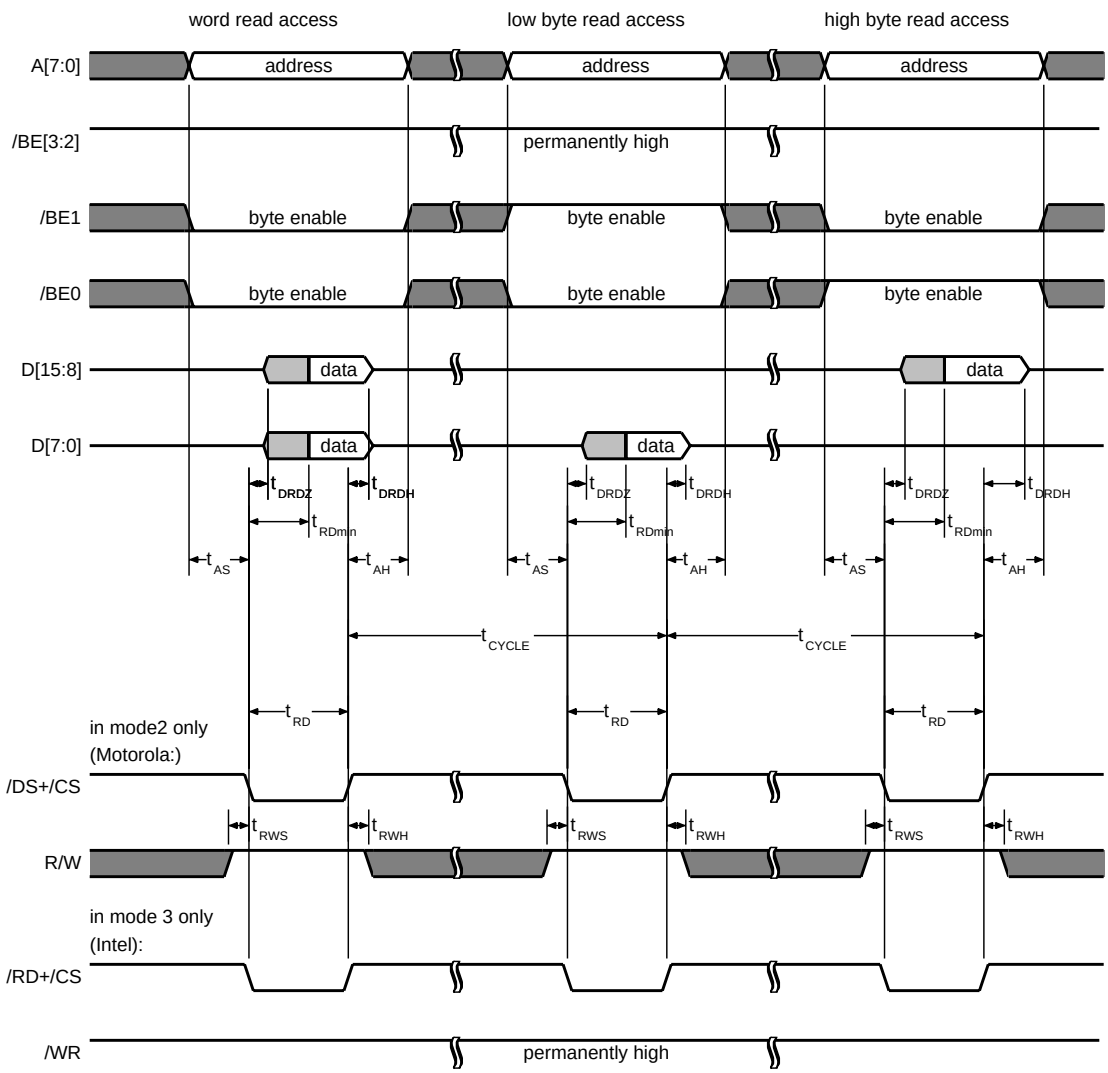

Figure 2.11: *Byte and word read access from 16 bit processors in mode 2 (Motorola) and mode 3 (Intel)*

16 bit processors can either read data with byte or word access like shown in Figure 2.11. FIFO and F-/Z-counter read access have 8 bit or 16 bit width alternatively. The 16 bit processor must support byte access because all other register read accesses must have a width of 8 bit.

/BE2 and /BE3 must always be '1'. /BE0 and /BE1 switch the data bus D15 ... D0 from tristate into data driven state (see Table 2.19).

Data can be read in mode 2 (Motorola) with

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

Table 2.19: *Data access width in mode 2 and 3*

A[0]	/BE1	/BE0	Data access
'X'	'1'	'1'	no access
'0'	'1'	'0'	byte access on D[7:0]
'1'	'0'	'1'	byte access on D[15:8]
'0'	'0'	'0'	word access

In mode 3 (Intel, non-multiplexed) the states

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

must be fulfilled to drive data out. The data bus is stable after t_{RDmin} and returns into tristate after t_{DRDH}.

Address and /BE require a setup time t_{AS} which starts when all address and byte enable signals are valid. The hold time of these lines is t_{AH}.

Table 2.20: *Symbols of read accesses in Figures 2.9 and 2.11*

Symbol	min / ns	max / ns	Characteristic
t_{AS}	10		Address and /BE valid to /DS+/CS (/RD+/CS) ⌉ setup time
t_{AH}	10		Address hold time after /DS+/CS (/RD+/CS) ⌋
t_{DRDZ}	2		/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 ⌋
t_{RD}			Read time:
	20		A[7] = '0' (address range 0 … 0x7F: normal register access)
	20		A[7,6] = '10' (address range 0x80 … 0xBF: FIFO data access)
	$5 \cdot t_{CLKI}$		A[7,6] = '11' (address range 0xC0 … 0xFF: direct RAM access, FIFO interrupt registers) *
t_{CYCLE}			Cycle time between two consecutive /DS+/CS (/RD+/CS) ⌋
	$1.5 \cdot t_{CLKI}$		A[7] = '0' (address range 0 … 0x7F: normal register access)
			A[7,6] = '10' (address range 0x80 … 0xBF: FIFO data access)
	$5.5 \cdot t_{CLKI}$		– after byte access
	$6.5 \cdot t_{CLKI}$		– after word access
	$5.5 \cdot t_{CLKI}$		A[7,6] = '11' (address range 0xC0 … 0xFF: direct RAM access, FIFO interrupt registers)

(*: See 'Short read method' on page 67.)

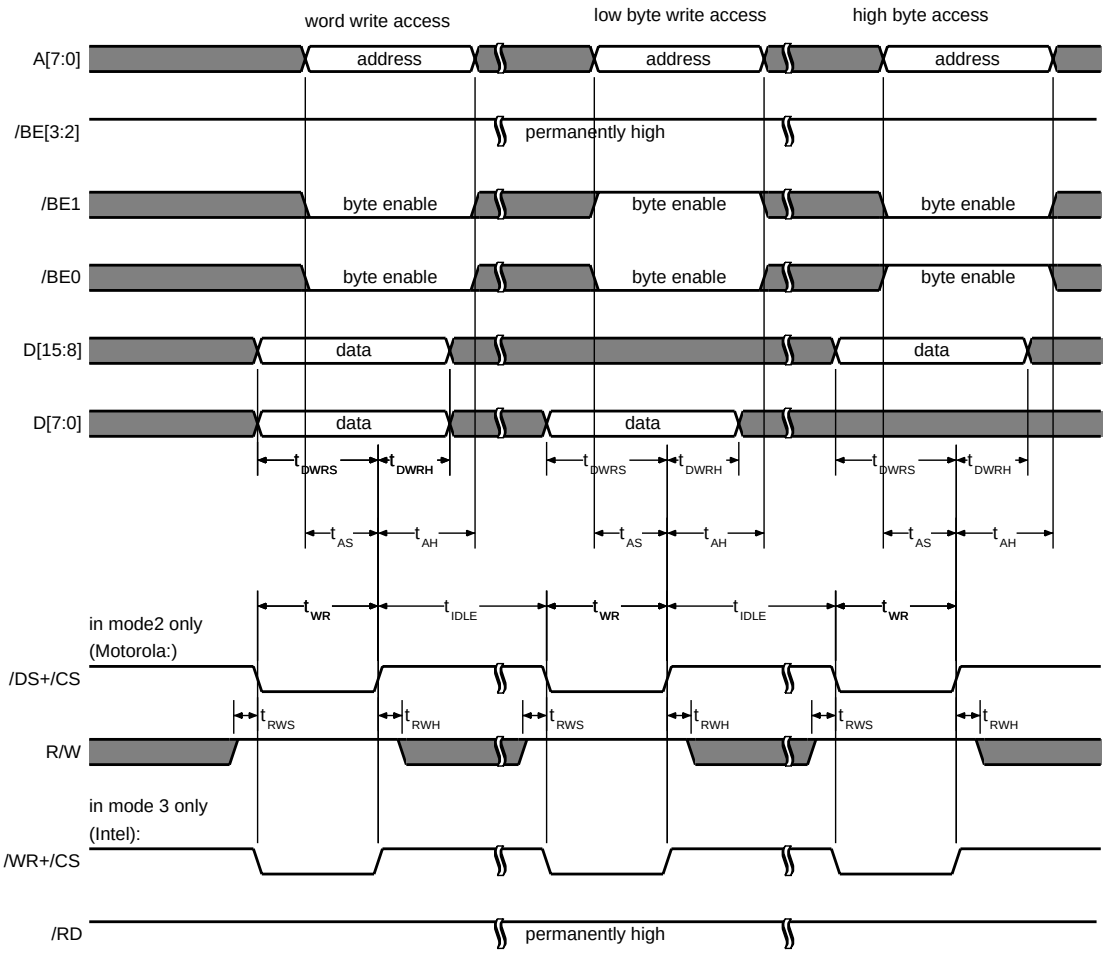

Figure 2.12: *Byte and word write access from 16 bit processors in mode 2 (Motorola) and mode 3 (Intel)*

16 bit processors can either write data with byte or word access like shown in Figure 2.12. FIFO write access have 8 bit or 16 bit width alternatively. The 16 bit processor must support byte access because all other register write accesses must have a width of 8 bit.

/BE2 and /BE3 must always be '1'. /BE0 and /BE1 control the low byte and high byte of the data bus D15 ... D0 (see Table 2.19).

Data is written with ⌋of (/DS + /CS) in mode 2 (Motorola) respective (/WR + /CS) in mode 3 (Intel, non-multiplexed). The HFC-E1 requires a data setup time t_{DWRS} and a data hold time t_{DWRH}.

Address and /BE require a setup time t_{AS} which starts when all address and byte enable signals are valid. The hold time of these lines is t_{AH}.

Table 2.21: *Symbols of write accesses in Figures 2.10 and 2.12*

Symbol	min / ns	max / ns	Characteristic
t_{AS}	10		Address and /BE valid to /DS+/CS (/RD+/CS) ⌋⌈ setup time
t_{AH}	10		Address hold time after /DS+/CS (/RD+/CS) ⌋⌈
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_{RWS}	2		R/W setup time to /DS+/CS ⌉⌊
t_{RWH}	2		R/W hold time after /DS+/CS ⌋⌈
t_{WR}	20		Write time
t_{IDLE}			/DS+/CS (/RD+/CS) high time
	$1.5 \cdot t_{CLKI}$		A[7] = '0' (address range 0 … 0x7F: normal register access)
			A[7,6] = '10' (address range 0x80 … 0xBF: FIFO data access)
	$3.5 \cdot t_{CLKI}$		– after byte access
	$4.5 \cdot t_{CLKI}$		– after word access
	$3.5 \cdot t_{CLKI}$		A[7,6] = '11' (address range 0xC0 … 0xFF: direct RAM access)

2.5.2.3 8 bit processors in mode 4 (Intel, multiplexed)

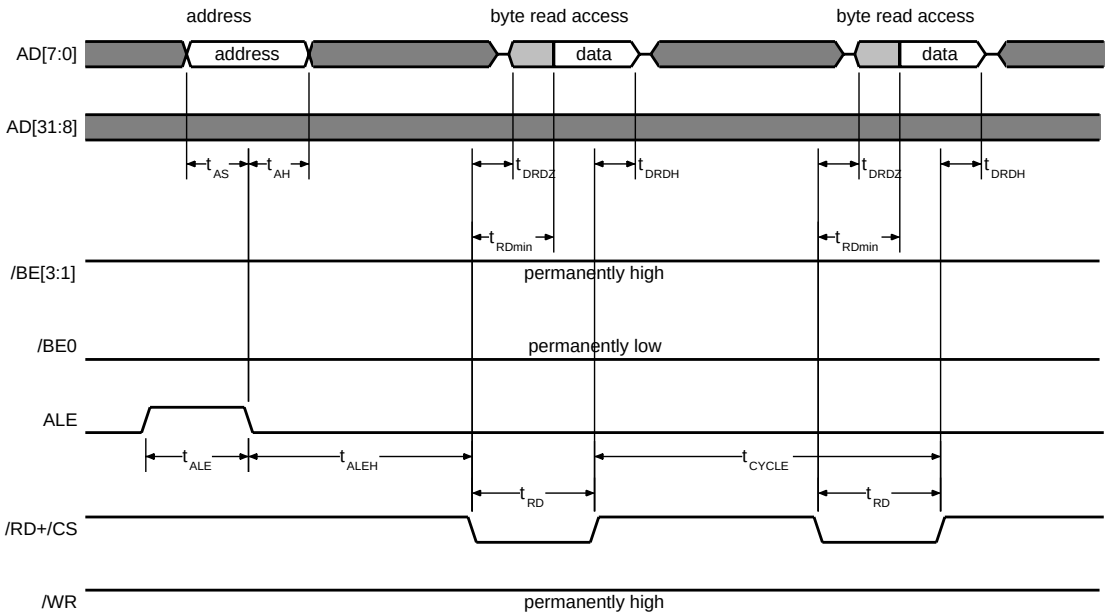

Figure 2.13: *Read access from 8 bit processors in mode 4 (Intel, multiplexed)*

8 bit processors read data like shown in Figure 2.13. Timing values are listed in Table 2.23.

/BE3 … /BE1 must always be '1'. /BE0 can be fixed to '0' or must be low during access to switch the data bus D7 … D0 from tristate into data driven state.

Data can be read in mode 4 (Intel, multiplexed) with [3]

$$\text{/BE0} = \text{'0'} \quad \text{and} \quad (\text{/RD} + \text{/CS}) = \text{'0'} \quad \text{and} \quad \text{/WR} = \text{'1'} \ .$$

The data bus is stable after t_{RDmin} and returns into tristate after t_{DRDH}.

Address and /BE0 (if not fixed to low) require a setup time t_{AS} which starts with the ⌋of ALE. The hold time of these lines is t_{AH}. If two consecutive read accesses are on the same address, multiple register address write is not required.

[3] /RD + /CS means logical OR function of the two signals.

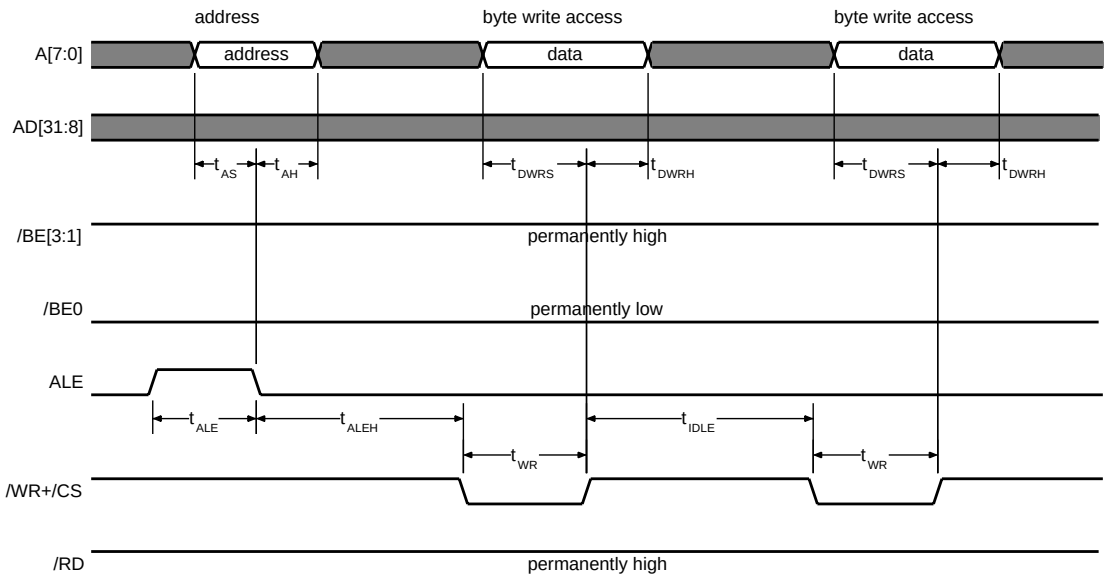

Figure 2.14: *Write access from 8 bit processors in mode 4 (Intel, multiplexed)*

8 bit processors write data like shown in Figure 2.14. Timing values are listed in Table 2.24.

/BE3 … /BE1 must always be '1'. /BE0 controls the data bus D7 … D0 and can be fixed to '0'.

Data is written with ⌋of (/WR + /CS) in mode 4 (Intel, multiplexed). The HFC-E1 requires a data setup time t_{DWRS} and a data hold time t_{DWRH}.

Address and /BE0 (if not fixed to low) require a setup time t_{AS} which starts with the ⌉of ALE. The hold time of these lines is t_{AH}. If two consecutive write accesses are on the same address, multiple register address write is not required.

2.5.2.4 16 bit processors in mode 4 (Intel, multiplexed)

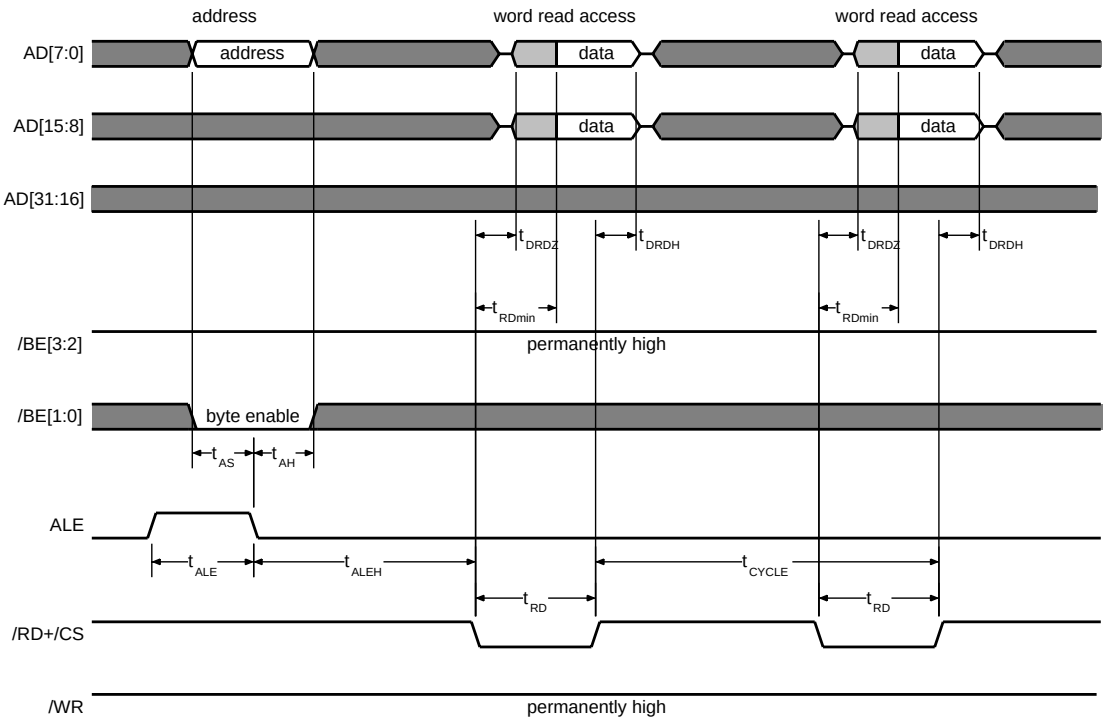

Figure 2.15: *Word read access from 16 bit processors in mode 4 (Intel, multiplexed)*

16 bit processors can either read data with byte or word access. Only 8 bit are used for address decoding. Thus the address on lines AD31 ... AD8 are ignored.

A word read is shown in Figure 2.15. FIFO and F-/Z-counter read access have 8 bit or 16 bit width alternatively. The 16 bit processor must support byte access because all other register read accesses must have a width of 8 bit.

/BE2 and /BE3 must always be '1'. /BE0 and /BE1 switch the data bus D15 ... D0 from tristate into data driven state (see Table 2.22 on page 77).

In mode 4 (Intel, multiplexed) the states

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

must be fulfilled to drive data out. The data bus is stable after t_{RDmin} and returns into tristate after t_{DRDH}.

Address and /BE require a setup time t_{AS} which starts with the ⌋of ALE. The hold time of these lines is t_{AH}. If two consecutive read accesses are on the same address, multiple register address write is not required.

An 8 bit read access (low byte) is performed in the same way as it is done with 8 bit processors. Thus see Figure 2.13 for the timing specification.

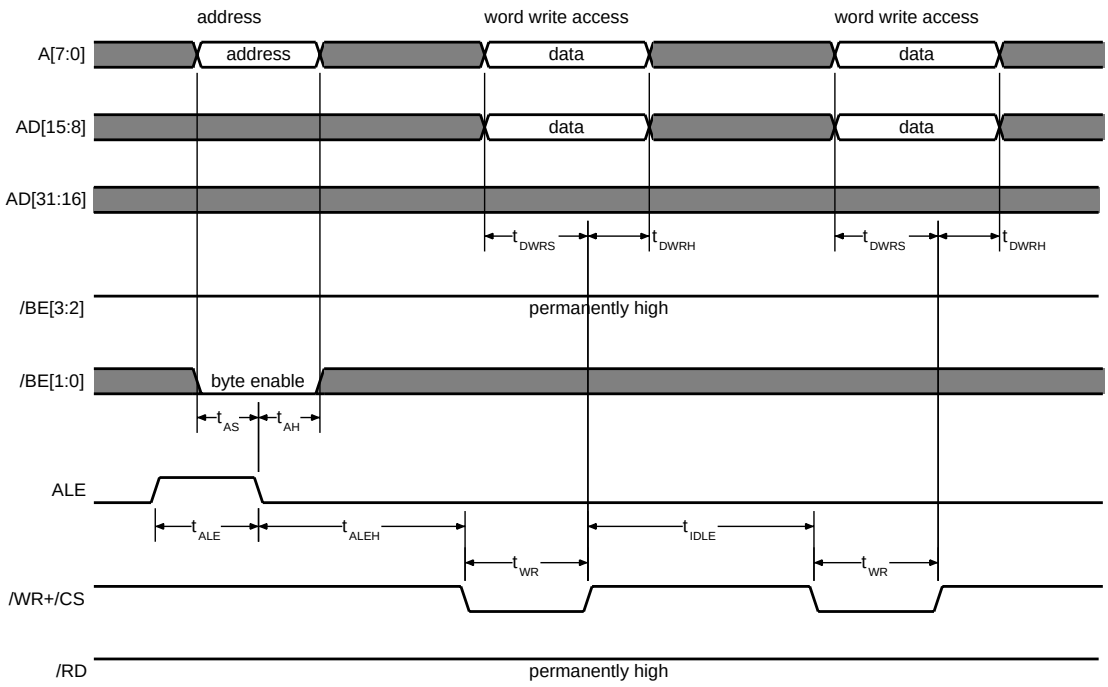

Figure 2.16: *Word write access from 16 bit processors in mode 4 (Intel, multiplexed)*

16 bit processors can either write data with byte or word access. Only 8 bit are used for address decoding. Thus the address on lines AD31 ... AD8 are ignored.

A word write is shown in Figure 2.16. FIFO write access have 8 bit or 16 bit width alternatively. The 16 bit processor must support byte access because all other register write accesses must have a width of 8 bit.

/BE2 and /BE3 must always be '1'. /BE0 and /BE1 control the low byte and high byte of the data bus D15 ... D0 (see Table 2.22 on page 77).

Data is written with ⌋of /WR + /CS in mode 4 (Intel, multiplexed). The HFC-E1 requires a data setup time t_{DWRS} and a data hold time t_{DWRH}.

Address and /BE require a setup time t_{AS} which starts with the ⌉of ALE. The hold time of these lines is t_{AH}. If two consecutive write accesses are on the same address, multiple register address write is not required.

An 8 bit write access (low byte) is performed in the same way as it is done with 8 bit processors. Thus see Figure 2.14 for the timing specification.

2.5.2.5 32 bit processors in mode 4 (Intel, multiplexed)

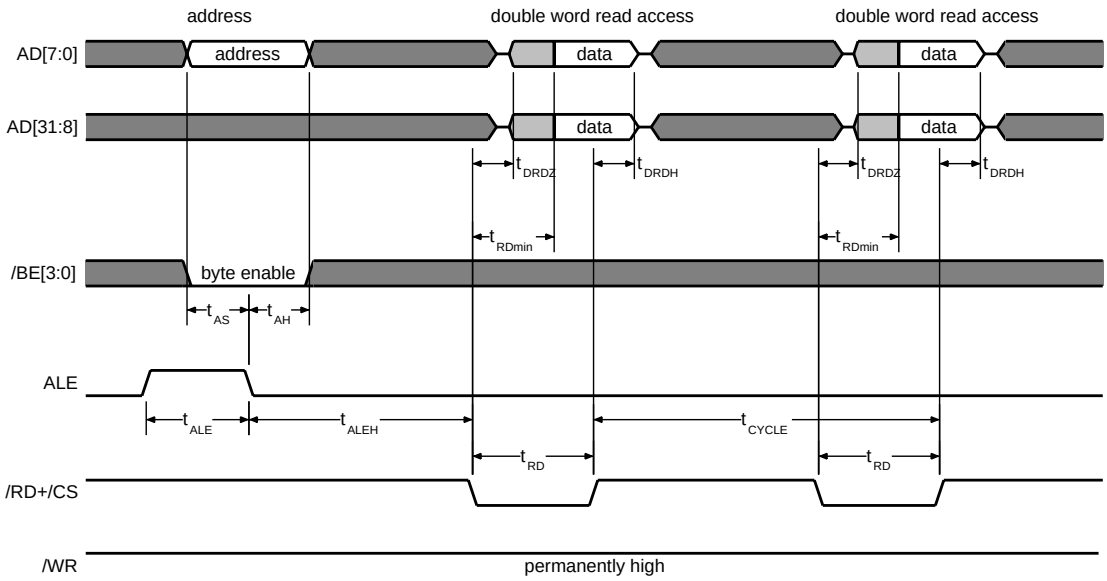

Figure 2.17: *Double word read access from 32 bit processors in mode 4 (Intel, multiplexed)*

32 bit processors can either read data with byte, word or double word access. Only 8 bit are used for address decoding. Thus the address on lines AD31 … AD8 are ignored.

A double word read is shown in Figure 2.17. FIFO and Z-counter read access have 8 bit, 16 bit or 32 bit width alternatively, F-counter read access have 8 bit or 16 bit width alternatively. The 32 bit processor must support byte access because all other register read accesses must have a width of 8 bit.

Table 2.22: *Data access width in mode 4*

A[0]	/BE3	/BE2	/BE1	/BE0	Data access
'X'	'1'	'1'	'1'	'1'	no access
'0'	'1'	'1'	'1'	'0'	byte access on AD[7:0]
'1'	'1'	'1'	'0'	'1'	byte access on AD[15:8]
'0'	'1'	'0'	'1'	'1'	byte access on AD[23:16]
'1'	'0'	'1'	'1'	'1'	byte access on AD[31:24]
'0'	'1'	'1'	'0'	'0'	word access on AD[15:0]
'0'	'0'	'0'	'1'	'1'	word access on AD[31:16]
'0'	'0'	'0'	'0'	'0'	double word access

/BE3 … /BE0 switch the bus lines AD31 … AD0 from tristate into data driven state during data phase (see Table 2.22).

In mode 4 (Intel, multiplexed) the states

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

must be fulfilled to drive data out. The data bus is stable after t_{RDmin} and returns into tristate after t_{DRDH}.

Address and /BE require a setup time t_{AS} which starts with the ⌉of ALE. The hold time of these lines is t_{AH}. If two consecutive read accesses are on the same address, multiple register address write is not required.

An 8 bit read access (low byte) is performed in the same way as it is done with 8 bit processors. Thus see Figure 2.13 for the timing specification.

Table 2.23: *Symbols of read accesses in Figures 2.13, 2.15 and 2.17*

Symbol	min / ns	max / ns	Characteristic
t_{ALE}	10		Address latch time
t_{ALEH}	0		ALE ⌋ to /WR+/CS ⌋
t_{AS}	10		Address and /BE valid to /RD+/CS ⌉ setup time
t_{AH}	10		Address hold time after /RD+/CS ⌋
t_{DRDZ}	2		/RD+/CS ⌉ to data buffer turn on time
t_{DRDH}	2	15	/RD+/CS ⌋ to data buffer turn off time
t_{RD}	20		Read time:
	20		A[7] = '0' (address range 0 … 0x7F: normal register access)
	20		A[7,6] = '10' (address range 0x80 … 0xBF: FIFO data access)
	$5 \cdot t_{CLKI}$		A[7,6] = '11' (address range 0xC0 … 0xFF: direct RAM access, FIFO interrupt registers) *
t_{CYCLE}			Cycle time between two consecutive /RD+/CS ⌋
	$1.5 \cdot t_{CLKI}$		A[7] = '0' (address range 0 … 0x7F: normal register access)
			A[7,6] = '10' (address range 0x80 … 0xBF: FIFO data access)
	$5.5 \cdot t_{CLKI}$		– after byte access
	$6.5 \cdot t_{CLKI}$		– after word access
	$5.5 \cdot t_{CLKI}$		A[7,6] = '11' (address range 0xC0 … 0xFF: direct RAM access, FIFO interrupt registers)

(*: See 'Short read method' on page 67.)

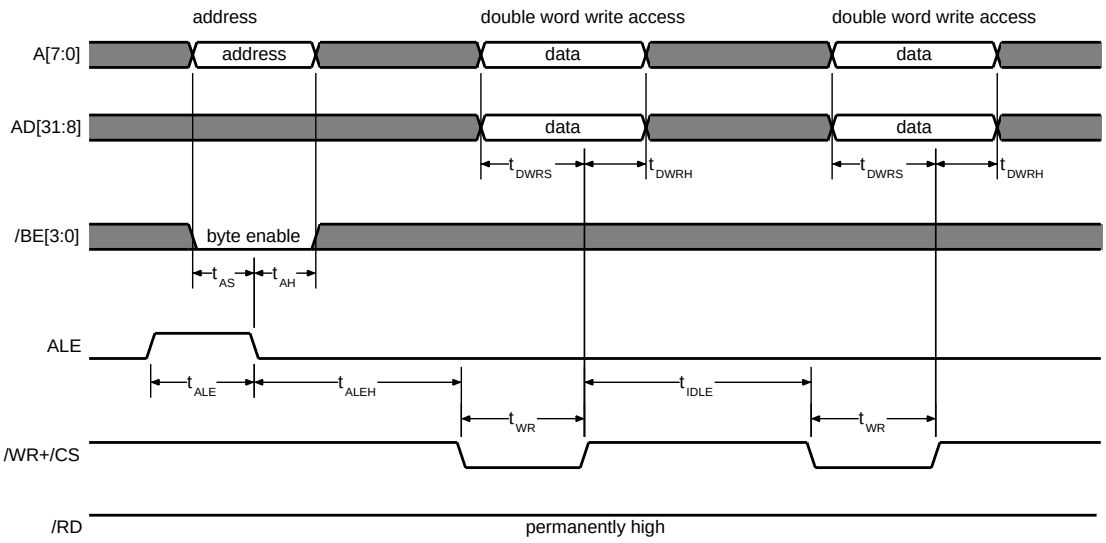

Figure 2.18: *Write access from 32 bit processors in mode 4 (Intel, multiplexed)*

32 bit processors can either write data with byte, word or double word access. Only 8 bit are used for address decoding. Thus the address on lines AD31 ... AD8 are ignored.

A double word write is shown in Figure 2.18. FIFO write access have 8 bit, 16 bit or 32 bit width alternatively. The 32 bit processor must support byte access because all other register write accesses must have a width of 8 bit.

/BE3 ... /BE0 control the bus lines AD31 ... AD0 during data phase (see Table 2.22).

Data is written with ⌋of /WR + /CS in mode 4 (Intel, multiplexed). The HFC-E1 requires a data setup time t_{DWRS} and a data hold time t_{DWRH}.

Address and /BE require a setup time t_{AS} which starts with the ⌉of ALE. The hold time of these lines is t_{AH}. If two consecutive write accesses are on the same address, multiple register address write is not required.

An 8 bit write access (low byte) is performed in the same way as it is done with 8 bit processors. Thus see Figure 2.14 for the timing specification.

Table 2.24: *Symbols of write accesses in Figures 2.14, 2.16 and 2.18*

Symbol	min / ns	max / ns	Characteristic
t_{ALE}	10		Address latch time
t_{ALEH}	0		ALE ⌋⌈ to /WR+/CS ⌋⌈
t_{AS}	10		Address and /BE valid to /WR+/CS ⌋⌈ 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
	$1.5 \cdot t_{CLKI}$		A[7] = '0' (address range 0 … 0x7F: normal register access)
			A[7,6] = '10' (address range 0x80 … 0xBF: FIFO data access)
	$3.5 \cdot t_{CLKI}$		– after byte access
	$4.5 \cdot t_{CLKI}$		– after word access
	$3.5 \cdot t_{CLKI}$		A[7,6] = '11' (address range 0xC0 … 0xFF: direct RAM access)

2.5.3 Examples of processor connection circuitries

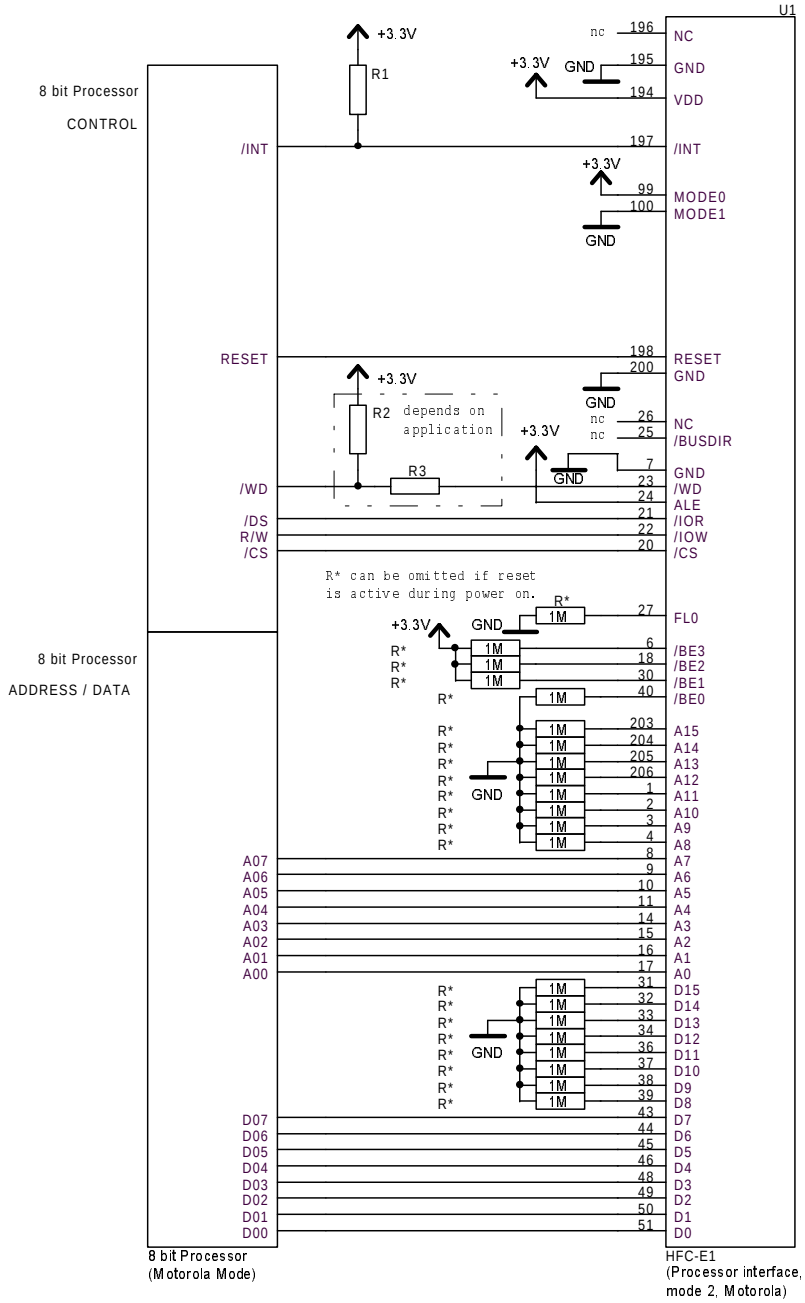

Figure 2.19: *8 bit Intel / Motorola processor circuitry example (mode 2)*

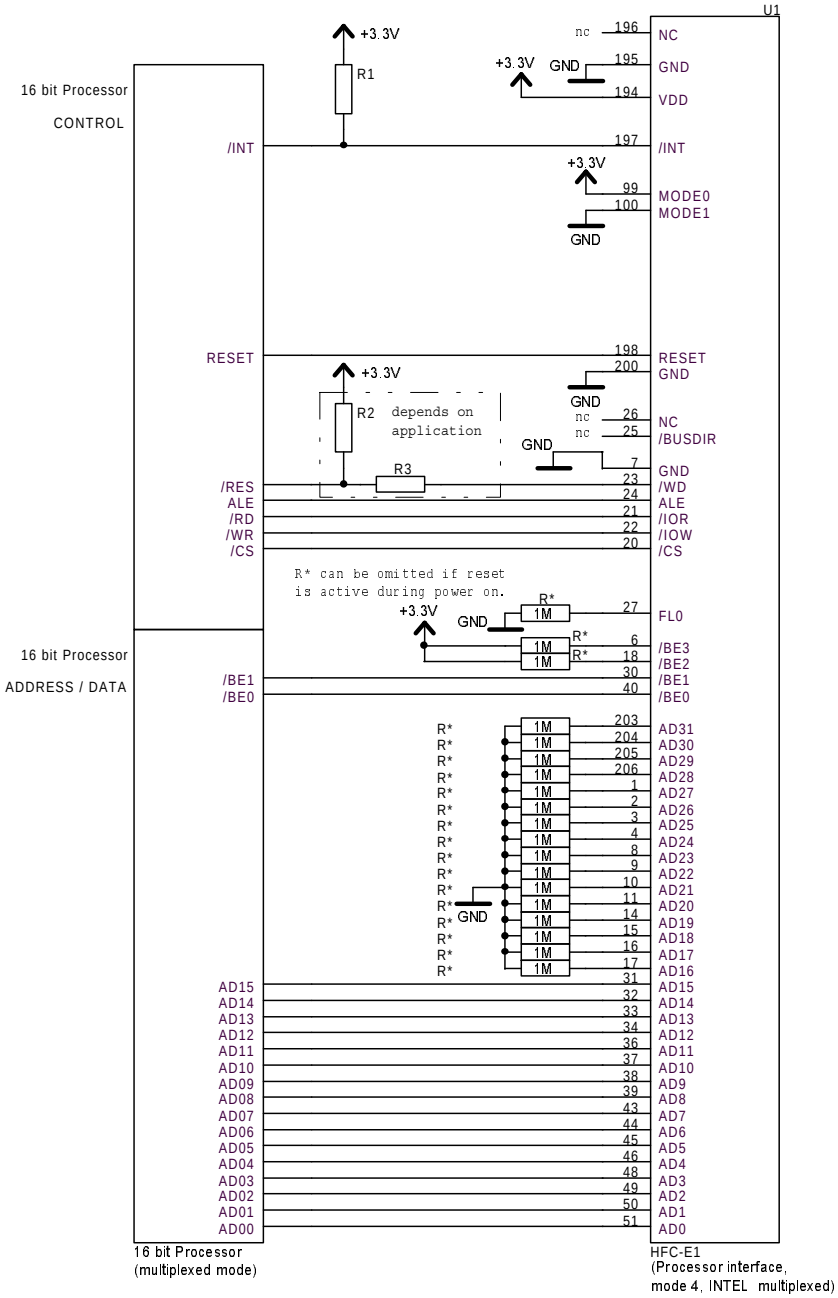

Figure 2.20: *16 bit Intel processor circuitry example (mode 4, multiplexed)*

2.6 Serial processor interface (SPI)

Table 2.25: *Overview of the SPI interface pins*

Number	Name	Description
194	/SPISEL	SPI device select low active
195	SPI_RX	SPI receive data input
196	SPI_TX	SPI transmit data output
197	/INT	Interrupt request
198	RESET	Reset high active
200	SPICLK	SPI clock input

The SPI interface mode is selected by MODE0 = 1, MODE1 = 0 and connecting pin 200 to SPI clock. /SPISEL must be high during reset. The first positive edge on SPICLK switches the interface from processor interface mode into SPI mode. This may be the first positive clock at the start of an SPI access.

The interface has 4 pins as shown in Table 2.25. For further information please see the SPI specification.

2.6.1 SPI read and write access

In SPI mode each data transfer is 16 bit long. From the first 8 bits only the bits $R/\overline{W}$ and $ADR/\overline{DAT}$ are used. The other 6 bits must be zero. Depending on the $R/\overline{W}$ bit the second 8 bits are read from the HFC-E1 or written into the HFC-E1 as shown in the Figures 2.21 and 2.22. So all data accesses in SPI mode handle 8 data bits.

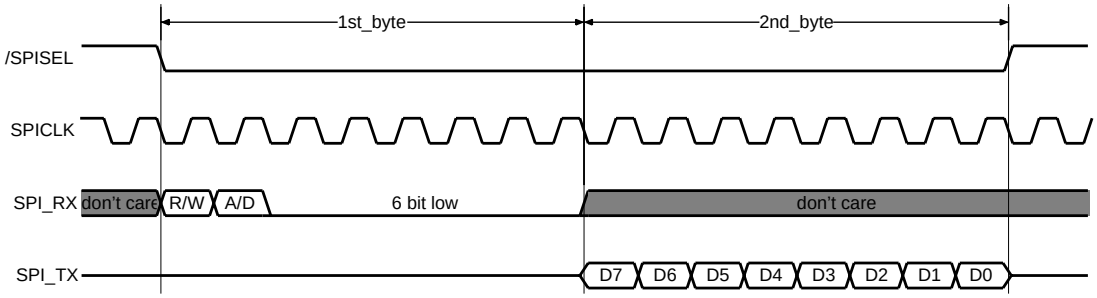

Figure 2.21: *SPI read access*

It is allowed to interrupt the /SPISEL signal between the two bytes. In this case the transmission pauses and will be continued after /SPISEL returns to low level. An example for an interrupted read access is shown in Figure 2.23.

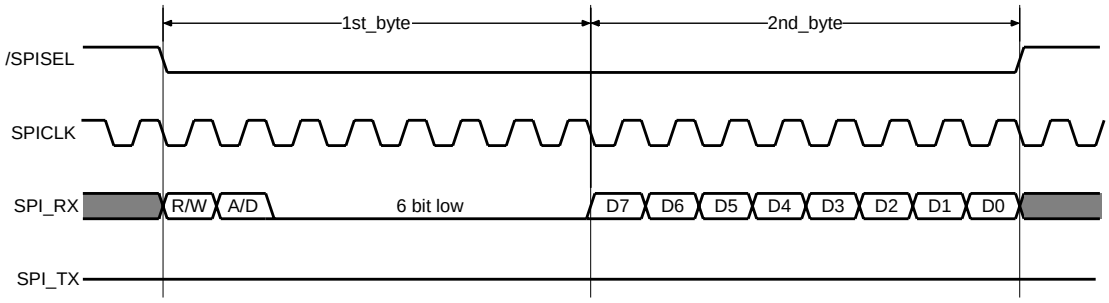

Figure 2.22: *SPI write access*

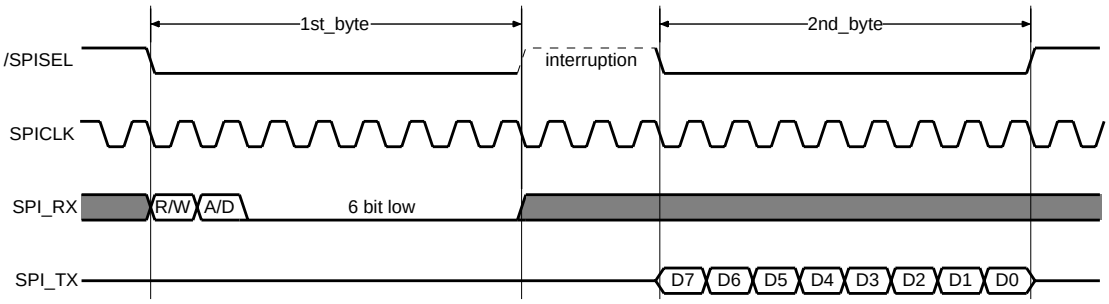

Figure 2.23: *Interrupted SPI read access*

2.6.2 SPI connection circuitry

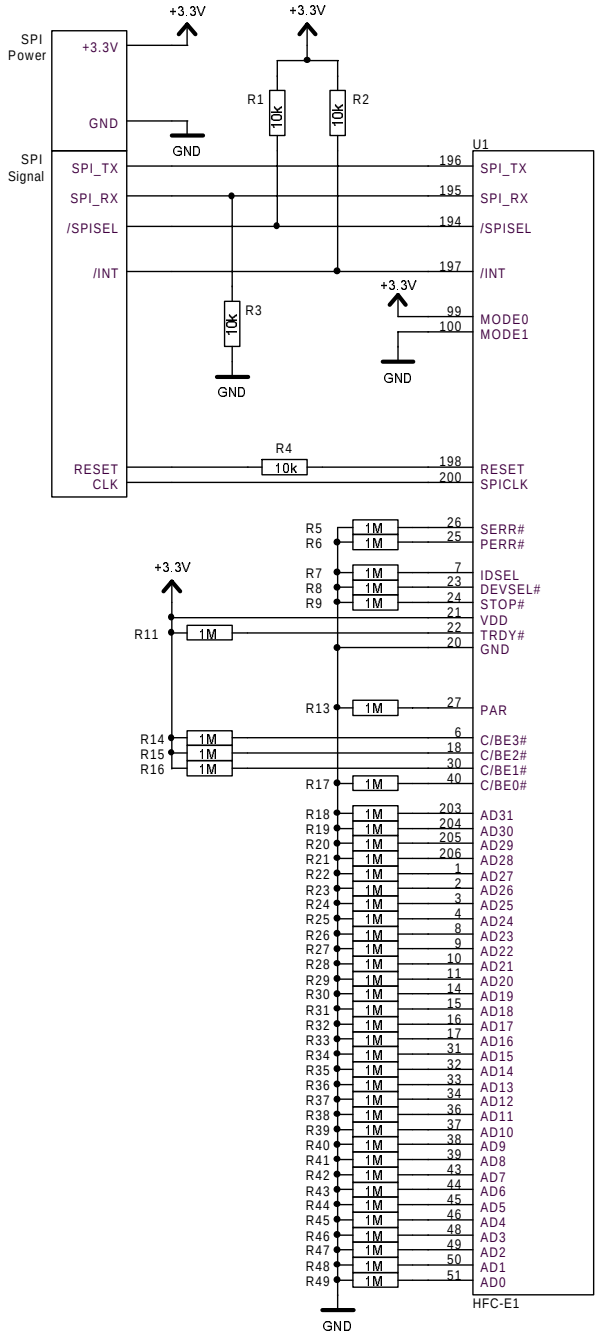

Figure 2.24: *SPI connection circuitry*

2.7 Register description

2.7.1 Write only registers

R_CIRM	(write only)	0x00
Interrupt and reset register		

Bits	Reset Value	Name	Description
2..0	0	V_IRQ_SEL	**IRQ channel selection in ISA PnP mode** '000' = interrupt lines disable '001' = IRQ0 '010' = IRQ1 '011' = IRQ2 '100' = IRQ3 '101' = IRQ4 '110' = IRQ5 '111' = IRQ6
3	0	V_SRES	**Soft reset** This reset is similar to the hardware reset. The selected I/O address (CIP) remains unchanged. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
4	0	V_HFCRES	**HFC-reset** Sets all FIFO and HDLC registers to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
5	0	V_PCMRES	**PCM reset** Sets all PCM registers to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
6	0	V_E1RES	**E1-reset** Sets all E1 interface registers to their initial values. The reset is active until the bit is cleared. '0' = deactivate reset '1' = activate reset
7	0	V_RLD_EPR	**EEPROM reload** '0' = normal operation '1' = reload EEPROM to SRAM This bit must be cleared by software. The reload is started when the bit is cleared.

(For reset group description see Table 12.4 on page 235.)

R_CTRL	(write only)	0x01

Common control register

Bits	Reset Value	Name	Description
0	0	(reserved)	Must be ’0’.
1	0	V_FIFO_LPRIO	**FIFO access priority for host accesses** ’0’ = normal priority ’1’ = low priority
2	0	V_SLOW_RD	**One additional wait cycle for PCI read accesses** ’0’ = normal operation ’1’ = additional wait (must be set for 66 MHz PCI operation)
3	0	V_EXT_RAM	**Use external RAM** The internal SRAM is switched off when external SRAM is used. ’0’ = internal SRAM is used in lower 32 kByte address space ’1’ = external SRAM is used
4	0	(reserved)	Must be ’0’.
5	0	V_CLK_OFF	**CLK oscillator** ’0’ = normal operation ’1’ = CLK oscillator is switched off This bit is reset at every write access to the HFC-E1.
7..6	0	(reserved)	Must be ’00’.

R_RAM_ADDR0	(write only)	0x08

Address pointer, register 0

1st address byte for internal / external SRAM access.

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

R_RAM_ADDR1	(write only)	0x09

Address pointer, register 1

2nd address byte for internal / external SRAM access.

Bits	Reset Value	Name	Description
7..0	0x00	V_RAM_ADDR1	**Address bits 15 . . . 8**

R_RAM_ADDR2	(write only)	0x0A

Address pointer, register 2

High address bits for internal / external SRAM access and access configuration.

Bits	Reset Value	Name	Description
3..0	0	V_RAM_ADDR2	**Address bits 19 . . . 16**
5..4		(reserved)	Must be '00'.
6	0	V_ADDR_RES	**Address reset** '0' = normal operation '1' = address bits 0 . . . 15 are set to zero This bit is automatically cleared.
7	0	V_ADDR_INC	**Address increment** '0' = no address increment '1' = automatically increment of the address after every write or read on register R_RAM_DATA

R_RAM_MISC	(write only)	0x0C

RAM size setup and miscellaneous functions register

Bits	Reset Value	Name	Description
1..0	0	**V_RAM_SZ**	**RAM size** '00' = 32k x 8 '01' = 128k x 8 '10' = 512k x 8 '11' = reserved After setting V_RAM_SZ to a value different from '00' a soft reset should be initiated.
3..2		**(reserved)**	Must be '00'.
4	0	**V_PWM0_16KHZ**	**16 kHz signal on pin PWM0** '0' = normal PWM0 function '1' = 16 kHz output
5	0	**V_PWM1_16KHZ**	**16 kHz signal on pin PWM1** '0' = normal PWM1 function '1' = 16 kHz output
6		**(reserved)**	Must be '0'.
7	0	**V_FZ_MD**	**Exchange F- / Z-counter context** (for transmit FIFOs only) '0' = A_Z1L, A_Z1H = $Z1(F1)$ and A_Z2L, A_Z2H = $Z2(F1)$ (normal operation) '1' = A_Z1L, A_Z1H = $Z1(F1)$ and A_Z2L, A_Z2H = $Z2(F2)$ (exchanged operation) This bit can be used to check the actual RAM usage of transmit FIFOs.

2.7.2 Read only registers

R_RAM_USE	(read only)	0x15

SRAM duty factor

Usage of SRAM access bandwidth by the internal data processor.

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

R_RAM_DATA	(read / write)	0xC0

SRAM data access

Direct access to internal / external SRAM

Bits	Reset Value	Name	Description
7..0	0	V_RAM_DATA	**SRAM data access** The address must be written into the registers R_RAM_ADDR0 . . . R_RAM_ADDR2 in advance.

R_CHIP_ID	(read only)	0x16

Chip identification register

Bits	Reset Value	Name	Description
3..0	0	V_PNP_IRQ	**IRQ assigned by the PnP BIOS** (only in ISA PnP mode) V_IRQ_SEL of the R_CIRM register must be set to the value corresponding to the hardware connected IRQ lines.
7..4	0xE	V_CHIP_ID	**Chip identification code** '1110' means HFC-E1.

R_CHIP_RV	(read only)	0x1F

HFC-E1 revision

Bits	Reset Value	Name	Description
3..0	1	V_CHIP_RV	**Chip revision 1** (Engineering samples were revision 0.)
7..4	0	(reserved)	

Chapter 3

HFC-E1 data flow

Table 3.1: *Overview of the HFC-E1 data flow registers*

Write only registers:					
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x0B	R_FIRST_FIFO	119	0x34	R_TX_OFF	167
0x0D	R_FIFO_MD	120	0xF4	A_CH_MSK	124
0x0F	R_FIFO	121	0xFA	A_CON_HDLC	125
0x0F	R_FSM_IDX	121	0xFB	A_SUBCH_CFG	126
0x10	R_SLOT	122	0xFC	A_CHANNEL	127
0xD0	A_SL_CFG	123	0xFD	A_FIFO_SEQ	127

3.1 Data flow concept

The HFC-E1 has a programmable data flow unit, in which the FIFOs are connected with the PCM and the E1 interface. Moreover the data flow unit can directly connect PCM and E1 interface or two PCM time slots [1].

The fundamental features of the HFC-E1 data flow are as follows:

- programmable interconnection capability between FIFOs, PCM time slots and E1 time slots
- in transmit and receive direction there are
  - up to 32 FIFOs
  - 16, 32 or 64 PCM time slots
  - 32 E1 time slots
  - 32 HFC-channels to connect the above-mentioned data interfaces
- 3 data flow modes to satisfy different application tasks
- subchannel processing for bitwise data handling

The complete HFC-E1 data flow block diagram is shown in Figure 3.1. Basically, data routing requires an allocation number at each block. So there are three areas where numbering is based on FIFOs, HFC-channels and PCM time slots.

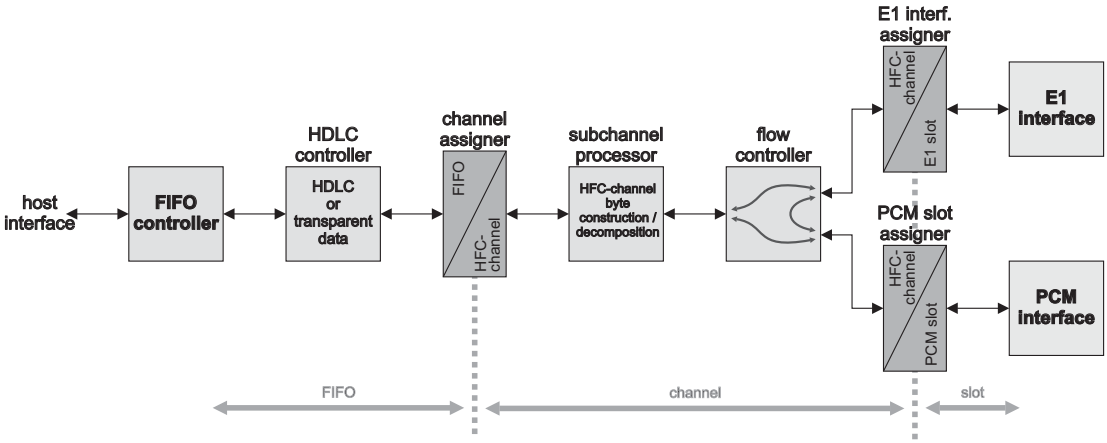

Figure 3.1: *Data flow block diagram*

FIFO handling and HDLC controller, PCM and E1 interface are described in Chapters 4 to 6. So this chapter deals with the data flow unit which is located between and including the channel assigner, the PCM slot assigner and the E1 slotassigner.

Term definitions

Figure 3.2 clarifies the relationship and the differences between the numbering of FIFOs, HFC-channels and PCM time slots. The inner circle symbolizes the HFC-channel oriented

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

part of the data flow, while the outer circle shows the connection of three data sources and data drains respectively. The E1 interface have a fixed mapping between HFC-channels and E1 time slots so that there is no need of a separate E1 time slot numbering.

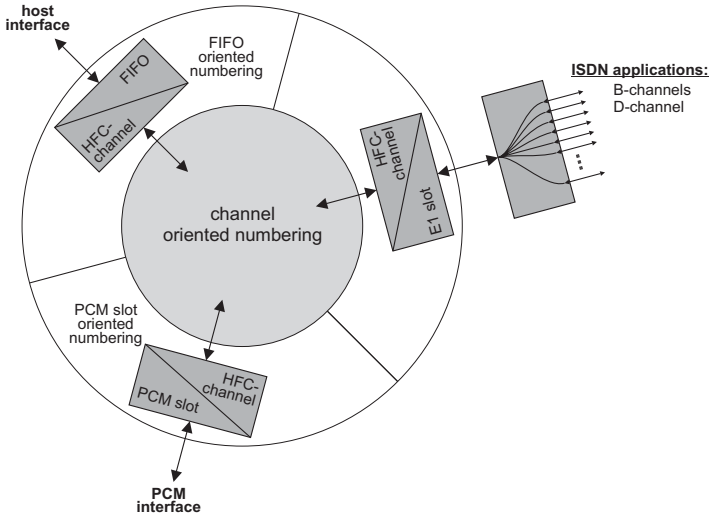

Figure 3.2: *Areas of FIFO oriented, HFC-channel oriented and PCM time slot oriented numbering*

FIFO: The FIFOs are buffers between the universal bus interface and the PCM and E1 interface. The HDLC controllers are located on the non host bus side of the FIFOs. The number of FIFOs depends on the FIFO size configuration (see Section 4.2) and starts with number 0. The maximum FIFO number is 31. Furthermore data directions transmit and receive are associated with every FIFO number.

HFC-channel: HFC-channels are used to define data paths between FIFOs on the one side and PCM and E1 interface on the other side. The HFC-channels are numbered 0 … 31. Furthermore data directions transmit and receive are associated with every HFC-channel number.

It is important not to mix up the HFC-channels of the here discussed data flow (inner circle of Figure 3.2) with the B-channels and the D-channel of the E1 interface.

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 (2 MBit/s), 64 time slots (4 MBit/s) or 128 time slots (8 MBit/s). As data directions transmit and receive are associated with every time slot number, slots are numbered 0 … 15, 0 … 31 or 0 … 63.

E1 time slot: The E1 data stream is organized in time slots. E1 time slots have always the same number and data direction as the associated HFC-channel.

Each FIFO, HFC-channel and time slot number exist for transmit and receive direction. The data rate is always 8 kByte/s for every E1 time slot and every PCM time slot. FIFOs, HFC-channels, E1 time slots and PCM time slots have always a width of 8 bit.

3.2 Flow controller

The various connections between FIFOs, E1 time slots and PCM time slots are set up by programming the flow controller, the channel assigner and the PCM slot assigner.

The flow controller sets up connections between FIFOs and the E1 interface, FIFOs and the PCM interface and between the E1 and PCM interface. The bitmap V_DATA_FLOW of the register A_CON_HDLC (which exists for each FIFO) configures these connections. The numbering of transmit and corresponding receive FIFOs, HFC-channels and PCM time slots is independent from each other. But in practice the connection table is more clear if the same number is chosen for corresponding transmit and receive direction.

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

The flow controller operates on HFC-channel data. Nevertheless it is programmed with a bitmap of a FIFO-indexed array register. With this concept it is possible to change the FIFO-to-HFC-channel assignment of a ready-configured FIFO without re-programming its parameters again.

The internal structure of the flow controller contains

- 4 switching buffers, i.e. one for the E1 and PCM interface in transmit and receive direction each and
- 3 switches to control the data paths.

Switching buffers

The switching buffers decouple the data inside the flow controller from the data that is transmitted / received from / to the E1 and PCM interfaces. With every 125 μs cycle the switching buffers change their pointers.

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

A direct PCM-to-E1 connection delays each data byte two cycles. That means the received byte is stored in the switching buffer during the first 125 μs cycle, then copied into the transmit buffer during the second 125 μs cycle and finally transmitted from the interface during the third 125 μs cycle. If the conference unit is switched on, there is an additional 125 μs delay, because the summation of the whole frame is processed in the memory (see Section 8).

Timed sequence

The data transmission algorithm of the flow controller is FIFO-oriented and handles all FIFOs every 125 μs in the following sequence [2]:

1. FIFO[0,TX]
2. FIFO[0,RX]
3. FIFO[1,TX]
4. FIFO[1,RX]

⋮

63. FIFO[31,TX]
64. FIFO[31,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.

The HFC-E1 has three data flow modes. One of them (*FIFO sequence mode*) is used to configure a programmable FIFO sequence which can be used instead of the ascending FIFO numbering. This is explained in Section 3.4.

Transmit operation

In transmit operation one HDLC or transparent byte is read and can be transmitted to the E1 and the PCM interface as shown in Figure 3.3. Furthermore, data can be transmitted from the E1 interface to the PCM interface. From the flow controller point of view, the switches select the source for outgoing data.The switches are controlled by the bitmap V_DATA_FLOW[2..0] of the register A_CON_HDLC[n,TX] where n is a FIFO number.

- FIFO data is only transmitted to the E1 interface if V_DATA_FLOW[1] = 0.
- The PCM interface can transmit a data byte which comes either from the FIFO or from the E1 interface. Bit V_DATA_FLOW[2] selects the source for the PCM transmit slot (see Figure 3.3). The receiving E1 time slot has always the same number as the transmitting E1 time slot.
- The bit V_DATA_FLOW[0] is ignored in transmit operation.

Receive operation

Figure 3.4 shows the flow controller structure in receive operation. The two switches are controlled with the bitmap V_DATA_FLOW[2..0]. FIFO data can either be received from the E1 or PCM interface. Furthermore, data can be transmitted from the PCM interface to the E1 interface.

[2] Due to the FIFO size setup (see Section 4.2) the maximum number of FIFOs might be less than 31.

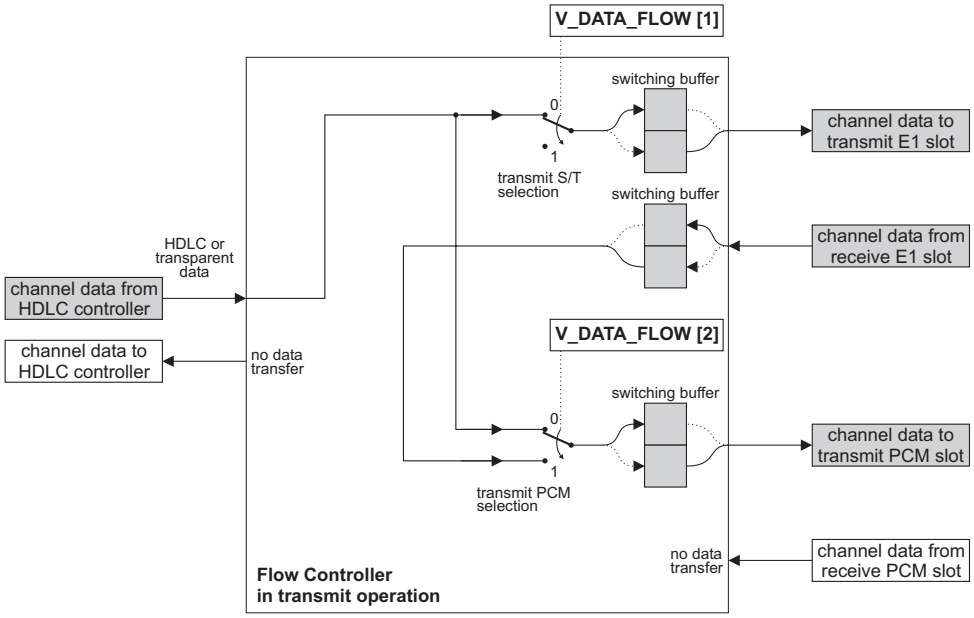

Figure 3.3: *The flow controller in transmit operation*

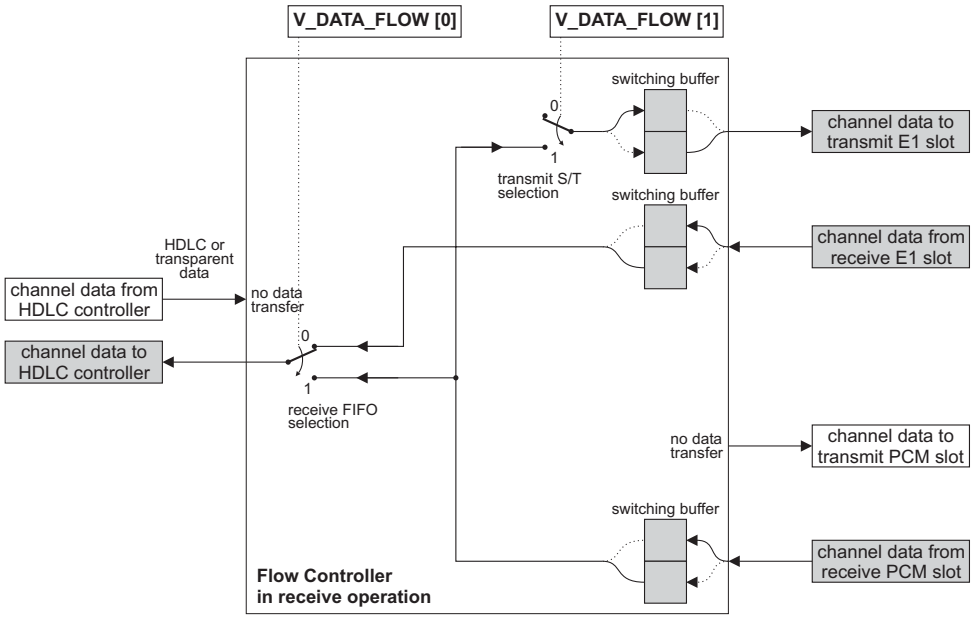

Figure 3.4: *The flow controller in receive FIFO operation*

- Bit V_DATA_FLOW[0] selects the source for the receive FIFO which can either be the PCM or the E1 interface.
- Furthermore, the received PCM byte can be transferred to the E1 interface. This requires bit V_DATA_FLOW[1] = 1.
- The bit V_DATA_FLOW[2] is ignored in receive FIFO operation.

Connection summary

Table 3.2 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. These rows have always an additional connection to a second destination.

Table 3.2: *Flow controller connectivity*

V_DATA_FLOW	Transmit		Receive FIFO	
000	FIFO → E1	FIFO → PCM	FIFO ← E1	
001	FIFO → E1	FIFO → PCM	FIFO ← PCM	
010	FIFO → PCM		FIFO ← E1	E1 ← PCM
011	FIFO → PCM		FIFO ← PCM	E1 ← PCM
100	FIFO → E1	E1 → PCM	FIFO ← E1	
101	FIFO → E1	E1 → PCM	FIFO ← PCM	
110	E1 → PCM		FIFO ← E1	E1 ← PCM
111	E1 → PCM		FIFO ← PCM	E1 ← PCM

The most important connections are data transmissions to a single destination. For these connections it is possible to manage the configuration programming of V_DATA_FLOW with only four different values for transmit and receive FIFO operations. Table 3.3 shows the suitable programming values which can be used to simplify the programming algorithm.

Table 3.3: *V_DATA_FLOW programming values for single-destination connections*

Connection			Required V_DATA_FLOW	Equalized V_DATA_FLOW	Data direction
FIFO	→	E1	'10x'	'100'	transmit
FIFO	←	E1	'x00'		receive
FIFO	→	PCM	'01x'	'011'	transmit
FIFO	←	PCM	'x01'	'001'	receive
E1	→	PCM	'11x'	'110'	transmit
E1	←	PCM	'x10'		receive

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

3.3 Assigners

The data flow block diagram in Figure 3.1 contains three assigners. These functional blocks are used to connect FIFOs, HFC-channels and E1 time slots and PCM time slots respectively with each other.

3.3.1 HFC-channel assigner

The channel assigner functionality depends on the data flow mode described in Section 3.4.

3.3.2 PCM slot assigner

The PCM slot assigner can connect each HFC-channel to an arbitrary PCM time slot. Therefore, for a specified time slot [4] the connected HFC-channel number and data direction must be written into the register A_SL_CFG[SLOT] as follows:

$$\begin{aligned} \text{A\_SL\_CFG}: \text{V\_CH\_DIR1[SLOT]} &= \text{<HFC-channel data direction>} \\ : \text{V\_CH\_NUM1[SLOT]} &= \text{<HFC-channel number>} \end{aligned}$$

Typically, the data direction of a HFC-channel and its connected slot is the same. However, for a direct connection between a PCM time slot and an E1 time slot, transmit and receive direction have to be connected.

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

3.3.3 E1 slot assigner

The E1 interface consists of 32 time slots for transmit data and 32 time slots for receive data.

In HFC-E1 applications these time slots are typically used for ISDN data transfer. Then time slot 0 is reserved for the synchronization process and time slot 16 is normally used to be the D-channel. All the other time slots are assigned to B-channels for the ISDN data transmission.

Between the HFC-channels [5] and the E1 time slots there is a simple assignment:

$$\begin{aligned} \text{HFC-channel}[n,\text{TX}] &\leftrightarrow \text{E1 slot}[n,\text{TX}] \\ \text{HFC-channel}[n,\text{RX}] &\leftrightarrow \text{E1 slot}[n,\text{RX}] \end{aligned}$$

with $n = 0 \ldots 31$. There is no possibility to change this allocation, so there are no registers for programming the E1 slot assigner.

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

[5] These channels have nothing to do with the mentioned D-channel and B-channels of the E1 interface, please refer to the inner circle of Figure 3.2.

If S/T-channels are coded as

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

it is possible to calculate

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

For a given HFC-channel number the belonging S/T-channel is calculated with [6]

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

In both cases the equivalence

$$\text{HFC-channel direction} = \text{S/T-channel direction}$$

is valid.

3.4 Data flow modes

The internal operation of the channel assigner and the subchannel processor depends on the selected data flow mode. The three available modes

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

are described in this section.

3.4.1 Simple Mode

In *Simple Mode* (SM) only one-to-one connections are possible. That means one FIFO, one E1 time slot or one PCM time slot can be connected to each other. All combinations except the FIFO-to-FIFO connection are possible. The number of connections is limited by the number of FIFOs. It is possible to establish as many connections as there are FIFOs [7]. The actual number of FIFOs depends on the FIFO setup (see Section 4.2).

Simple Mode is selected with V_CSM_MD = V_FSM_MD = 0 in the register R_FIFO_MD.

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

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

The FIFO number is always the same as the HFC-channel number whereas the PCM time slot number can be chosen independently from the HFC-channel number.

Due to the fixed correspondence between FIFO number and HFC-channel, a pair of transmit and receive FIFOs is allocated even if a bidirectional data connection between the PCM interface and the E1 interface is established. Please note that in this case the FIFO must be enabled to enable the data transmission.

A direct coupling of two PCM time slots uses a PCM switching buffer. This connection requires a HFC-channel number (resp. the same FIFO number). An arbitrary HFC-channel number can be chosen. If there are less than 31 transmit and receive FIFOs it is usefull to chose a HFC-channel number that is greater than the maximum FIFO number generally. This saves FIFO resources where no data is stored in a FIFO.

Subchannel processing

In most applications the subchannel processor is not used in *Simple Mode*. However, if the data stream of a FIFO does not require full 8 kByte/s data rate, the subchannel processor might be used. Unused bits can be masked out with an arbitrary mask byte.

In transparent mode only the non-masked bits of a byte are transmitted. 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 113 for details concerning the subchannel processor.

Example for SM

Figure 3.5 shows an example with three bidirectional connections (FIFO-to-E1, FIFO-to-PCM and PCM-to-E1). The FIFO box on the left side contains number and direction of the used FIFOs. The E1 and PCM boxes on the right side contain the E1 time slots and PCM time slot numbers and directions which are used in this example. Black lines illustrate data paths, whereas dotted lines symbolize blocked resources. These are not used for data transmission, but they are necessary to enable the settings.

> ☝ **Please note !**
>
> All settings in Figure 3.5 are configured in bidirectional data paths due to typical applications of the HFC-E1. 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 connection. All involved FIFOs have to be enabled with V_HDLC_TRP + V_TRP_IRQ $\neq 0$ in the register A_CON_HDLC[FIFO]. The non-specified bitmap values depend on the desired FIFO configuration.

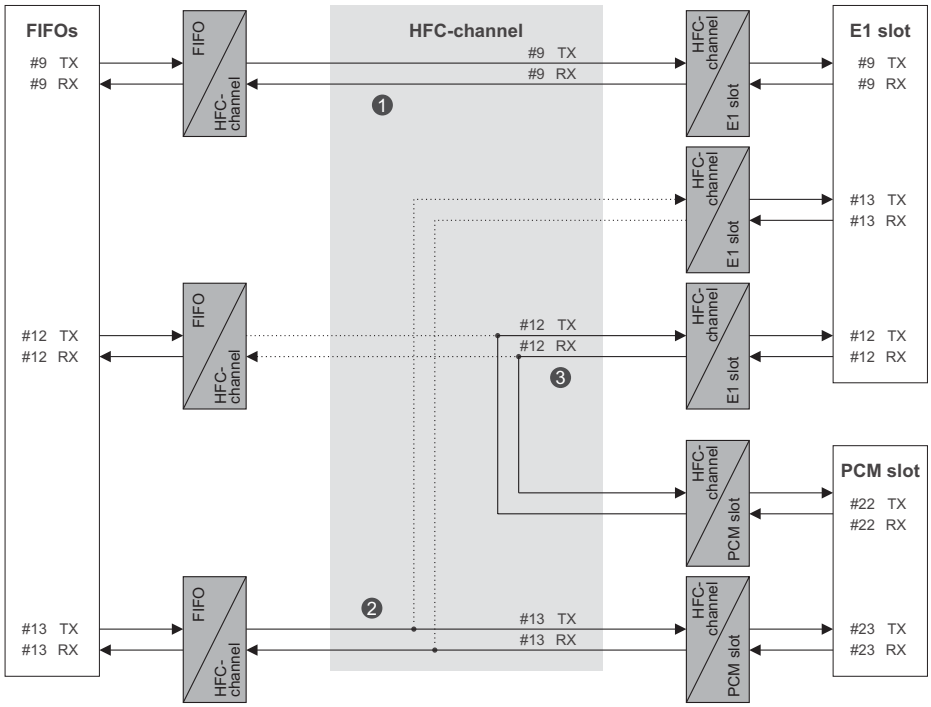

Figure 3.5: *SM example*

❶ FIFO-to-E1

As HFC-channel and FIFO numbers are the same, a selected E1 time slot specifies the corresponding FIFO (and same in inverse, of course). There is no need of programming this assigner.

R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	9	(FIFO #9)
A_CON_HDLC[9,TX]	: V_DATA_FLOW	=	'100'	FIFO → E1
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	9	(FIFO #9)
A_CON_HDLC[9,RX]	: V_DATA_FLOW	=	'100'	FIFO ← E1

❷ FIFO-to-PCM

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

R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	13	(FIFO #13)
A_CON_HDLC[13,TX]	: V_DATA_FLOW	=	'011'	(FIFO → PCM)
R_SLOT	: V_SL_DIR	=	0	(transmit slot)
	: V_SL_NUM	=	23	(slot #23)
A_SL_CFG[23,TX]	: V_CH_DIR1	=	0	(transmit HFC-channel)
	: V_CH_NUM1	=	13	(HFC-channel #13)

R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	13	(FIFO #13)
A_CON_HDLC[13,RX]	: V_DATA_FLOW	=	'001'	(FIFO ← PCM)
R_SLOT	: V_SL_DIR	=	1	(receive slot)
	: V_SL_NUM	=	23	(slot #23)
A_SL_CFG[23,RX]	: V_CH_DIR1	=	1	(receive HFC-channel)
	: V_CH_NUM1	=	13	(HFC-channel #13)

❸ **PCM-to-E1**

A direct PCM-to-E1 coupling is shown in the last connection set. FIFO[12,TX] and FIFO[12,RX] contain the data flow settings, so they must be configured and enabled to switch on the data transmission.

R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	12	(FIFO #12)
A_CON_HDLC[12,TX]	: V_DATA_FLOW	=	'110'	(E1 → PCM)
R_SLOT	: V_SL_DIR	=	0	(transmit slot)
	: V_SL_NUM	=	22	(slot #22)
A_SL_CFG[22,TX]	: V_CH_DIR1	=	1	(receive HFC-channel)
	: V_CH_NUM1	=	12	(HFC-channel #12)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	12	(FIFO #12)
A_CON_HDLC[12,RX]	: V_DATA_FLOW	=	'110'	(E1 ← PCM)
R_SLOT	: V_SL_DIR	=	1	(receive slot)
	: V_SL_NUM	=	22	(slot #22)
A_SL_CFG[22,RX]	: V_CH_DIR1	=	0	(transmit HFC-channel)
	: V_CH_NUM1	=	12	(HFC-channel #12)

Rule

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

3.4.2 Channel Select Mode

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 allocate only one 8 kByte/s E1 time slot or PCM time slot with multiple data streams with lower data rate of the assigned FIFOs. In this case the subchannel processor is involved.

The *Channel Select Mode* is selected with V_CSM_MD $= 1$ and V_FSM_MD $= 0$ in the register R_FIFO_MD.

Figure 3.6: *Channel assigner in CSM*

Channel assigner

The connection between a FIFO and a HFC-channel can be established by the A_CHANNEL register for each FIFO. For a specified FIFO, the HFC-channel to be connected must be written to V_CH_NUM0. Typically, the data direction in V_CH_DIR0 is the same as the FIFO data direction V_FIFO_DIR in the register R_FIFO. With the register settings

$$\begin{aligned} \text{A\_CHANNEL} : \text{V\_CH\_DIR0[FIFO]} &= \text{V\_FIFO\_DIR} \\ : \text{V\_CH\_NUM0[FIFO]} &= n \end{aligned}$$

the channel assigner connects the nominated FIFO to HFC-channel n.

A direct connection between a PCM time slot and an E1 time slot 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. If there are less than 31 FIFOs in transmit and receive direction, it is necessary to select an existing FIFO number.

Subchannel Processing

If more than one FIFO is to be connected to one HFC-channel, this HFC-channel number must be written into the V_CH_NUM0 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 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 113 for details concerning the subchannel processor.

Example for CSM

The example of a *Channel Select Mode* configuration in Figure 3.7 shows four bidirectional connections (FIFO-to-E1, FIFO-to-PCM, PCM-to-E1 and multiple FIFOs to E1). 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.

The following settings demonstrate only the required register values to establish the connections. All involved FIFOs have to be enabled with V_HDLC_TRP + V_TRP_IRQ $\neq 0$ in the register A_CON_HDLC[FIFO]. The non-specified bitmap values depend on the desired FIFO configuration.

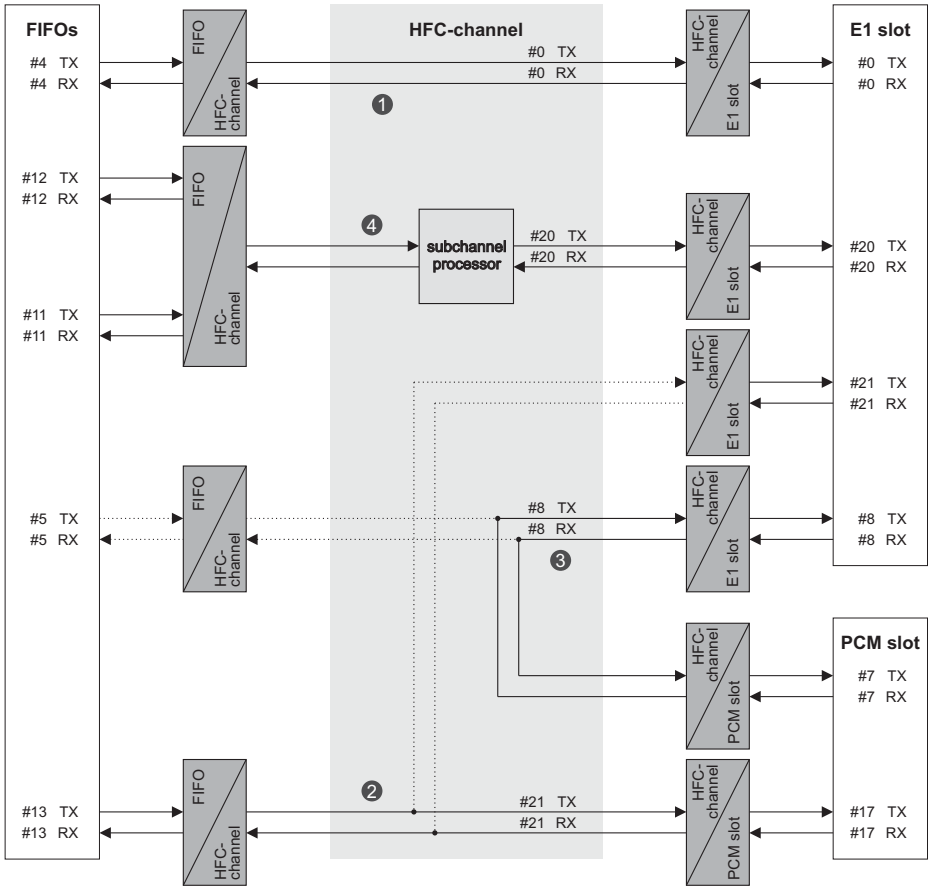

Figure 3.7: *CSM example*

❶ FIFO-to-E1

HFC-channel and FIFO numbers can be chosen independently from each other. This is shown with the FIFO-to-E1 connection:

R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	4	(FIFO #4)
A_CON_HDLC[4,TX]	: V_DATA_FLOW	=	'100'	(FIFO → E1)
A_CHANNEL[4,TX]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	0	(HFC-channel #0)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	4	(FIFO #4)
A_CON_HDLC[4,RX]	: V_DATA_FLOW	=	'100'	(FIFO ← E1)
A_CHANNEL[4,RX]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	0	(HFC-channel #0)

❷ FIFO-to-PCM

The FIFO-to-PCM connection blocks two E1 time slots and it requires two slot configuration settings:

R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	13	(FIFO #13)
A_CON_HDLC[13,TX]	: V_DATA_FLOW	=	'011'	(FIFO → PCM)
A_CHANNEL[13,TX]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	21	(HFC-channel #21)
R_SLOT	: V_SL_DIR	=	0	(transmit slot)
	: V_SL_NUM	=	17	(slot #17)
A_SL_CFG[17,TX]	: V_CH_DIR1	=	0	(transmit HFC-channel)
	: V_CH_NUM1	=	21	(HFC-channel #21)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	13	(FIFO #13)
A_CON_HDLC[13,RX]	: V_DATA_FLOW	=	'001'	(FIFO ← PCM)
A_CHANNEL[13,RX]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	21	(HFC-channel #21)
R_SLOT	: V_SL_DIR	=	1	(receive slot)
	: V_SL_NUM	=	17	(slot #17)
A_SL_CFG[17,RX]	: V_CH_DIR1	=	1	(receive HFC-channel)
	: V_CH_NUM1	=	21	(HFC-channel #21)

❸ PCM-to-E1

The PCM-to-E1 connection blocks two FIFOs[8]. Although there is no data stored in these FIFOs, they must be enabled to switch on the data transmission between the PCM and the E1 interface.

R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	5	(FIFO #5)
A_CON_HDLC[5,TX]	: V_DATA_FLOW	=	'110'	(PCM ← E1)
A_CHANNEL[5,TX]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	8	(HFC-channel #8)
R_SLOT	: V_SL_DIR	=	0	(transmit slot)
	: V_SL_NUM	=	7	(slot #7)
A_SL_CFG[7,TX]	: V_CH_DIR1	=	1	(receive HFC-channel)
	: V_CH_NUM1	=	8	(HFC-channel #8)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	5	(FIFO #5)
A_CON_HDLC[5,RX]	: V_DATA_FLOW	=	'110'	(PCM → E1)
A_CHANNEL[5,RX]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	8	(HFC-channel #8)
R_SLOT	: V_SL_DIR	=	1	(receive slot)
	: V_SL_NUM	=	7	(slot #7)
A_SL_CFG[7,RX]	: V_CH_DIR1	=	0	(transmit HFC-channel)
	: V_CH_NUM1	=	8	(HFC-channel #8)

❹ multiple FIFOs to E1

Finally, the bidirectional connection between two FIFOs and one E1 time slot completes the example.

[8] Hint: Here it is possible to occupy HFC-channels that are assigned to E-channels (HFC-channel[3, 7, 11, . . . , 31]) because these are normally not used.

R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	12	(FIFO #12)
A_CON_HDLC[12,TX]	: V_DATA_FLOW	=	'100'	(FIFO → E1)
A_CHANNEL[12,TX]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	20	(HFC-channel #20)
R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	11	(FIFO #11)
A_CON_HDLC[11,TX]	: V_DATA_FLOW	=	'100'	(FIFO → E1)
A_CHANNEL[11,TX]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	20	(HFC-channel #20)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	11	(FIFO #11)
A_CON_HDLC[11,RX]	: V_DATA_FLOW	=	'100'	(FIFO ← E1)
A_CHANNEL[11,RX]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	20	(HFC-channel #20)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	12	(FIFO #12)
A_CON_HDLC[12,RX]	: V_DATA_FLOW	=	'100'	(FIFO ← E1)
A_CHANNEL[12,RX]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	20	(HFC-channel #20)

In addition to the above register settings, the subchannel processor must be configured now. It is important to see that the subchannel processor programming has no influence to the connection setup. So there is no need to describe these settings here. Please see Section 3.5 on page 113 for a detailed subchannel description.

Rule

In *Channel Select Mode*

- every HFC-channel used requires at least one enabled FIFO (except for the PCM-to-PCM connection) with the same data direction and
- every PCM time slot used requires one HFC-channel (except for the PCM-to-PCM connection where a full duplex connection allocates one HFC-channel).

3.4.3 FIFO Sequence Mode

In contrast to the PCM and E1 time slots, the FIFO data rate is not fixed to 8 kByte/s. In the previous section the CSM allows the functional capability of a FIFO data rate less than 8 kByte/s. In this section, the third data flow mode shows how to use FIFOs with a higher data rate with the *FIFO Sequence Mode* (FSM). In transmit direction one FIFO can cyclically distribute its data to several HFC-channels. In opposite direction, received data from several HFC-channels can be collected cyclically in one FIFO (see Fig. 3.8, right side). A one-to-one connection between FIFO and HFC-channel is of course possible in FSM, too (Fig. 3.8, left side).

Figure 3.8: *FIFO / channel assigner*

FIFO Sequence Mode is selected with V_FSM_MD = '1' in the register R_FIFO_MD). CSM and FSM should be used at the same time. Actually, this is necessary for nearly all FSM applications. The HFC-E1 works in *Simple Mode* if none of these two modes is selected.

FIFO sequence

To achieve a FIFO data rate higher than 8 kByte/s a FIFO must be connected to more than one HFC-channel. As there is only one register A_CHANNEL[FIFO] the FSM programming path must differ from the previous modes.

In FSM all FIFOs are organized in a list with up to 64 entries. Every list entry is assigned to a FIFO. FIFO configuration can be set up as usual. I.e. HFC-channel allocation, flow controller programming and subchannel processing can be configured as described in the previous sections. Additionally, each list entry specifies the next FIFO of the sequence. The list is terminated by an 'end of list' entry. This procedure is shown in Figure 3.9 with $j + 1$ list entries.

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

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

Important !

FIFO data rates higher than 8 kByte/s require an arbitrary assignment between a FIFO number and the connected HFC-channel. Therefore, the *Channel Select Mode* must be enabled. For this reason FSM is mostly selected in combination with CSM. All data transfer configuration possible with FSM but without CSM are also possible with CSM only – but with lower configuration effort!

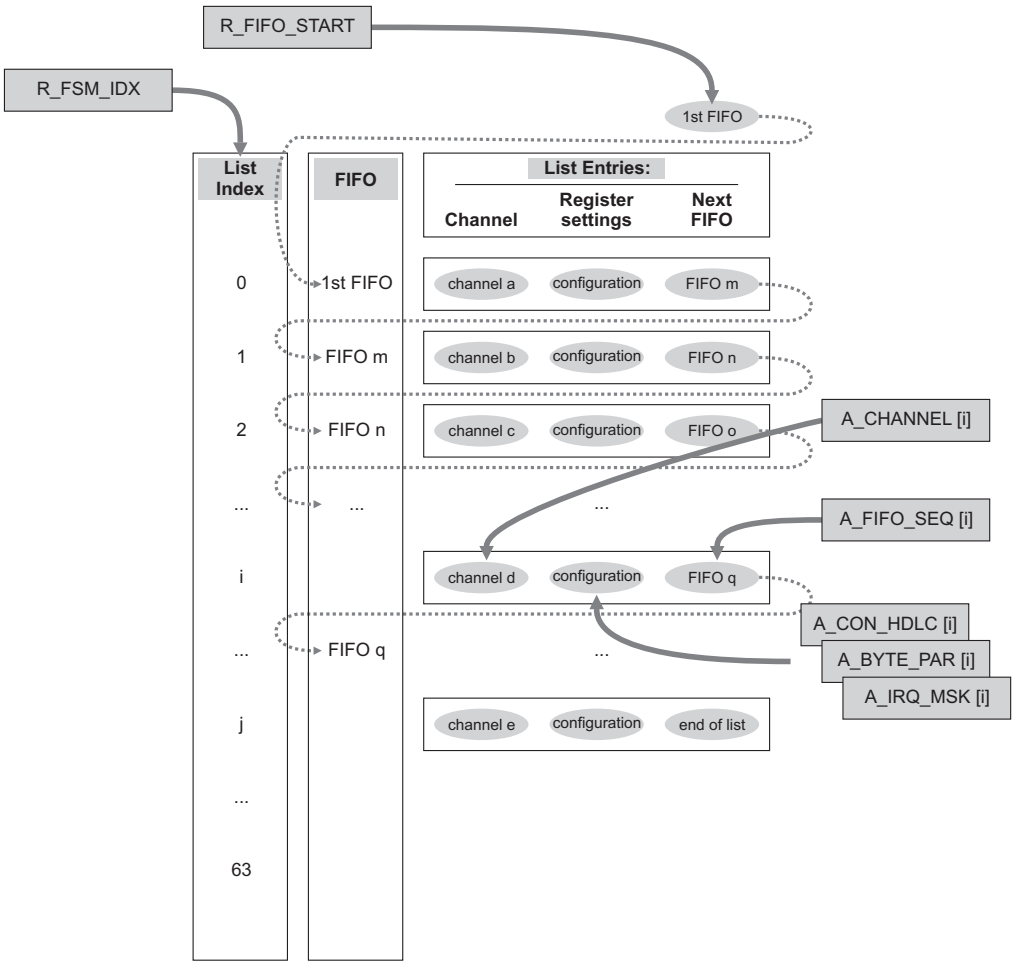

Figure 3.9: *FSM list processing*

FSM programming

The list index register R_FSM_IDX specifies the list index with bitmap V_IDX in the range of $0 \ldots 63$. R_FSM_IDX has the same address as R_FIFO because in FSM it replaces R_FIFO for list programming. So all array registers indexed with [FIFO] are indexed with the V_IDX value instead.

The first FIFO of the list has to be specified in the register R_FIRST_FIFO with the direction bit V_FIRST_FIFO_DIR and the FIFO number V_FIRST_FIFO_NUM. The next FIFO has to be specified in the register A_FIFO_SEQ. Referring to Figure 3.9 the array registers of the list entry $i + 1$ are assigned to FIFO q because 'next FIFO' entry at list index i is 'FIFO q'.

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

The connected HFC-channel and the FIFO configuration must be programmed in the same way as in CSM. These settings belong to the FIFO which is specified in the previous list entry under 'next FIFO' (or the R_FIRST_FIFO register for the first list entry).

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

Example for FSM

Figure 3.10 shows an example with three bidirectional connections. The black lines illustrate data paths, whereas the dotted lines symbolize blocked HFC-channels. These are not used for data transmission, but they are necessary to enable the settings.

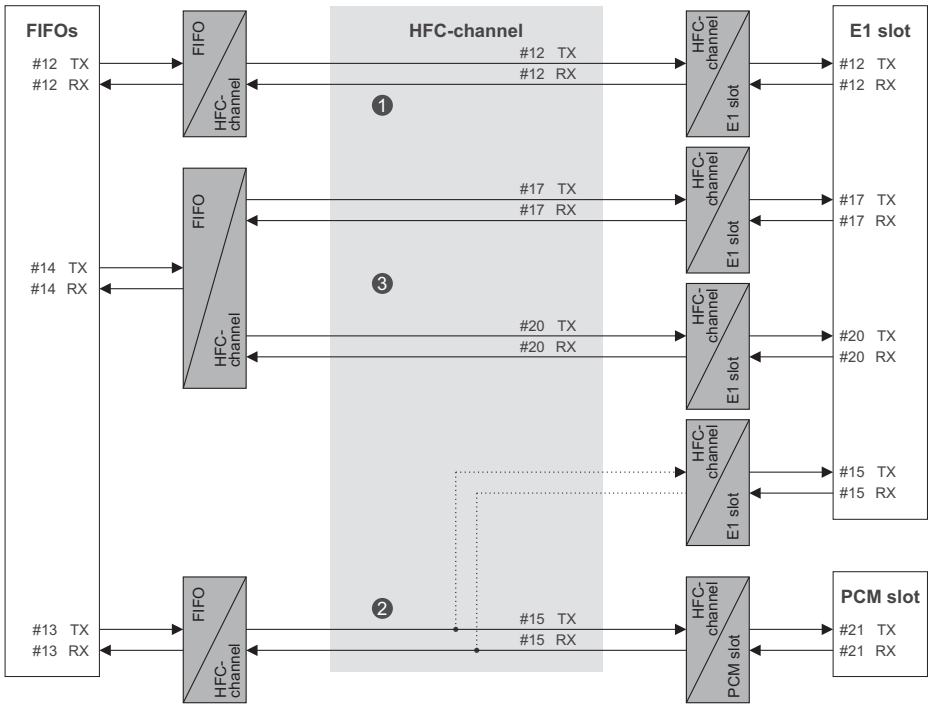

Figure 3.10: *FSM example*

All FIFOs can be arranged in arbitrary order. In the example the list specification of Table 3.4 is chosen. To select FIFO[12,TX] as first FIFO R_FIRST_FIFO is set as follows:

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

❶ **FIFO-to-E1**

The bidirectional FIFO-to-E1 connection allocates the list indices 0 and 1 as follows:

Table 3.4: *List specification of the example in Figure 3.10*

List index	Connection		
0	FIFO[12,TX]	→	E1 slot[12,TX]
1	FIFO[12,RX]	←	E1 slot[12,TX]
2	FIFO[13,RX]	←	PCM slot[21,RX]
3	FIFO[13,TX]	→	PCM slot[21,TX]
4	FIFO[14,TX]	→	E1 slot[17,TX]
5	FIFO[14,RX]	←	E1 slot[17,RX]
6	FIFO[14,TX]	→	E1 slot[20,TX]
7	FIFO[14,RX]	←	E1 slot[20,RX]

R_FSM_IDX	: V_IDX	=	0	(list index 0, FIFO[12,TX])
A_CON_HDLC[0]	: V_DATA_FLOW	=	'100'	(FIFO → E1)
A_CHANNEL[0]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	12	(HFC-channel #12)
A_FIFO_SEQ[0]	: V_NEXT_FIFO_DIR	=	1	(next: receive FIFO)
	: V_NEXT_FIFO_NUM	=	12	(next: FIFO #12)
	: V_SEQ_END	=	0	(continue)
R_FSM_IDX	: V_IDX	=	1	(list index 1, FIFO[12,RX])
A_CON_HDLC[1]	: V_DATA_FLOW	=	'100'	(FIFO ← E1)
A_CHANNEL[1]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	12	(HFC-channel #12)
A_FIFO_SEQ[1]	: V_NEXT_FIFO_DIR	=	1	(next: receive FIFO)
	: V_NEXT_FIFO_NUM	=	13	(next: FIFO #13)
	: V_SEQ_END	=	0	(continue)

❷ FIFO-to-PCM

The following two list entries (indices 2 and 3) define the bidirectional FIFO-to-PCM connections. Two E1 time slots are blocked. But E1 time slot resources are saved because HFC-channels that are assigned to not used E-channels are selected.

R_FSM_IDX	: V_IDX	=	2	(list index 2, FIFO[13,RX])
A_CON_HDLC[2]	: V_DATA_FLOW	=	'011'	(FIFO ← PCM)
A_CHANNEL[2]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	15	(HFC-channel #15)
R_SLOT	: V_SL_DIR	=	1	(receive slot)
	: V_SL_NUM	=	21	(slot #21)
A_SL_CFG[21,RX]	: V_CH_DIR1	=	1	(receive HFC-channel)
	: V_CH_NUM1	=	15	(HFC-channel #15)
A_FIFO_SEQ[2]	: V_NEXT_FIFO_DIR	=	0	(next: transmit FIFO)
	: V_NEXT_FIFO_NUM	=	13	(next: FIFO #13)
	: V_SEQ_END	=	0	(continue)

R_FSM_IDX	: V_IDX	=	3	(list index 3, FIFO[13,TX])
A_CON_HDLC[3]	: V_DATA_FLOW	=	'011'	(FIFO → PCM)
A_CHANNEL[3]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	15	(HFC-channel #15)
R_SLOT	: V_SL_DIR	=	0	(transmit slot)
	: V_SL_NUM	=	21	(slot #21)
A_SL_CFG[21,TX]	: V_CH_DIR1	=	0	(transmit HFC-channel)
	: V_CH_NUM1	=	15	(HFC-channel #15)
A_FIFO_SEQ[32]	: V_NEXT_FIFO_DIR	=	0	(next: transmit FIFO)
	: V_NEXT_FIFO_NUM	=	14	(next: FIFO #14)
	: V_SEQ_END	=	0	(continue)

❸ FIFO to multiple E1 time slots

The last settings connect one FIFO with two E1 time slots in transmit and in receive direction. So both FIFOs have a data rate of 16 kByte/s.

R_FSM_IDX	: V_IDX	=	4	(list index 4, FIFO[14,TX])
A_CON_HDLC[4]	: V_DATA_FLOW	=	'100'	(FIFO → E1)
A_CHANNEL[4]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	17	(HFC-channel #17)
A_FIFO_SEQ[4]	: V_NEXT_FIFO_DIR	=	1	(next: receive FIFO)
	: V_NEXT_FIFO_NUM	=	14	(next: FIFO #18)
	: V_SEQ_END	=	0	(continue)
R_FSM_IDX	: V_IDX	=	5	(list index 5, FIFO[14,RX])
A_CON_HDLC[5]	: V_DATA_FLOW	=	'100'	(FIFO → E1)
A_CHANNEL[5]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	17	(HFC-channel #17)
A_FIFO_SEQ[5]	: V_NEXT_FIFO_DIR	=	0	(next: transmit FIFO)
	: V_NEXT_FIFO_NUM	=	14	(next: FIFO #14)
	: V_SEQ_END	=	0	(continue)
R_FSM_IDX	: V_IDX	=	6	(list index 6, FIFO[14,TX])
A_CON_HDLC[6]	: V_DATA_FLOW	=	'100'	(FIFO ← E1)
A_CHANNEL[6]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	20	(HFC-channel #20)
A_FIFO_SEQ[6]	: V_NEXT_FIFO_DIR	=	1	(next: receive FIFO)
	: V_NEXT_FIFO_NUM	=	14	(next: FIFO #14)
	: V_SEQ_END	=	0	(continue)
R_FSM_IDX	: V_IDX	=	7	(list index 7, FIFO[14,RX])
A_CON_HDLC[7]	: V_DATA_FLOW	=	'100'	(FIFO ← E1)
A_CHANNEL[7]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	20	(HFC-channel #20)
A_FIFO_SEQ[7]	: V_SEQ_END	=	1	(end of chain)

3.5 Subchannel Processing

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 (*Channel Select Mode* enabled / disabled) 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.

A general overview of the subchannel processor in transmit direction is given in Figure 3.11. It shows an example with three FIFOs connected to one HFC-channel. Details of subchannel processing are described in the following sections, categorized into the different modes of the data flow and the HDLC controller.

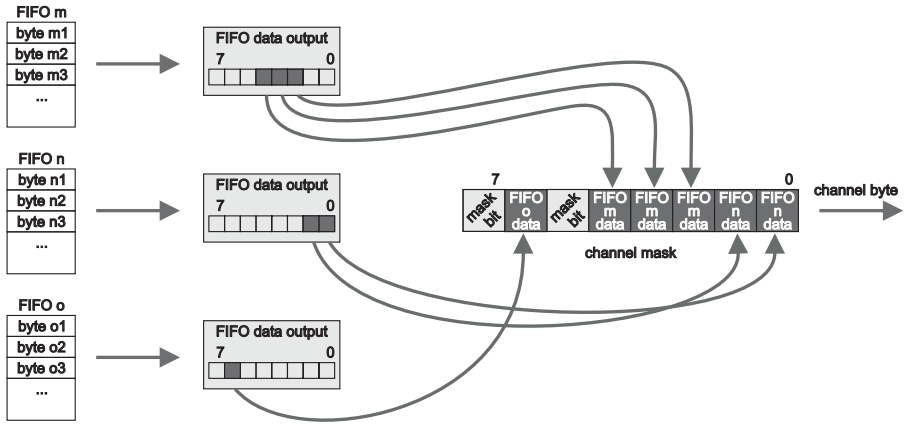

Figure 3.11: *General structure of the subchannel processor shown with an example of three connected FIFOs*

The essence of the subchannel processor is a bit extraction (transmit) respectively insertion (receive) unit for every FIFO and a byte mask for every HFC-channel. The subchannel parameters V_BIT_CNT and V_START_BIT of the register A_SUBCH_CFG define the bits of the HFC-channel byte that are claimed by the FIFO. On the other side, the channel mask defines the bit values of those HFC-channel data bits, that are not occupied by FIFO data.

Registers

The FIFO bit extraction / insertion requires two register settings. V_BIT_CNT defines the number of bits to be extracted / inserted. The start bit can be selected with V_START_BIT in the range of 0 . . . 7. Both values are located in the register A_SUBCH_CFG[FIFO].

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

> **Important !**
>
> Typically, the R_FIFO register contains always an FIFO index. There is one exception where the R_FIFO value has a different meaning: The HFC-channel mask byte is programmed by writing the HFC-channel into the R_FIFO register.

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

> **Important !**
>
> The A_CH_MSK array register is indexed by R_FIFO to write the mask byte. However the mask is assigned to a HFC-channel, namely that HFC-channel which is assigned to the indexing FIFO.

3.5.1 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 subchannel processor takes only the bits that are defined by the FIFO parameters and inserts them into the channel mask A_CH_MSK.

Received HFC-channel data bytes are stored completely in the FIFO and are independently from the V_BIT_CNT and V_START_BIT settings.

Simple Mode

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

Suppose FIFO[m,TX] has the register A_SUBCH_CFG settings V_BIT_CNT $= 3$ and V_START_BIT $= 2$ (see Fig. 3.11). Further, the channel mask is defined as A_CH_MSK $= [M_7 \ldots M_0]$. Then the FIFO[m,TX] data bytes $m1 \ldots mi$ with bit index $0 \ldots 7$ build up the HFC-channel data bytes as shown in Table 3.5. From every FIFO byte only three bits are transmitted to the HFC-channel. These bits are accentuated in the table. The other bits are defined by the channel mask.

In receive direction, the subchannel processor has no effect in *Simple mode* combined with transparent mode. So received HFC-channel bytes are stored in the FIFO without changing.

Channel Select Mode

In *Channel Select Mode* it is possible to connect more than one FIFO to a HFC-channel. The configuration in Figure 3.11 with three FIFOs can be taken as example. The bit extraction / insertion units must be configured with the following register settings:

Table 3.5: *Subchannel processing example in SM combined with transparent mode (transmit direction)*

	7							0
channel mask:	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0
HFC-channel transmit byte 1:	M_7	M_6	M_5	$m1_4$	$m1_3$	$m1_2$	M_1	M_0
HFC-channel transmit byte 2:	M_7	M_6	M_5	$m2_4$	$m2_3$	$m2_2$	M_1	M_0
HFC-channel transmit byte 3:	M_7	M_6	M_5	$m3_4$	$m3_3$	$m3_2$	M_1	M_0
...					...			

A_SUBCH_CFG[m,TX] :	V_BIT_CNT	=	3	(3 bits)
:	V_START_BIT	=	2	(beginning at bit 2)
A_SUBCH_CFG[n,TX] :	V_BIT_CNT	=	2	(2 bits)
:	V_START_BIT	=	0	(beginning at bit 0)
A_SUBCH_CFG[o,TX] :	V_BIT_CNT	=	1	(1 bit)
:	V_START_BIT	=	6	(bit 6)

Each FIFO occupies one or more bits in a HFC-channel data byte. In this example 2 bits are not used for data. They are filled with the channel mask bits M_7 and M_5. Table 3.6 shows the HFC-channel data bytes which are constructed from three FIFOs.

Table 3.6: *Subchannel processing example in CSM combined with transparent mode (transmit direction)*

	7							0
channel mask:	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0
HFC-channel transmit byte 1:	M_7	$o1_6$	M_5	$m1_4$	$m1_3$	$m1_2$	$n1_1$	$n1_0$
HFC-channel transmit byte 2:	M_7	$o2_6$	M_5	$m2_4$	$m2_3$	$m2_2$	$n2_1$	$n2_0$
HFC-channel transmit byte 3:	M_7	$o3_6$	M_5	$m3_4$	$m3_3$	$m3_2$	$n3_1$	$n3_0$
...					...			

In the opposite data direction the incoming HFC-channel bytes are stored unchanged in all connected FIFOs. Therefore it is unnecessary to connect more than one receive FIFO to a receive HFC-channel if CSM and transparent mode are selected.

3.5.2 HDLC mode

HDLC mode allows to reduce the data rate of a FIFO. In the example of Figure 3.11 FIFO[m,TX] delivers 3 bits every 125 μs which leads to a FIFO data rate of e.g. 3 kByte/s.

With V_BIT_CNT $= x$, the first x bits of a FIFO byte are transferred to the connected HFC-

channel during the first 125 μs cycle. During the next 125 μs cycle the next x bits of the same byte are processed, and so on. When 8 FIFO bits are processed, the next FIFO byte is processed. The byte boundaries are neglected.

Simple Mode

HDLC mode combined with *Simple Mode* can transmit one FIFO bit stream (e.g. of FIFO[m,TX]) to the connected HFC-channel. The result is given in Table 3.7 [9].

Table 3.7: *Subchannel processing example in SM combined with HDLC mode (transmit direction)*

	7							0
channel mask:	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0
HFC-channel transmit byte 1:	M_7	M_6	M_5	$m1_2$	$m1_1$	$m1_0$	M_1	M_0
HFC-channel transmit byte 2:	M_7	M_6	M_5	$m1_5$	$m1_4$	$m1_3$	M_1	M_0
HFC-channel transmit byte 3:	M_7	M_6	M_5	$m2_0$	$m1_7$	$m1_6$	M_1	M_0
HFC-channel transmit byte 4:	M_7	M_6	M_5	$m2_3$	$m2_2$	$m2_1$	M_1	M_0
...					...			

Received HFC-channel data are processed similar. FIFO[m,RX] with the setting

A_SUBCH_CFG[m,RX] :	V_BIT_CNT	= 3	(3 bits)
:	V_START_BIT	= 2	(beginning at bit 2)

stores 3 bits every 125 μs cycle. These bits are taken from the connected HFC-channel at position [4 ... 2].

Channel Select Mode

In *Channel Select Mode* several FIFOs can transmit a bit stream to one connected HFC-channel. Figure 3.11 with three connected FIFOs to HFC-channel[a,TX] is taken again as an example. HFC-channel transmit data for this configuration is shown in Table 3.8 [10].

Received HFC-channel data are processed similary. Assuming that three receive FIFOs are configured with the same settings as their corresponding transmit FIFOs, then FIFO[m,RX] receives a bit stream with 3 kByte/s, FIFO[n,RX] receives 2 kByte/s and FIFO[o,RX] receives 1 kByte/s.

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

Table 3.8: *Subchannel processing example in CSM combined with HDLC mode (transmit direction)*

	7							0
channel mask:	M_7	M_6	M_5	M_4	M_3	M_2	M_1	M_0
HFC-channel transmit byte 1:	M_7	$o1_0$	M_5	$m1_2$	$m1_1$	$m1_0$	$n1_1$	$n1_0$
HFC-channel transmit byte 2:	M_7	$o1_1$	M_5	$m1_5$	$m1_4$	$m1_3$	$n1_3$	$n1_2$
HFC-channel transmit byte 3:	M_7	$o1_2$	M_5	$m2_0$	$m1_7$	$m1_6$	$n1_5$	$n1_4$
HFC-channel transmit byte 4:	M_7	$o1_3$	M_5	$m2_3$	$m2_2$	$m2_1$	$n1_7$	$n1_6$
HFC-channel transmit byte 5:	M_7	$o1_4$	M_5	$m2_6$	$m2_5$	$m2_4$	$n2_1$	$n2_0$
...					...			

3.6 Register description

R_FIRST_FIFO	(write only)	0x0B

First FIFO of the FIFO sequence

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

Bits	Reset Value	Name	Description
0	0	V_FIRST_FIFO_DIR	**Data direction** This bit defines the data direction of the first FIFO in FIFO sequence. '0' = transmit FIFO data '1' = receive FIFO data
5..1	0x00	V_FIRST_FIFO_NUM	**FIFO number** This bitmap defines the number of the first FIFO in FIFO sequence.
7..6		(reserved)	Must be '00'.

R_FIFO_MD	(write only)	0x0D

FIFO mode configuration

Bits	Reset Value	Name	Description
1..0	0	**V_FIFO_MD**	**FIFO mode** This bitmap and V_FIFO_SZ are used to organize the FIFOs in the internal or external SRAM.
2	0	**V_CSM_MD**	***Channel select mode* (CSM)** '0' = disable CSM (FIFO number = HFC-channel number) '1' = enable CSM **Note:** The HFC-E1 works in *Simple Mode* (SM) if CSM and FSM are both disabled.
3	0	**V_FSM_MD**	***FIFO sequence mode* (FSM)** '0' = disable FSM '1' = enable FSM **Note:** In most cases where FSM is selected, also CSM should be enabled.
5..4	0	**V_FIFO_SZ**	**FIFO size** This bitmap and V_FIFO_MD are used to organize the FIFOs in the internal or external SRAM. The actual FIFO sizes depend on the used SRAM size.
7..6		**(reserved)**	Must be '00'.

(See Table 4.3 on page 132 for suitable V_FIFO_MD and V_FIFO_SZ values.)

R_FIFO	(write only)	0x0F

FIFO selection register

This multi-register is selected with bitmap V_FSM_MD = 0 of the register R_FIFO_MD. It is only used in SM and CSM.

Bits	Reset Value	Name	Description
0	0	**V_FIFO_DIR**	**FIFO data direction** '0' = transmit FIFO data '1' = receive FIFO data
5..1	0x00	**V_FIFO_NUM**	**FIFO number**
6		**(reserved)**	Must be '0'.
7	0	**V_REV**	**Bit order** '0' = normal bit order '1' = reversed bit order Normal bit order means LSB first in HDLC mode and MSB first in transparent mode. The bit order is being reversed for the data stored into the FIFO or when the data is read from the FIFO.

R_FSM_IDX	(write only)	0x0F

Index register of the FIFO sequence

This multi-register is selected with bitmap V_FSM_MD = 1 of the register R_FIFO_MD. It is only used in FSM.

Bits	Reset Value	Name	Description
5..0	0	**V_IDX**	**List index** The list index must be in the range 0 . . . 63.
7..6		**(reserved)**	Must be '00'.

R_SLOT	(write only)	0x10

PCM time slot selection

The selected time slot is used for all slot depending registers. Depending on the V_PCM_DR value in the R_PCM_MD1 register 16, 32 or 64 time slots are available for each data direction.

Bits	Reset Value	Name	Description
0	0	V_SL_DIR	**PCM time slot data direction** '0' = transmit PCM data '1' = receive PCM data
7..1	0x00	V_SL_NUM	**PCM time slot number**

A_SL_CFG [SLOT]	(write only)	0xD0

HFC-channel assignment for the selected PCM time slot and PCM output buffer configuration

With this register a HFC-channel can be assigned to the selected PCM time slot. Additionally, the PCM buffers can be configured.

Before writing this array register the PCM time slot must be selected by the register R_SLOT.

Bits	Reset Value	Name	Description
0	0	**V_CH_DIR1**	**HFC-channel data direction** '0' = HFC-channel for transmit data '1' = HFC-channel for receive data
5..1	0	**V_CH_NUM1**	**HFC-channel number** (0 . . . 31)
7..6	0	**V_ROUT**	**PCM output buffer configuration** For transmit time slots: '00' = disable output buffers, no data transmision '01' = transmit data internally, output buffers disabled '10' = output buffer enable for STIO1 '11' = output buffer enable for STIO2 For receive time slots: '00' = input data is ignored '01' = loop PCM data internally '10' = data in from STIO2 '11' = data in from STIO1

(See Figure 6.1 on page 179 for detailed information).

A_CH_MSK [FIFO]	(write only)	0xF4

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

For receive FIFOs this register is ignored.

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

Bits	Reset Value	Name	Description
7..0	0	V_CH_MSK	**Mask byte** This bitmap defined 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 the register A_SUBCH_CFG.

A_CON_HDLC [FIFO]	(write only)	0xFA

HDLC and connection settings of the selected FIFO

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

Bits	Reset Value	Name	Description
0	0	V_IFF	**Inter frame fill** '0' = write HDLC flags 0x7F as inter frame fill '1' = write all '1' s as inter frame fill **Note:** For D-channel this bit must be '1'.
1	0	V_HDLC_TRP	**HDLC mode / transparent mode selection** '0' = HDLC mode '1' = transparent mode **Note:** For D-channel this bit must be '0'.
4..2	0	V_TRP_IRQ	**Transparent mode interrupt selection** An interrupt is generated all 2^n bytes when the bits [n-1:0] of the $Z1$- or $Z2$-counter become '1'. 0 = interrupt disabled 1 = all $2^6 = 64$ bytes an interrupt is generated 2 = all $2^7 = 128$ bytes an interrupt is generated 3 = all $2^8 = 256$ bytes an interrupt is generated 4 = all $2^9 = 512$ bytes an interrupt is generated 5 = all $2^{10} = 1024$ bytes an interrupt is generated 6 = all $2^{11} = 2048$ bytes an interrupt is generated 7 = all $2^{12} = 4096$ bytes an interrupt is generated **Note:** No interrupt occurs, if the Z-counters do never reach the selected values. This depends on the Z_{MAX} setting.
7..5	0	V_DATA_FLOW	**Data flow configuration** 0 = FIFO ↔ E1, FIFO → PCM 1 = FIFO ↔ PCM, FIFO → E1 2 = FIFO → PCM, E1 → FIFO, PCM → E1 3 = FIFO ↔ PCM, PCM → E1 4 = FIFO ↔ E1, E1 → PCM 5 = FIFO → E1, E1 → PCM, PCM → FIFO 6 = E1 ↔ PCM, E1 → FIFO 7 = E1 ↔ PCM, PCM → FIFO

(For details on bitmap V_DATA_FLOW see Fig. 3.3 and 3.4 on page 98.)

> ☝ **Important !**
>
> A FIFO is disabled if V_HDLC_TRP + V_TRP_IRQ = 0 in the 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_SUBCH_CFG [FIFO] **(write only)** 0xFB

Subchannel parameters for bit processing of the selected FIFO

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

Note: For D-channel this register must be 0x02.

Bits	Reset Value	Name	Description
2..0	0	**V_BIT_CNT**	**Bit counter for HDLC and transparent mode** This bitmap contains the number of bits to be processed. '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 for HDLC and transparent mode** '000' = start processing with bit 0 '001' = start processing with bit 1 '010' = start processing with bit 2 '011' = start processing with bit 3 '100' = start processing with bit 4 '101' = start processing with bit 5 '110' = start processing with bit 6 '111' = start processing with bit 7
6	0	**V_LOOP_FIFO**	**FIFO loop** '0' = normal operation '1' = repeat current frame (in transparent mode only)
7	0	**V_INV_DATA**	**Inverted data** '0' = normal data out '1' = inverted data out

A_CHANNEL [FIFO]	(write only)	0xFC

HFC-channel assignment for the selected FIFO

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

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

Bits	Reset Value	Name	Description
0	0	**V_CH_DIR0**	**HFC-channel data direction** '0' = HFC-channel for transmit data '1' = HFC-channel for receive data
5..1	0	**V_CH_NUM0**	**HFC-channel number** (0 … 31)
7..6	0	**(reserved)**	Must be '00'.

A_FIFO_SEQ [FIFO]	(write only)	0xFD

FIFO sequence list

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

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

Bits	Reset Value	Name	Description
0	0	**V_NEXT_FIFO_DIR**	**FIFO data direction** This bit defines the data direction of the next FIFO in FIFO sequence. '0' = transmit FIFO data '1' = receive FIFO data
5..1	0	**V_NEXT_FIFO_NUM**	**FIFO number** This bitmap defines the FIFO number of the next FIFO in FIFO sequence.
6	0	**V_SEQ_END**	**End of FIFO list** '0' = FIFO list goes on '1' = FIFO list is terminated after this FIFO (V_NEXT_FIFO_DIR and V_NEXT_FIFO_NUM are ignored)
7	0	**(reserved)**	Must be '0'.

Chapter 4

FIFO handling and HDLC controller

Table 4.1: *Overview of the HFC-E1 FIFO registers*

Write only registers:			Read only register:			Read / write registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**	**Address**	**Name**	**Page**
0x0E	R_INC_RES_FIFO	138	0x04	A_Z1L	139	0x80	A_FIFO_DATA0	143
0x0F	R_FIFO	121	0x05	A_Z1H	139	0x84	A_FIFO_DATA0_NOINC	144
0x0F	R_FSM_IDX	121	0x06	A_Z2L	140			
0xFA	A_CON_HDLC	125	0x07	A_Z2H	140			
0xFB	A_SUBCH_CFG	126	0x0C	A_F1	141			
			0x0D	A_F2	141			
			0x88	R_INT_DATA	142			

There are up to 32 receive FIFOs and up to 32 transmit FIFOs with 64 HDLC controllers in whole. The HDLC circuits are located on the E1 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 E1 interface or to the PCM bus (transmit FIFO operation). Automatic zero deletion is done in HDLC mode when the HDLC data comes from the E1 interface or PCM bus (receive FIFO operation).

There is a transmit and a receive FIFO for each E1 time slot (even for time slot 0).

The FIFO control registers are used to select and control the FIFOs of the HFC-E1. The FIFO register set exists for every FIFO number and receive / transmit direction. The FIFO is selected by the FIFO select register R_FIFO.

All FIFOs are disabled after reset (hardware reset, soft reset or HFC 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.

4.1 FIFO counters

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

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

After every pulse on the F0IO signal HDLC bytes are written into the E1 interface (from a transmit FIFO) and HDLC bytes are read from the E1 interface (to a receive FIFO).

Additionally there are two counters $F1$ and $F2$ for every FIFO for counting the HDLC frames. Their width is 4 bit for 32 kByte SRAM and 5 bit for larger SRAMs. They form a ring buffer as $Z1$ and $Z2$ do, too.

Table 4.2: *F-counter range with different RAM sizes*

RAM size	F_{MIN}	F_{MAX}
32k x 8	0x00	0x0F
128k x 8	0x00	0x1F
512k x 8	0x00	0x1F

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

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

- $Z1 = Z2 = Z_{MAX}$ [1] and
- $F1 = F2 = F_{MAX}$ [2].

This initialization can be carried out with a soft reset or a HDLC reset. For this, the bit V_SRES or the bit V_HFCRES in the register R_CIRM have to be set. Individual FIFOs can be reset with bit V_RES_F of the register R_INC_RES_FIFO.

In addition, a hardware reset initializes the counters.

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-E1. This means an access to FIFO control registers is not allowed until BUSY status is reset (bit V_BUSY of R_STATUS register). The maximum duration takes 25 clock cycles ($\sim$1 μs). Status, interrupt and control registers can be read and written at any time.

Please note !

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

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

4.2 FIFO size setup

The HFC-E1 can operate with 32k x 8 internal or alternatively with 128k x 8 or 512k x 8 external SRAM. The bitmap V_RAM_SZ of the register R_RAM_MISC must be set accordingly to the RAM size. Table 4.3 shows how the FIFO size can be varied with the different RAM sizes. Additionally, the initial Z_{max} and Z_{min} values are given in Table 4.3.

After changing the FIFO size or RAM size a soft reset should be initiated.

[1] See Z_{max} value in Table 4.3.
[2] See F_{max} value in Table 4.2.

Table 4.3: *FIFO size setup*

		32k x 8 RAM (internal) V_RAM_SZ = 0x00 F_{MIN} = 0x00, F_{MAX} = 0x0F				128k x 8 RAM (external) V_RAM_SZ = 0x01 F_{MIN} = 0x00, F_{MAX} = 0x1F				512k x 8 RAM (external) V_RAM_SZ = 0x02 F_{MIN} = 0x00, F_{MAX} = 0x1F			
V_FIFO_MD	V_FIFO_SZ	FIFO number	Z_{MIN}	Z_{MAX}	FIFO size (byte)	FIFO number	Z_{MIN}	Z_{MAX}	FIFO size (byte)	FIFO number	Z_{MIN}	Z_{MAX}	FIFO size (byte)
’00’	’00’	0 … 31	0x80	0x1FF	384	0 … 31	0xC0	0x07FF	1856	0 … 31	0xC0	0x1FFF	8000
’10’	’00’	0 … 15	0x80	0x0FF	128	0 … 15	0xC0	0x03FF	832	0 … 15	0xC0	0x0FFF	3904
		16 … 31	0x00	0x1FF	512	16 … 31	0x00	0x07FF	2048	16 … 31	0x00	0x1FFF	8192
’10’	’01’	0 … 23	0x80	0x0FF	128	0 … 23	0xC0	0x03FF	832	0 … 23	0xC0	0x0FFF	3904
		24 … 31	0x00	0x3FF	1024	24 … 31	0x00	0x0FFF	4096	24 … 31	0x00	0x3FFF	16384
’10’	’10’	0 … 27	0x80	0x0FF	128	0 … 27	0xC0	0x03FF	832	0 … 27	0xC0	0x0FFF	3904
		28 … 31	0x00	0x7FF	2048	28 … 31	0x00	0x1FFF	8192	28 … 31	0x00	0x7FFF	32768
’10’	’11’	0 … 29	0x80	0x0FF	128	0 … 29	0xC0	0x03FF	832	0 … 29	0xC0	0x0FFF	3904
		30 … 31	0x00	0xFFF	4096	30 … 31	0x00	0x3FFF	16384	30 … 31	0x00	0xFFFF	65536
’11’	’00’	0 … 15	0x00	0x0FF	256	0 … 15	0x00	0x03FF	1024	0 … 15	0x00	0x0FFF	4096
		16 … 31	0x00	0x1FF	512	16 … 31	0x00	0x07FF	2048	16 … 31	0x00	0x1FFF	8192
’11’	’01’	0 … 7	0x00	0x1FF	512	0 … 7	0x00	0x07FF	2048	0 … 7	0x00	0x1FFF	8192
		8 … 15	0x00	0x3FF	1024	8 … 15	0x00	0x0FFF	4096	8 … 15	0x00	0x3FFF	16384
’11’	’10’	0 … 3	0x00	0x3FF	1024	0 … 3	0x00	0x0FFF	4096	0 … 3	0x00	0x3FFF	16384
		4 … 7	0x00	0x7FF	2048	4 … 7	0x00	0x1FFF	8192	4 … 7	0x00	0x7FFF	32768
’11’	’11’	0 … 1	0x00	0x7FF	2048	0 … 1	0x00	0x1FFF	8192	0 … 1	0x00	0x7FFF	32768
		2 … 3	0x00	0xFFF	4096	2 … 3	0x00	0x3FFF	16384	2 … 3	0x00	0xFFFF	65536

4.3 FIFO operation

> **Important !**
> Without F0IO and C4IO clocks the HDLC controller does not work!

4.3.1 HDLC transmit FIFOs

Data can be transmitted from the host bus interface to the FIFO with write access to the registers A_FIFO_DATA0 and A_FIFO_DATA0_NOINC. The HFC-E1 converts the data into HDLC code and tranfers it from the FIFO to the E1 or the PCM bus interface.

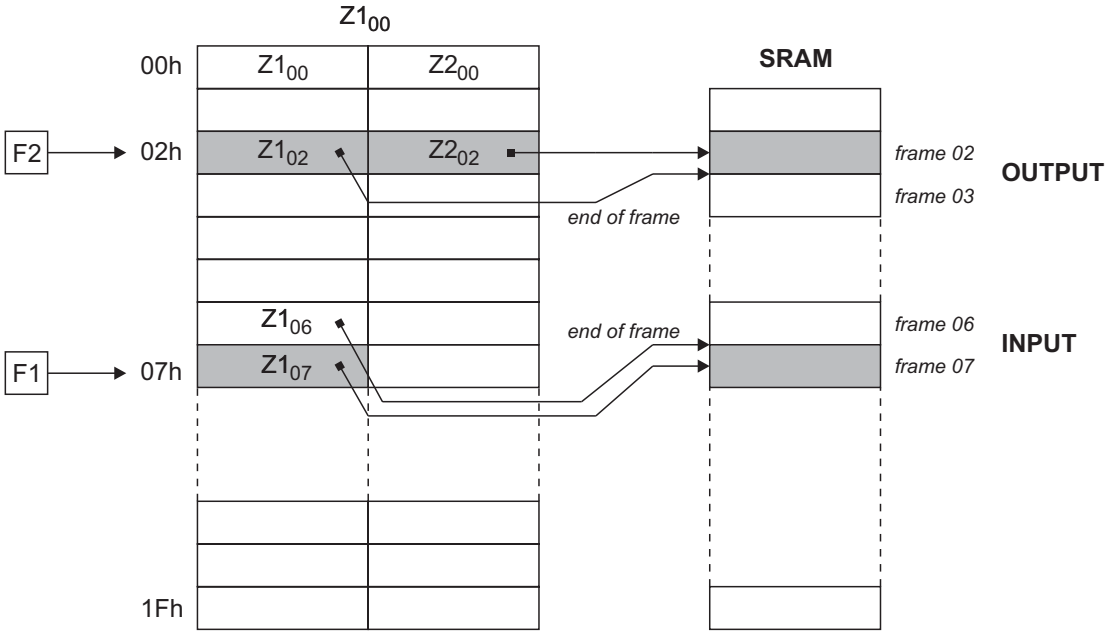

Figure 4.1: *FIFO organization*

The HFC-E1 checks $Z1$ and $Z2$. If $Z1 = Z2$ (FIFO empty) the HFC-E1 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 E1 interface. In this case $Z2$ is not incremented. If also $F1 = F2$ only HDLC flags or continuous '1's are sent to the E1 interface and all counters remain unchanged. If the frame counters are unequal $F2$ is incremented and the HFC-E1 tries to transmit the next frame to the E1 interface. At the end of a frame ($Z2$ reaches $Z1$) it automatically generates the 16 bit CRC checksum and adds an ending flag. If there is another frame in the FIFO ($F1 \neq F2$) the $F2$ counter is incremented again.

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

$Z1(F1)$ is used for the frame which is just written from the host bus side. $Z2(F2)$ is used for the frame which is just being transmitted to the E1 interface side of the HFC-E1. $Z1(F2)$ 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 R_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. $Z2(F2)$ can not be accessed while $Z1(F2)$ can be accessed for transmit FIFOs if V_FZ_MD in the register R_RAM_MISC is set.

> **Please note !**
>
> The HFC-E1 begins to transmit the bytes from a FIFO at the moment the FIFO is changed (writing R_FIFO) or the $F1$ counter is incremented. Switching to the FIFO that is already selected also starts the transmission. Thus by selecting the same FIFO again transmission can be started.

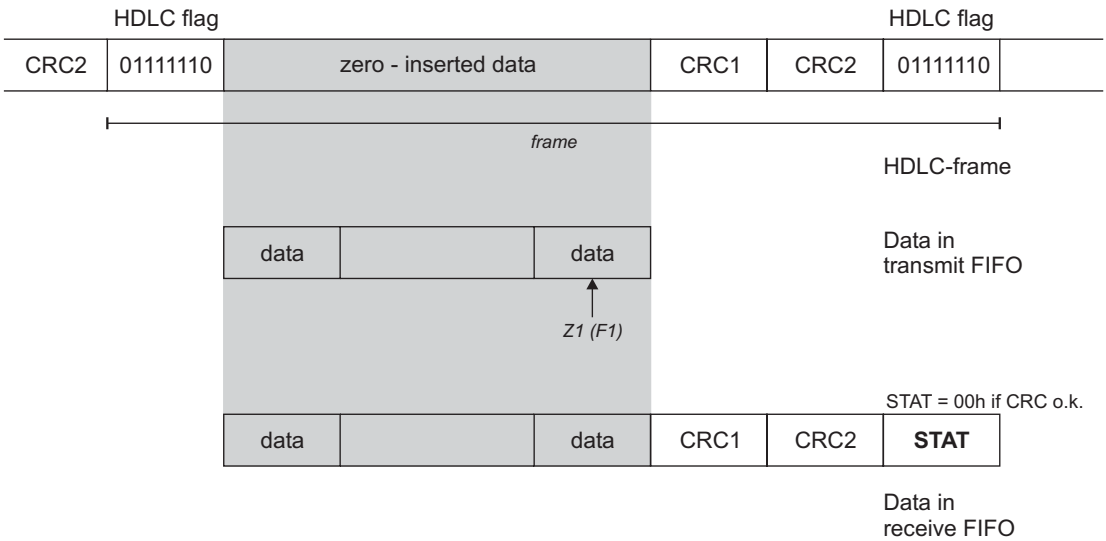

Figure 4.2: *FIFO data organization in HDLC mode*

4.3.2 FIFO full condition in HDLC transmit HFC-channels

Due to the limited number of registers in the HFC-E1 the driver software must maintain a list of frame start and end addresses to calculate the actual FIFO size and to check the FIFO full condition. Because there is a maximum of 32 (resp. 16 with 32k RAM) frame counter values and the start address of a frame is the incremented value of the end address of the last frame the memory table needs to have only 32 (resp. 16) values of 16 bit instead of 64 (resp. 32).

Remember that an increment of Z-value Z_{MAX} is Z_{MIN} in all FIFOs!

There are two different FIFO full conditions. The first one is met when the FIFO contents comes up to 31 frames (128k or 512k RAM) or 15 frames (32k RAM). There is no possibility

for HFC-E1 to manage more frames even if the frames are very small. The second limitation is the overall size of the FIFO.

4.3.3 HDLC receive FIFOs

The receive HFC-channels receive data from the E1 or PCM bus interface read registers. The data is converted from HDLC into plain data and sent to the FIFO. The data can then be read via the host bus interface.

The HFC-E1 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-E1 into plain data. After the ending flag of a frame the HFC-E1 checks the HDLC CRC checksum. If it is correct one byte with all '0's is inserted behind the CRC data in the FIFO named STAT (see Fig. 4.2). This last byte of a frame in the FIFO is different from all '0's if there is no correct CRC field at the end of the frame.

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

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

After a frame is received completely $F1$ is incremented by the HFC-E1 automatically and the next frame can be received.

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

$Z1(F1)$ is used for the frame which is just received from the E1 interface side of the HFC-E1. $Z2(F2)$ is used for the frame which is just beeing transmitted to the host bus interface. $Z1(F2)$ is the end of frame pointer of the current output frame.

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

In the receive HFC-channels $F2$ must be incremented from the host interface side after the software detects an end of receive frame ($Z1 = Z2$) and $F1 \neq F2$. Then the current value of $Z2$ is stored, $F2$ is incremented and $Z2$ is copied as start address of the next frame. This is done by setting the bit V_INC_F in the register R_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-E1. Otherwise the first 4 bytes of the FIFO will be taken from the internal buffer and may be invalid.

4.3.4 FIFO full condition in HDLC receive HFC-channels

Because of the E1 time slots 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-E1. The HFC-E1 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 31 input frames (resp. 15 frames with 32k RAM) or a memory overflow of the FIFO because of excessive data.

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-E1 FIFOs it is easy to poll the HFC-E1 even in large time intervalls without having to fear a FIFO overflow condition.

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_F in the register R_INC_RES_FIFO.

4.3.5 Transparent mode of the HFC-E1

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 E1 or PCM bus interface and data from the E1 or PCM bus interface is sent directly to the FIFO.

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

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

Normally the last byte is undefined because of the Z-counter pointing to a previously unwritten address. To define the last byte, the last write access to the FIFO must be done without Z increment (see register A_FIFO_DATA0_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 E1 or PCM bus interface.

Transmit and receive transparent data can be done in two ways. The usual way is transporting FIFO data to the E1 interface with the LSB first as usual in HDLC mode. The second way is transmitting the bytes in reverse bit order as usual for PCM data. So the first bit is the MSB. The bit order can be reversed by setting bit V_REV of the register R_FIFO when the FIFO is selected.

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

4.3.6 Reading F- and Z-counters

For all asynchronous host accesses to the HFC-E1 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 Register description

4.4.1 Write only registers

R_INC_RES_FIFO [FIFO]	(write only)	0x0E

Increment and reset FIFO register

This register is automatically cleared.

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

Bits	Reset Value	Name	Description
0		V_INC_F	**Increment the F-counters of the selected FIFO** '0' = no increment '1' = increment
1		V_RES_F	**FIFO reset** '0' = no reset '1' = reset selected FIFO (F- and Z-counters and channel mask are resetted, but not the A_CON_HDLC register)
2		V_RES_LOST	**LOST error bit reset** '0' = no reset '1' = reset LOST
7..3		(reserved)	Must be '00000'.

4.4.2 Read only registers

A_Z1L [FIFO]	(read only)	0x04

FIFO input counter $Z1$, low byte

This address can also be accessed with word and double word width to read the complete $Z1$-counter or $Z1$- and $Z2$-counters together (see registers A_Z1 and A_Z12).

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

Bits	Reset Value	Name	Description
7..0		V_Z1L	Bits [7..0] counter value of $Z1$

(See Table 4.3 for reset value.)

A_Z1H [FIFO]	(read only)	0x05

FIFO input counter $Z1$, high byte

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

Bits	Reset Value	Name	Description
7..0		V_Z1H	Bits [15..8] counter value of $Z1$

(See Table 4.3 for reset value.)

A_Z1 [FIFO]	(read only)	0x04

FIFO input counter $Z1$

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

Bits	Reset Value	Name	Description
15..0		V_Z1	Bits [15..0] counter value of $Z1$

(See Table 4.3 for reset value.)

A_Z2L [FIFO]	(read only)	0x06

FIFO output counter $Z2$, low byte

This address can also be accessed with word width to read the complete $Z2$-counter (see register A_Z2).

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

Bits	Reset Value	Name	Description
7..0	0	V_Z2L	Bits [7..0] counter value of $Z2$

(See Table 4.3 for reset value.)

A_Z2H [FIFO]	(read only)	0x07

FIFO output counter $Z2$, high byte

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

Bits	Reset Value	Name	Description
7..0	0	V_Z2H	Bits [15..8] counter value of $Z2$

(See Table 4.3 for reset value.)

A_Z2 [FIFO]	(read only)	0x06

FIFO output counter $Z2$

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

Bits	Reset Value	Name	Description
15..0	0	V_Z2	Bits [15..0] counter value of $Z2$

(See Table 4.3 for reset value.)

A_Z12 [FIFO]	(read only)	0x04

FIFO input counters $Z1$ and $Z2$

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

Bits	Reset Value	Name	Description
31..0		V_Z12	**Bits [15..0] are counter value of $Z1$ and bits [31..16] are counter value of $Z2$**

(See Table 4.3 for reset value.)

A_F1 [FIFO]	(read only)	0x0C

FIFO input HDLC frame counter $F1$

This address can also be accessed with word width to read the $F1$- and $F2$-counters together (see register A_F12).

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

Bits	Reset Value	Name	Description
7..0		V_F1	**Counter value** Up to 31 HDLC frames (resp. 15 with 32k RAM) can be stored in each FIFO.

(See Table 4.3 for reset value.)

A_F2 [FIFO]	(read only)	0x0D

FIFO output HDLC frame counter $F2$

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

Bits	Reset Value	Name	Description
7..0		V_F2	**Counter value** Up to 31 HDLC frames (resp. 15 with 32k RAM) can be stored in each FIFO.

(See Table 4.3 for reset value.)

A_F12 [FIFO]	(read only)	0x0C

FIFO input HDLC frame counter $F1$

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

Bits	Reset Value	Name	Description
7..0		V_F1	**Bits [7..0] are counter value of** $F1$ **and bits [15..8] are counter value of** $F2$ Up to 31 HDLC frames (resp. 15 with 32k RAM) can be stored in each FIFO.

(See Table 4.3 for reset value.)

R_INT_DATA	(read only)	0x88

Internal data register

This register can be read to access data with short read signal.

Bits	Reset Value	Name	Description
7..0		V_INT_DATA	**Internal data buffer**

4.4.3 Read / write registers

A_FIFO_DATA0 [FIFO]	**(read / write)**	0x80

FIFO data register

This address can also be accessed with word and double word width to access two or four data bytes (see registers A_FIFO_DATA1 and A_FIFO_DATA2).

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

Bits	Reset Value	Name	Description
7..0	0	**V_FIFO_DATA0**	**Data byte** Read / write one byte from / to the FIFO selected in the R_FIFO register and increment Z-counter by 1.

A_FIFO_DATA1 [FIFO]	**(read / write)**	0x80

FIFO data register

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

Bits	Reset Value	Name	Description
15..0	0	**V_FIFO_DATA1**	**Data word** Read / write one word from / to the FIFO selected in the R_FIFO register and increment Z-counter by 2.

A_FIFO_DATA2 [FIFO]	(read / write)	0x80

FIFO data register

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

Bits	Reset Value	Name	Description
31..0	0	V_FIFO_DATA2	**Data double word** Read / write two words from / to the FIFO selected in the R_FIFO register and increment Z-counter by 4.

A_FIFO_DATA0_NOINC [FIFO]	(read / write)	0x84

FIFO data register

This address can also be accessed with word and double word width to access two or four data bytes (see registers A_FIFO_DATA1_NOINC and A_FIFO_DATA2_NOINC).

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

Bits	Reset Value	Name	Description
7..0	0	V_FIFO_DATA0_NOINC	**Data byte** Read access: Read one byte from the FIFO selected in the R_FIFO register and increment Z-counter by 1. Write access: Write one byte 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 repeately transmitted automatically.)

A_FIFO_DATA1_NOINC [FIFO] (read / write)			0x84

FIFO data register

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

Bits	Reset Value	Name	Description
15..0	0	V_FIFO_DATA1_NOINC	**Data word** Read access: Read one word from the FIFO selected in the R_FIFO register and increment Z-counter by 2. Write access: Write one word to the FIFO selected in the R_FIFO register without incrementing Z-counter.

A_FIFO_DATA2_NOINC [FIFO] (read / write)			0x84

FIFO data register

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

Bits	Reset Value	Name	Description
31..0	0	V_FIFO_DATA2_NOINC	**Data double word** Read access: Read two words from the FIFO selected in the R_FIFO register and increment Z-counter by 4. Write access: Write two words to the FIFO selected in the R_FIFO register without incrementing Z-counter.

Chapter 5

E1 interface

Table 5.1: *Overview of the HFC-E1 E1 pins*

Number	Name	Description
184	T_B	E1 interface transmit data B
185	T_A	E1 interface transmit data A
187	ADJ_LEV	E1 interface level generator
188	R_B	E1 interface receive input B
189	LEV_B	E1 interface level detect B
190	LEV_A	E1 interface level detect A
191	R_A	E1 interface receive input A

Table 5.2: *Overview of the HFC-E1 E1 interface registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x20	R_E1_WR_STA	155	0x20	R_STATE	169
0x22	R_LOS0	155	0x24	R_RX_STA0	170
0x23	R_LOS1	156	0x25	R_RX_STA1	171
0x24	R_RX0	157	0x26	R_RX_STA2	171
0x25	R_RX_FR0	158	0x27	R_RX_STA3	172
0x26	R_RX_FR1	159	0x2C	R_SLIP	172
0x28	R_TX0	160	0x30	R_FAS_ECL	173
0x29	R_TX1	161	0x31	R_FAS_ECH	173
0x2D	R_TX_FR1	163	0x32	R_VIO_ECL	173
0x2E	R_TX_FR2	164	0x33	R_VIO_ECH	174
0x30	R_RX_OFF	165	0x34	R_CRC_ECL	174
0x31	R_SYNC_OUT	166	0x35	R_CRC_ECH	174
0x34	R_TX_OFF	167	0x36	R_E_ECL	175
0x35	R_SYNC_CTRL	168	0x37	R_E_ECH	175
0x38	R_PWM0	197	0x38	R_SA6_SA13_ECL	175
0x39	R_PWM1	197	0x39	R_SA6_SA13_ECH	176
0x46	R_PWM_MD	198	0x3A	R_SA6_SA23_ECL	176
			0x3B	R_SA6_SA23_ECH	176

5.1 Interface functionality

The HFC-E1 is aquipped with a fully ETSI compliant (TBR4) E1 interface which handles 32 time slots of 8 bits each. The time slots are numbered 0 … 31. Slot 0 is used for synchronization purposes and for the CRC4 prodcedure which checks for data integrity. All other slots can be used for data transmission. In ISDN environments slot 16 is normally used as D-channel.

The HFC-E1 provides the F/G state of the E1 interface in the register R_E1_WR_STA. It is also implemented to force a specific state by overwriting the automatic E1 state machine.

Fundamental interface mode selections can be done by writing registers R_RX0 for receive direction and R_TX0 for transmit direction. Fiber optical interface can be selcted by setting V_RX_CODE and V_TX_CODE to NRZ. In this case R_A is data input and R_B is clock input in receive direction; accordingly T_A is data output and T_B is clock output then.

For normal E1 operation with the interface circuit of Figure 5.3 and 5.4 the register R_RX0 should be set to 0x01.

In R_RX_FR1 and R_TX_FR2 double frame or multiframe format is selectable. Double frame format uses a simple synchronization algorithm (see Figure **??**) and Multiframe format uses the CRC4 procedure (see Figure **??**).

There are several bits to configure different behavior of the time slot 0 synchronization data. Time slot 0 data can be generated automatically according to the selected mode or can be generated by FIFO data or from a special area in the RAM of the HFC-E1. If the RAM buffer is used the area is organized as an alternating buffer. So one half can be read or written by the host processor when the other half sends or receives via the E1 interface. V_RX_SL0_RAM in register R_RX_FR1 and V_TX_SL0_RAM in register R_TX_FR2 switch between the RAM area and the HFC-channel[0] as data destination / source.

The registers R_RX_FR0 and R_RX_FR1 are for the selection of different synchonisation options and how slot 0 of the E1 interface is interpreted. R_TX_FR0, R_TX_FR1 and R_TX_FR2 are used for the selection of different slot 0 data generation.

The HFC-E1 includes an elastic buffer in receive and transmit direction which can be 0 ... 3 times 125 μs. Bigger buffers lead to more delay between receive or transmit data in the FIFOs or the PCM interface and real data on the E1 interface.

The registers R_RX_OFF and R_TX_OFF are used for buffer size selection and buffer initialisation. After initialisation the buffers are FIFOs. In the register R_SLIP a bit is set when there is a buffer underrun or overrun. This is reported only when the full 4 frame FIFO is not enough to handle the data without a slip. Two other bits are slip detection bits which remain set after a slip until the R_SLIP register is read.

The loss of receive signal (LOS) condition can be set in the registers R_LOS0 and R_LOS1. A LOS condition is reported in the bit V_SIG_LOS of the register R_RX_STA0 and by changing the state of the state machine accordingly.

The Receiving Alarm Indication Signal (AIS) is reported in the bit V_AIS of the register R_RX_STA0. Sending of AIS can be switched on with V_AIS_OUT in register R_TX1.

Some receive status bits in the rigisters R_RX_STA1 ... R_RX_STA3 are readable but only for some diagnostic purpose. These bits are only valid for 125 μs or those related to a multiframe are valid for 2 ms.

There are 6 error counters of 16 bit size in the HFC-E1 interface. They can operate in 2 modes. If V_AUTO_ERR_RES is 0 then they function as normal counters. If V_AUTO_ERR_RES is 1 then every second the counter value is latched and the counter starts again with 0.

5.2 Clock synchronization

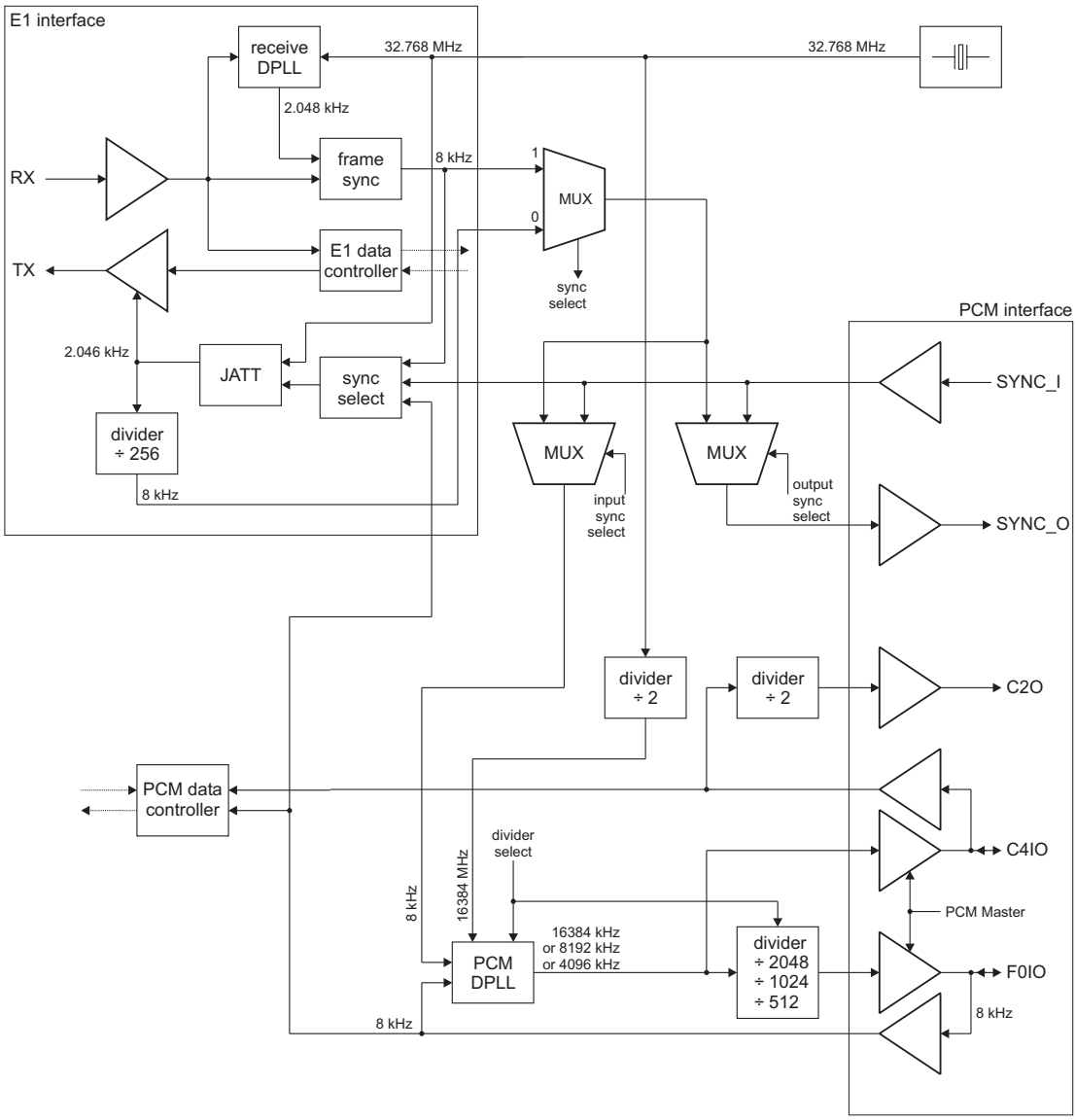

Figure 5.1: *E1 clock synchronization*

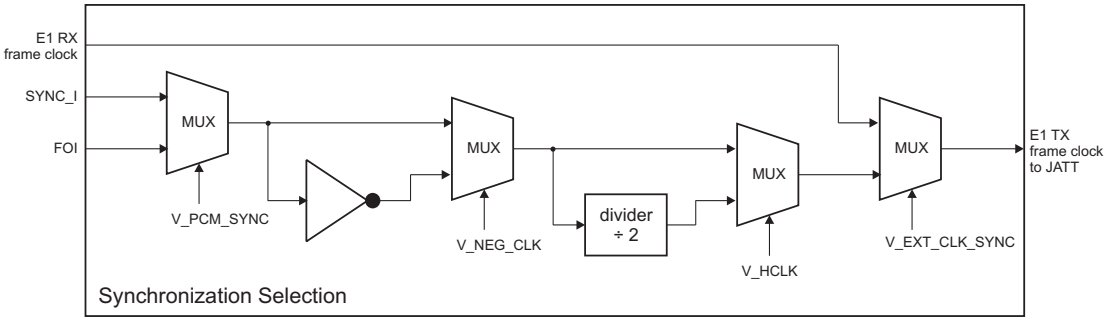

Figure 5.2: *Detail of the E1 interface synchronization selection shown in Figure 5.1*

5.3 External circuitries

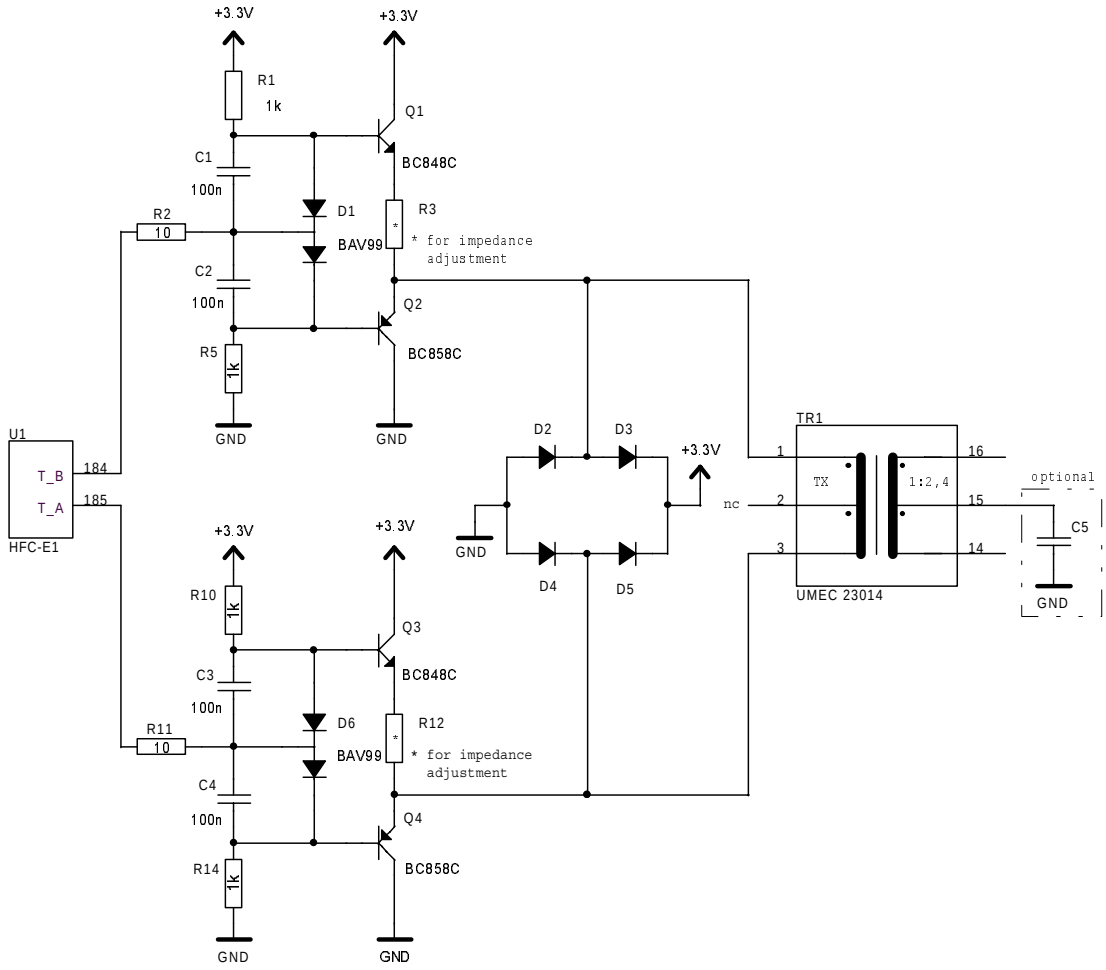

Figure 5.3: *External E1 transmit circuitry*

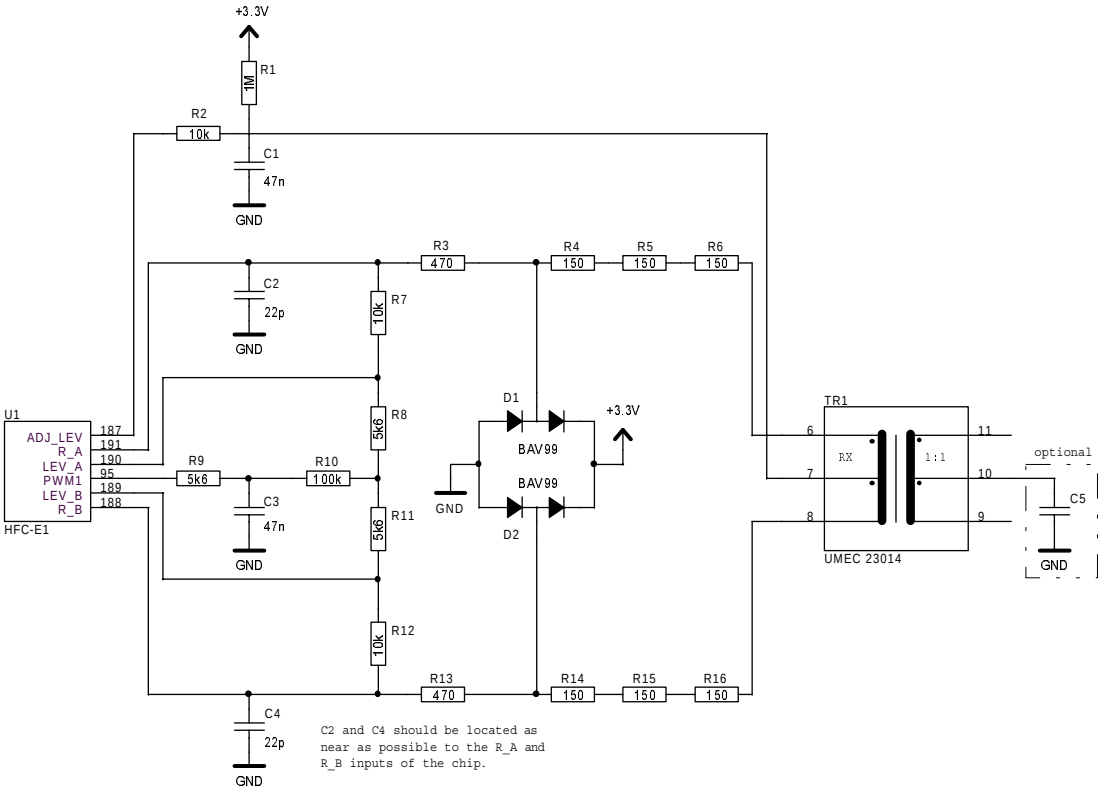

Figure 5.4: *External E1 receive circuitry*

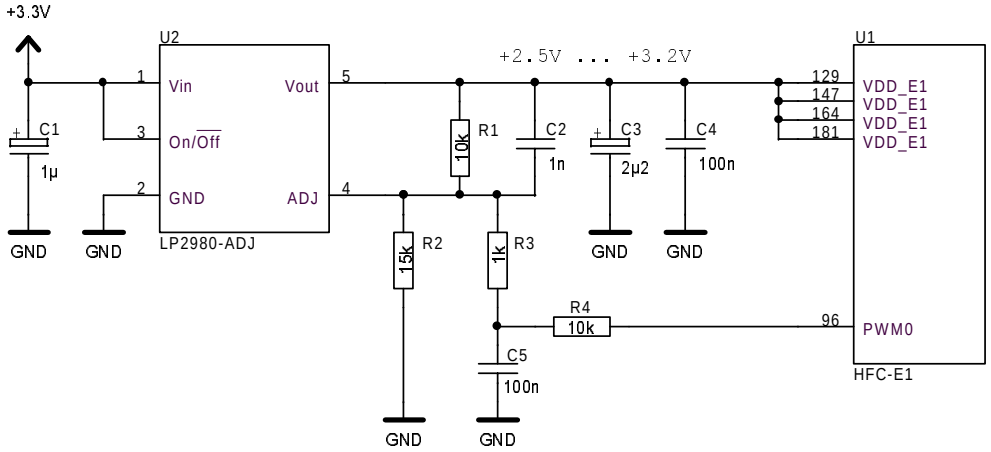

Figure 5.5: *VDD_E1 voltage generation*

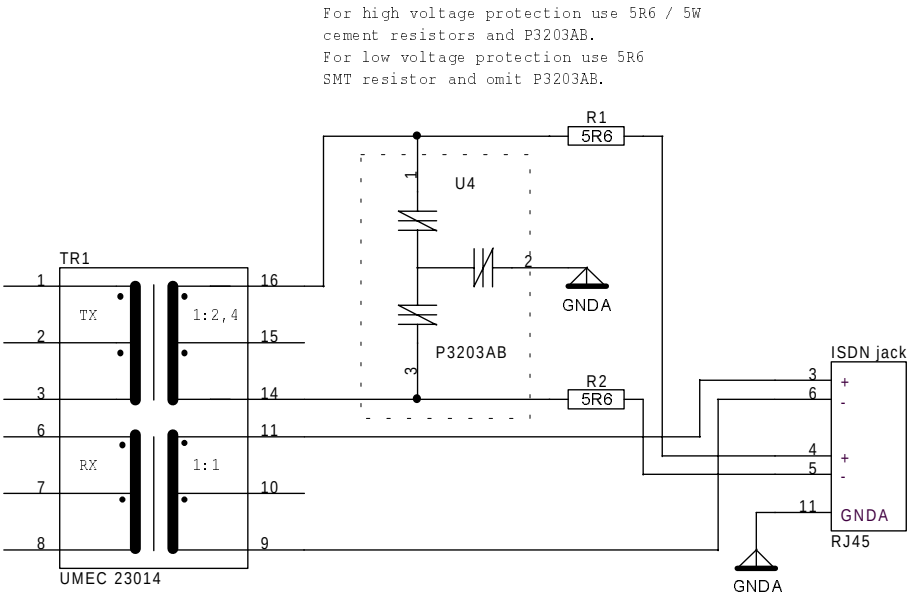

Figure 5.6: *Connector circuitry in LT mode*

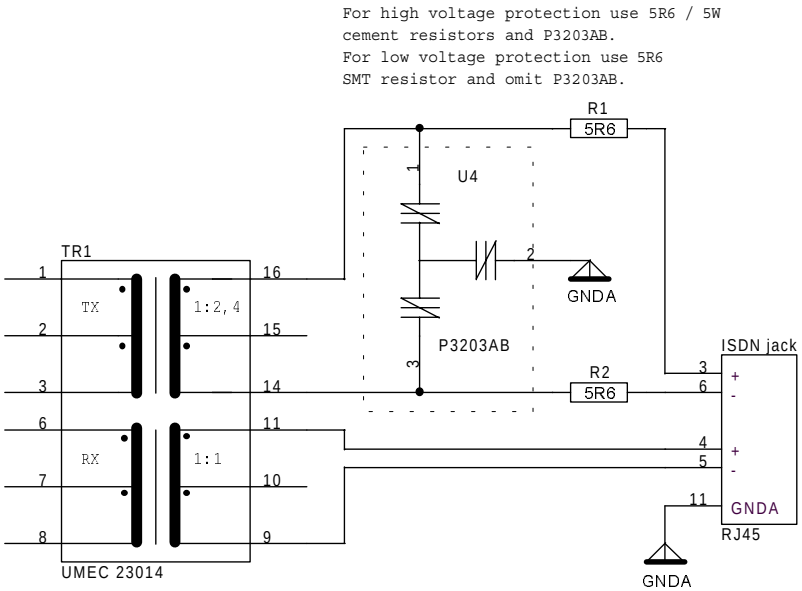

Figure 5.7: *Connector circuitry in TE mode*

5.4 Register description

5.4.1 Write only register

R_E1_WR_STA	(write only)	0x20

E1 state machine register

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

Bits	Reset Value	Name	Description
2..0	0	**V_E1_SET_STA**	**Binary value of new state** (LT: Gx, TE: Fx) V_E1_LD_STA must also be set to load the state.
3		**(reserved)**	Must be '0'.
4	1	**V_E1_LD_STA**	**Load the new state** '0' = enable the state machine '1' = load the prepared state (V_E1_SET_STA) and stops the state machine **Note:** After writing an invalid state the state machine goes to deactivated state.
7..5		**(reserved)**	Must be '000'.

R_LOS0	(write only)	0x22

Alarm set value for loss of input signal

Bits	Reset Value	Name	Description
7..0	0	**V_LOS0**	**LOS alarm** LOS alarm will be active if the incoming data stream has no transitions in (V_LOS0 $+1) \cdot 16$ consecutive data bit times. Maximum time is $256 \cdot 16 \cdot 488\,\mathrm{ns} = 2\,\mathrm{ms}$.

R_LOS1	(write only)	0x23

Alarm clear value for loss of input signal

Bits	Reset Value	Name	Description
7..0	0	V_LOS1	**LOS alarm** LOS alarm will be cleared if the incoming data stream has V_LOS1 +1 transitions in LOS0 time interval. After LOS alarm is cleared a new LOS0 time interval will be started.

R_RX0	(write only)	0x24

E1 receiver configuration register 0

Bits	Reset Value	Name	Description
1..0	0	V_RX_CODE	**Receive code** '00' = NRZ (pin R_A is data input and pin R_B is clock input in NRZ mode) '01' = HDB3 code '10' = AMI code '11' = reserved
2	0	V_RX_FBAUD	**Full / half bauded** '0' = receive pulse is half bit long '1' = receive pulse is full bit long
3	0	V_RX_CMI	**Code mark inversion (CMI)** '0' = CMI off '1' = CMI on In CMI mode pin R_B is not used.
4	0	V_RX_INV_CMI	**Inverted CMI code** This bit is only valid if CMI is on. '0' = CMI code '1' = inverted CMI code
5	0	V_RX_INV_CLK	**Polarity of clock** This bit is only valid if data clock input is used (NRZ mode). '0' = clock is not inverted '1' = clock is inverted
6	0	V_RX_INV_DATA	**Polarity of input data** '0' = non-inverted data '1' = inverted data
7	0	V_AIS_ITU	**AIS alarm specification** '0' = according to ETS 300233 '1' = according to ITU-T G.775

R_RX_FR0	(write only)	0x25

E1 receive frame configuration, register 0

Bits	Reset Value	Name	Description
0	0	V_NO_INSYNC	**Transparent mode** '0' = normal operation '1' = no synchronization to input data
1	0	V_AUTO_RESYNC	**Automatic resynchronization** '0' = normal operation '1' = after loss of synchronization the search for multiframe synchronization pattern is initiated again This bit is only valid in CRC multiframe format.
2	0	V_AUTO_RECO	**Automatic error recovery** '0' = normal operation '1' = if there are more than 914 CRC errors in one second the receiver will search for new basic- and multiframing This bit is only valid in multiframing synchronous state.
3	0	V_SWORD_COND	**Service word condition** '0' = loss of synchronization if there are 3 or 4 (depending on V_SYNC_LOSS) consecutive incorrect service words '1' = incorrect service words have no influence in synchronous state
4	0	V_SYNC_LOSS	**Loss of synchronization** '0' = loss of synchronization if there are 3 consecutive incorrect FAS or service words '1' = loss of synchronization if there are 4 consecutive incorrect FAS or service words
5	0	V_XCRC_SYNC	**Extended CRC4 to non-CRC4** '0' = according to ITU-T G.706 '1' = according to ITU-T G.706 except that the synchronizer will still search for multiframing even if the 400 ms is expired
6	0	V_MF_RESYNC	**Multiframe resynchronization** When this bit is set, the resynchronization of CRC multiframe alignment is initiated without influencing doubleframe synchronous state. If V_AUTO_RESYNC is enabled and multiframe alignment can not be regained, a new search of doubleframe is initiated. **Note:** This bit is only valid in CRC multiframe format.
7	0	V_RESYNC	**Resynchronization** '1' = initiate resynchronization of receive frame

R_RX_FR1	(write only)	0x26

E1 receive frame configuration, register 1

Bits	Reset Value	Name	Description
0	0	V_RX_MF	**Multiframe mode** '0' = normal doubleframe mode '1' = multiframe mode (CRC4)
1	0	V_RX_MF_SYNC	**Multiframe alignment error** '0' = normal operation '1' = MFA error leads to loss of synchronization
2	0	V_RX_SL0_RAM	**Time slot 0 data destination** '0' = time slot 0 data is written into HFC-channel 0 '1' = time slot 0 data is written into alternating RAM buffer
4..3		(reserved)	Must be '00'.
5	0	V_ERR_SIM	**Error simulation** This bit is for diagnostic purpose only. '0' = no action '1' = increment all error counters
6	0	V_RES_NMF	**Reset 'no multiframe found' (NMF) status** '0' = no action '1' = reset no MFA found status which is set after 400 ms of MFA searching This bit is automatically cleared.
7		(reserved)	Must be '0'.

R_TX0	(write only)	0x28

E1 transmitter configuration, register 0

Bits	Reset Value	Name	Description
1..0	0	V_TX_CODE	**Transmit code** '00' = NRZ (pin R_A is data output and pin R_B is clock output in NRZ mode) '10' = AMI code '01' = HDB3 code '11' = reserved
2	0	V_TX_FBAUD	**Full / half bauded** '0' = transmit pulse is half bit long '1' = transmit pulse is full bit long
3	0	V_TX_CMI_CODE	**Code mark inversion (CMI)** '0' = CMI off '1' = CMI on (only R_A is used as data output)
4	0	V_TX_INV_CMI_CODE	**Inverted CMI code** This bit is only valid if CMI is on. '0' = CMI code '1' = inverted CMI code
5	0	V_TX_INV_CLK	**Polarity of clock** This bit is only valid if data clock output is enabled. '0' = non-inverted clock '1' = inverted clock
6	0	V_TX_INV_DATA	**Polarity of output data** '0' = non-inverted data '1' = inverted data
7	0	V_OUT_EN	**Buffer enable** '0' = output buffers disabled (tristate) '1' = output buffers enabled

Important !

Transmit data is only generated if V_OUT_EN bit of the register R_TX0 is set to '1'.

R_TX1	(write only)	0x29

E1 transmitter configuration, register 1

Bits	Reset Value	Name	Description
0	0	V_INV_CLK	**Polarity of mark** '0' = normal operation '1' = inverted clock This bit is only valid with CMI code.
1	0	V_EXCHG_DATA_LI	**TxD-exchange** '0' = normal operation '1' = exchange data output lines R_A and R_B
2	0	V_AIS_OUT	**Generate AIS output signal** Continuous '1's are generated.
4..3		(reserved)	Must be '00'.
5	0	V_ATX	**Transmitter mode** '0' = standard transmitter '1' = analog transmitter tandem mode
6	0	V_NTRI	**No tristate** '0' = tristate for gap between pulses enabled '1' = tristate for gap between pulses disabled
7	0	V_AUTO_ERR_RES	**Error counter mode** '0' = normal counter operation after reaching maximum count, counter starts at 0 again '1' = every second the error counters will be reset automatically after they are latched **Note:** The latched state should be read within the next second. During updating reading should be avoided.

Please note !

The default settings are:

V_INV_CLK:	'0'
V_EXCHG_DATA_LI:	'0'
V_ATX:	'1'
V_NTRI:	'1'

R_TX_FR0	(write only)	0x2C

E1 time slot 0 configuration, register 0

Bits	Reset Value	Name	Description
0	0	V_TRP_FAS	**Transparent $S_i(\mathrm{FAS})$ bit** '0' = S_i bit will be taken from V_TX_FAS '1' = HFC-channel 0 data or RAM data will be used (see V_TX_SL0_RAM of register R_TX_FR2)
1	0	V_TRP_NFAS	**Transparent $S_i(\mathrm{NFAS})$ bit** '0' = S_i bit will be taken from V_TX_NFAS '1' = HFC-channel 0 data or RAM data will be used (see V_TX_SL0_RAM of register R_TX_FR2)
2	0	V_TRP_RAL	**Transparent remote alarm** '0' = remote alarm bit will be generated internally from the state machine '1' = HFC-channel 0 data or RAM data will be used (see V_TX_SL0_RAM of register R_TX_FR2)
7..3	0	V_TRP_SA	**Transparent $S_{a4} \ldots S_{a8}$ bits** '0' = S_a bits will be taken from V_TX_SA '1' = HFC-channel 0 data or RAM data will be used (see V_TX_SL0_RAM of register R_TX_FR2)

R_TX_FR1	(write only)	0x2D

E1 time slot 0 configuration, register 1

This register is only used if V_TRP_SL0 of the register R_TX_FR2 is not set.

Bits	Reset Value	Name	Description
0	0	**V_TX_FAS**	$S_i(\mathrm{FAS})$ **bit** This bit is only used in doubleframe format.
1	0	**V_TX_NFAS**	$S_i(\mathrm{NFAS})$ **bit**
2	0	**V_TX_RAL**	**Remote alarm bit** '0' = normal operation '1' = remote alarm bit generated from the state machine is fixed to '1'.
7..3	0	**V_TX_SA**	$S_{a4} \ldots S_{a8}$ **bits**

R_TX_FR2	(write only)	0x2E

E1 time slot 0 configuration, register 2

Bits	Reset Value	Name	Description
0	0	V_TX_MF	**framing selection** '0' = doubleframe format '1' = multiframe mode (CRC4)
1	0	V_TRP_SL0	**Time slot 0 transparent mode** '0' = normal operation '1' = the HFC-channel 0 data or RAM data will be used and the registers R_TX_FR0 and R_TX_FR1 are ignored
2	0	V_TX_SL0_RAM	**Time slot 0 data source** '0' = time slot 0 data comes from HFC-channel 0 '1' = time slot 0 data comes from alternating RAM buffer
3		(reserved)	Must be '0'.
4	0	V_TX_E	**Automatic transmission of submultiframe status** '0' = XS13 and XS15 bits from V_XS13_ON and V_XS15_ON of this register are transmitted '1' = E-bits are transmitted (CRC calculation result)
5	0	V_NEG_E	**Polarity of E-bits** '0' = positive E-bits '1' = negative E-bits
6	0	V_XS13_ON	**Transmit spare bit XS13** (Frame 13 of multiframe) '0' = XS13 is '0' '1' = XS13 is '1' **Note:** This bit is only valid in CRC multiframe.
7	0	V_XS15_ON	**Transmit spare bit XS15** (Frame 15 of multiframe) '0' = XS15 is '0' '1' = XS15 is '1' **Note:** This bit is only valid in CRC multiframe.

R_RX_OFF	(write only)	0x30

E1 receive buffer configuration register

Bits	Reset Value	Name	Description
1..0	0	V_RX_SZ	**Buffer size** Elastic buffer size in number of frames (0 . . . 3)
2	0	V_RX_INIT	**Buffer initialization** Some data may be lost when this bit is set. This bit is automatically cleared.
7..3		(reserved)	Must be ’00000’.

R_SYNC_OUT	(write only)	0x31

E1 synchronization source selection for PCM master

Bits	Reset Value	Name	Description
0	0	V_SYNC_E1_RX	**PCM master synchronization** '0' = PCM master synchronizes on the E1 TX end of frame (EOF) signal '1' = PCM master synchronizes on the E1 RX end of frame (EOF) signal
4..1		(reserved)	Must be '00000'.
5	0	V_IPATS0	**RAI pulse configuration for IPATS test** '0' = normal opeartion '1' = delete short RAI low pulses, increase RAI to a minimum of more than 1 ms **Note:** This bit is only used for passing IPATS test equipment.
6	0	V_IPATS1	**CRC configuration for IPTAS test** '0' = normal operation '1' = delete CRC reporting over E-bits up to 8 ms after MFA synchronization **Note:** This bit is only used for passing IPATS test equipment.
7	0	V_IPATS2	**JATT configuration for IPATS test** '0' = normal operation '1' = stop jitter attenuator (JATT) adaptation when in F3 or G3 state **Note:** This bit is only used for passing IPATS test equipment.

R_TX_OFF	(write only)	0x34

E1 transmit buffer configuration register

Bits	Reset Value	Name	Description
1..0	0	V_TX_SZ	**Buffer size** Elastic buffer size in number of frames (0 . . . 3)
2	0	V_TX_INIT	**Buffer initialization** Some data may be lost when this bit is set. This bit is automatically cleared.
7..3		(reserved)	Must be ’00000’.

R_SYNC_CTRL	(write only)	0x35

E1 transmit clock sychronization register

Bits	Reset Value	Name	Description
0	0	V_EXT_CLK_SYNC	**E1 synchronization source selection** '0' = clock synchronization derived from receive data '1' = synchronization is determined from V_PCM_SYNC, V_NEG_CLK and V_HCLK
1	0	V_SYNC_OFFS	**E1 synchronization type selection** '0' = TX and RX frame synchronization phase offset 0 '1' = TX and RX frame synchronization phase offset arbitrary **Note:** If this bit is set the synchronization process is faster because the phase offset can be arbitrary.
2	0	V_PCM_SYNC	**E1 synchronization source select** '0' = pin SYNC_I '1' = synchronization from PCM pin F0IO
3	0	V_NEG_CLK	**External synchronization clock polarity** '0' = positive edge '1' = negative edge
4	0	V_HCLK	**Half clock frequency** '0' = normal operation '1' = external synchronization clock will be divided by 2
5	0	V_JATT_AUTO_DEL	**Restricted frequency search** '0' = automatic frequency search is initiated after 3 frequency mismatches every 0.5 s '1' = automatic frequency search is initiated after 10 frequency mismatches every 0.5 s
6	0	V_JATT_AUTO	**Automatic JATT adjustment** '0' = automatic JATT adjust enabled '1' = automatic JATT adjust disabled
7	0	V_JATT_EN	**JATT enable** '0' = JATT enabled '1' = JATT disabled (transmit clock is generated from crystal clock)

5.4.2 Read only register

R_STATE	(read only)	0x20
E1 state machine register		

Bits	Reset Value	Name	Description
2..0	0	V_E1_STA	**E1 state** Binary value of actual state (LT: Gx, TE: Fx).
5..3		(reserved)	
6	0	V_ALT_FR_RX	**Alternating RAM bank** Shows which bank of time slot 0 data in RAM is currently used for receive data. Receive data is written to the RAM. This bit is toggled with every multiframe.
7	0	V_ALT_FR_TX	**Alternating RAM bank** Shows which bank of time slot 0 data in RAM is currently used for transmit data. Transmit data is read from the RAM. This bit is toggled with every multiframe.

R_RX_STA0	(read only)	0x24

E1 receive status, register 0

Bits	Reset Value	Name	Description
1..0	0	V_RX_STA	**Receive status** '00' = not synchronized '01' = FAS found '10' = NFAS found after FAS '11' = synchronized (FAS - NFAS - FAS found)
2	0	V_FR_SYNC	**Frame synchronization status** Frame synchronization status according to the state machine.
3	0	V_SIG_LOS	**LOS status** Loss of receive signal detected.
5..4	0	V_MFA_STA	**Status of multi frame alignment (MFA)** '01' = MFA pattern found '10' = MFA reached (2 consecutive MFA patterns found)
6	0	V_AIS	**Receiving Alarm Indication Signal (AIS)**
7	0	V_NO_MF_SYNC	**No multiframe (NMF) synchronization** '1' = no multiframe synchronization found for 400 ms This bit is reset by asserting V_RES_NMF of the register R_RX_FR1.

R_RX_STA1	(read only)	0x25

E1 receive status, register 1

Bits	Reset Value	Name	Description
0	0	V_SI_FAS	$S_i(\mathrm{FAS})$ **in time slot 0**
1	0	V_SI_NFAS	$S_i(\mathrm{NFAS})$ **in time slot 0**
2	0	V_A	**A-bit of time slot 0**
3	0	V_CRC_OK	**CRC result** '1' = CRC4 ok
4	0	V_TX_E1	**Transmit CRC4 E1-bit**
5	0	V_TX_E2	**Transmit CRC4 E2-bit**
6	0	V_RX_E1	**Receive CRC4 E1-bit**
7	0	V_RX_E2	**Receive CRC4 E2-bit**

R_RX_STA2	(read only)	0x26

E1 receive status, register 2

Bits	Reset Value	Name	Description
3..0	0	V_SA6	$S_{A6}[4..1]$ **bits of time slot 0**
5..4		(reserved)	
6	0	V_SA6_OK	S_{A6} **OK** The same value was received in 3 consecutive SMFs.
7	0	V_SA6_CHG	S_{A6} **pattern has changed** This bit is automatically reset after register read.

R_RX_STA3	(read only)	0x27

E1 receive status, register 3

Bits	Reset Value	Name	Description
4..0	0	V_SA84	$S_a[8..4]$ **bits**
7..5		(reserved)	

R_SLIP	(read only)	0x2C

Frequency slip warning register

Bits	Reset Value	Name	Description
0	0	V_SLIP_RX	**Frequency slip in receive transmission** This bit is set when an overflow of the elastic receive buffer has occured as a result of a frequency slip. This bit is automatically cleared with new buffer write access.
2..1	0	(reserved)	
3	0	V_FOSLIP_RX	**Force slip warning** This bit is set when bit V_SLIP_RX had been set at least one time after the last read access to this register. This bit is automatically cleared with an read access to this register.
4	0	V_SLIP_TX	**Frequency slip in transmit transmission** This bit is set when an overflow of the elastic transmit buffer has occured as a result of a frequency slip. This bit is automatically cleared with new buffer read access.
6..5	0	(reserved)	
7	0	V_FOSLIP_TX	**Force slip warning** This bit is set when bit V_SLIP_TX had been set at least one time after the last read access to this register. This bit is automatically cleared with an read access to this register.

R_FAS_ECL	(read only)	0x30

Error counter for missing or wrong FAS, low byte

Bits	Reset Value	Name	Description
7..0	0	V_FAS_ECL	Bits [7..0] of FAS error count

R_FAS_ECH	(read only)	0x31

Error counter for missing or wrong FAS, high byte

Bits	Reset Value	Name	Description
7..0	0	V_FAS_ECH	Bits [15..8] of FAS error count

R_VIO_ECL	(read only)	0x32

Error counter for code violation of HDB3 code, low byte

Bits	Reset Value	Name	Description
7..0	0	V_VIO_ECL	Bits [7..0] of code violation error count

R_VIO_ECH	(read only)	0x33

Error counter for code violation of HDB3 Code, high byte

Bits	Reset Value	Name	Description
7..0	0	V_VIO_ECH	Bits [15..8] of code violation error count

R_CRC_ECL	(read only)	0x34

Receive CRC4 error count, low byte

Bits	Reset Value	Name	Description
7..0	0	V_CRC_ECL	Bits [7..0] of CRC4 error count

R_CRC_ECH	(read only)	0x35

Receive CRC4 error count, high byte

Bits	Reset Value	Name	Description
7..0	0	V_CRC_ECH	Bits [15..8] of CRC4 error count

R_E_ECL	(read only)	0x36

Error counter for CRC4 error reporting by received E-bits, low byte

Bits	Reset Value	Name	Description
7..0	0	V_E_ECL	Bits [7..0] of CRC4 error count

R_E_ECH	(read only)	0x37

Error counter for CRC4 error reporting by received E-bits, high byte

Bits	Reset Value	Name	Description
7..0	0	V_E_ECH	Bits [15..8] of CRC4 error count

R_SA6_SA13_ECL	(read only)	0x38

Error count of [SA64, SA63, SA62, SA61] = '0001' **or** '0011'**, low byte**

Bits	Reset Value	Name	Description
7..0	0	V_SA6_SA13_ECL	Bits [7..0] of error count

R_SA6_SA13_ECH	(read only)	0x39

Error count of [SA64, SA63, SA62, SA61] = '0001' or '0011', high byte

Bits	Reset Value	Name	Description
7..0	0	V_SA6_SA13_ECH	Bits [15..8] of error count

R_SA6_SA23_ECL	(read only)	0x3A

Error count of [SA64, SA63, SA62, SA61] = '0010' or '0011', low byte

Bits	Reset Value	Name	Description
7..0	0	V_SA6_SA23_ECL	Bits [7..0] of error count

R_SA6_SA23_ECH	(read only)	0x3B

Error count of [SA64, SA63, SA62, SA61] = '0010' or '0011', high byte

Bits	Reset Value	Name	Description
7..0	0	V_SA6_SA23_ECH	Bits [15..8] of error count

Chapter 6

PCM interface

Table 6.1: *Overview of the HFC-E1 PCM interface registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x10	R_SLOT	122	0x18	R_F0_CNTL	193
0x14	R_PCM_MD0	183	0x19	R_F0_CNTH	193
0x15	R_SL_SEL0	184			
0x15	R_SL_SEL1	185			
0x15	R_SL_SEL2	186			
0x15	R_SL_SEL3	186			
0x15	R_SL_SEL4	187			
0x15	R_SL_SEL5	187			
0x15	R_SL_SEL6	188			
0x15	R_SL_SEL7	188			
0x15	R_PCM_MD1	189			
0x15	R_PCM_MD2	190			
0x15	R_SH0L	191			
0x15	R_SH0H	191			
0x15	R_SH1L	191			
0x15	R_SH1H	192			

Table 6.2: *Overview of the HFC-E1 PCM pins*

PCM pins:

Number	Name	Description
97	SYNC_I	Synchronization Input
98	SYNC_O	Synchronization Output
117	C2O	PCM bit clock output
118	C4IO	PCM double bit clock I/O
119	F0IO	PCM frame clock I/O (8 kHz)
120	STIO1	PCM data bus 1, I or O per time slot
121	STIO2	PCM data bus 2, I or O per time slot

CODEC select via enable lines:

Number	Name	Description
107	F1_7	PCM CODEC enable 7
108	F1_6	PCM CODEC enable 6
109	F1_5	PCM CODEC enable 5
110	F1_4	PCM CODEC enable 4
111	F1_3	PCM CODEC enable 3
112	F1_2	PCM CODEC enable 2
113	F1_1	PCM CODEC enable 1
114	F1_0	PCM CODEC enable 0

CODEC select via time slot number:

Number	Name	Description
106 *	F_Q6	PCM time slot count 6
107 *	F_Q5	PCM time slot count 5
108 *	F_Q4	PCM time slot count 4
109 *	F_Q3	PCM time slot count 3
110 *	F_Q2	PCM time slot count 2
111 *	F_Q1	PCM time slot count 1
112 *	F_Q0	PCM time slot count 0
113 *	SHAPE1	PCM CODEC enable shape signal 1
114 *	SHAPE0	PCM CODEC enable shape signal 0

(*: Second pin function)

6.1 PCM interface function

The PCM interface has up to 32, 64 or 128 time slots for receive and transmit data depending on the PCM clock frequency and the selected mode. The functional block diagram is shown in Figure 6.1.

The HFC-E1 has two PCM data pins STIO1 and STIO2 which can both be input or output. PCM output data is transmitted to two output buffers. These can be enabled independently from each other. PCM input data can either come from one of the two PCM data pins or from the PCM output channel. This way PCM data can be looped internally.

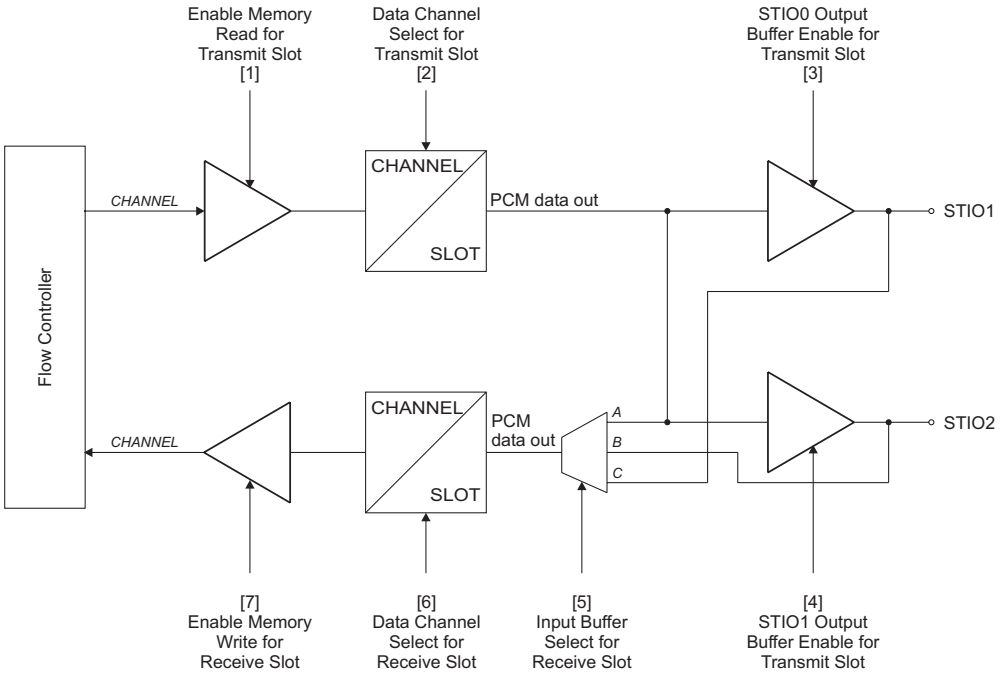

Figure 6.1: *PCM interface function block diagram*

6.2 PCM initialization

After hard or soft reset the PCM interface starts an initialization sequence to set all A_SL_CFG registers of the PCM time slots to the reset value 0. This can be done only if valid C4IO and F0IO signals exist. The initialization process stops after 2 F0IO periods. To check if the initialization sequence is finished after a reset, the register R_F0_CNTL value must be equal or greater than 2.

6.3 External CODECs

External CODECs can be connected to the HFC-E1 PCM interface. There are two ways of programming the PCM–CODEC–interconnection. First, a set of eight CODEC enable lines

Table 6.3: *PCM interface configuration with bitmaps of the register A_SL_CFG (The reference numbers relate to the numbers given in Figure 6.1)*

Reference	Function		Bitmap	Value
[1]	Enable memory read for transmit slot		V_ROUT	$\neq$ '00'
[2]	HFC-channel select for transmit slot		V_CH_NUM1	0 . . . 31
[3]	STIO1 output buffer enable for transmit slot		V_ROUT	'10'
[4]	STIO2 output buffer enable for transmit slot		V_ROUT	'11'
[5]	Input buffer select for receive slot	(MUX A)	V_ROUT	'01' (Loop PCM internally)
		(MUX B)	V_ROUT	'10' (Data In from STIO1)
		(MUX C)	V_ROUT	'11' (Data In from STIO2)
[6]	HFC-channel select for receive slot		V_CH_NUM1	0 . . . 31
[7]	Enable memory write for receive slot		V_ROUT	$\neq$ '00'

allow to connect up to eight external CODECs to the HFC-E1. The second way uses the current time slot number that must be decoded to a CODEC's select signal. Then up to 128 external CODECs can be connected to the HFC-E1. The choice of these connectivities is done with V_CODEC_CON of the register R_PCM_MD1.

6.3.1 CODEC select via enable lines

The HFC-E1 has eight CODEC enable signals F1_7 . . . F1_0. Every external CODEC has to be assigned to a PCM time slot via the bitmaps V_SL_SEL7 . . . V_SL_SEL0 of the registers R_SL_SEL7 . . . R_SL_SEL0.

Two shape signals can be programmed. The last bit determines the inactive level by which non-inverted and inverted shape signals can be programmed. Every external CODEC can choose one of the two shape signals with the bits V_SH_SEL7 . . . V_SH_SEL0 of the registers R_SL_SEL7 . . . R_SL_SEL0.

Figure 6.2 shows an example with two external CODECs with F1_0 and F1_1 enable signals. Time slot 0 starts with the F0IO pulse. In this example – assuming that PCM30 is configured – F1_0 enables the first CODEC on time slot 0 and shape bytes on R_SH0L and R_SH0H with

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

and the second CODEC on time slot 1 and shape bytes on R_SH1L and R_SH1H with

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

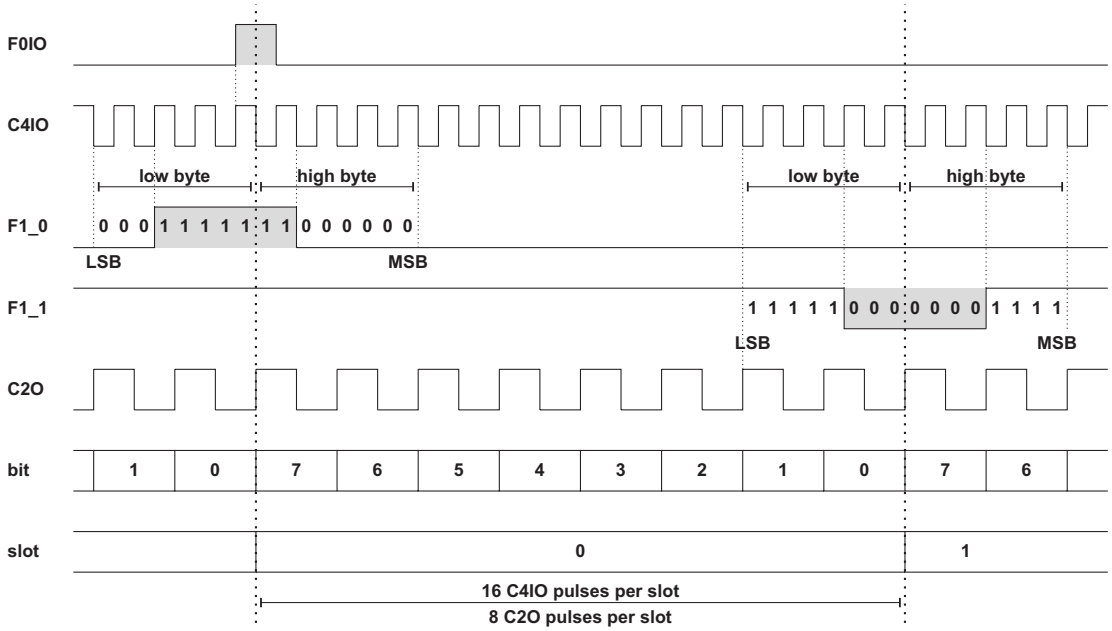

Figure 6.2: *Example for two CODEC enable signal shapes with SHAPE0 and SHAPE1.*

The shown shape signals have to be programmed in reverse bit order by

R_PCM_MD0	: V_PCM_ADDR	=	0xC	(R_SH0L register accessible)	
R_SH0L	: V_SH0L	=	0xF8	(0xF8 = '11111000' $\xrightarrow{reverse}$	'00011111')
R_PCM_MD0	: V_PCM_ADDR	=	0xD	(R_SH0H register accessible)	
R_SH0L	: V_SH0L	=	0x03	(0x03 = '00000011' $\xrightarrow{reverse}$	'11000000')
R_PCM_MD0	: V_PCM_ADDR	=	0xE	(R_SH1L register accessible)	
R_SH0L	: V_SH0L	=	0x1F	(0x1F = '00011111' $\xrightarrow{reverse}$	'11111000')
R_PCM_MD0	: V_PCM_ADDR	=	0xF	(R_SH1H register accessible)	
R_SH0L	: V_SH0L	=	0xF0	(0xF0 = '11110000' $\xrightarrow{reverse}$	'00001111')

6.3.2 CODEC select via time slot number

Alternatively, external CODECs can be enabled by decoding the time slot number. In this case, two programmable shape signals SHAPE0 and SHAPE1 are put out with every time slot. The current time slot number is issued on the pins F_Q6 ... F_Q0.

The shape signals can be programmed. The example in Figure 6.3 shows shape signals that are programmed in the same way as shown above (see Section 6.3.1).

F_Q6 ... F_Q0 must be decoded externally to generate CODEC select signals in dependence on the PCM time slot.

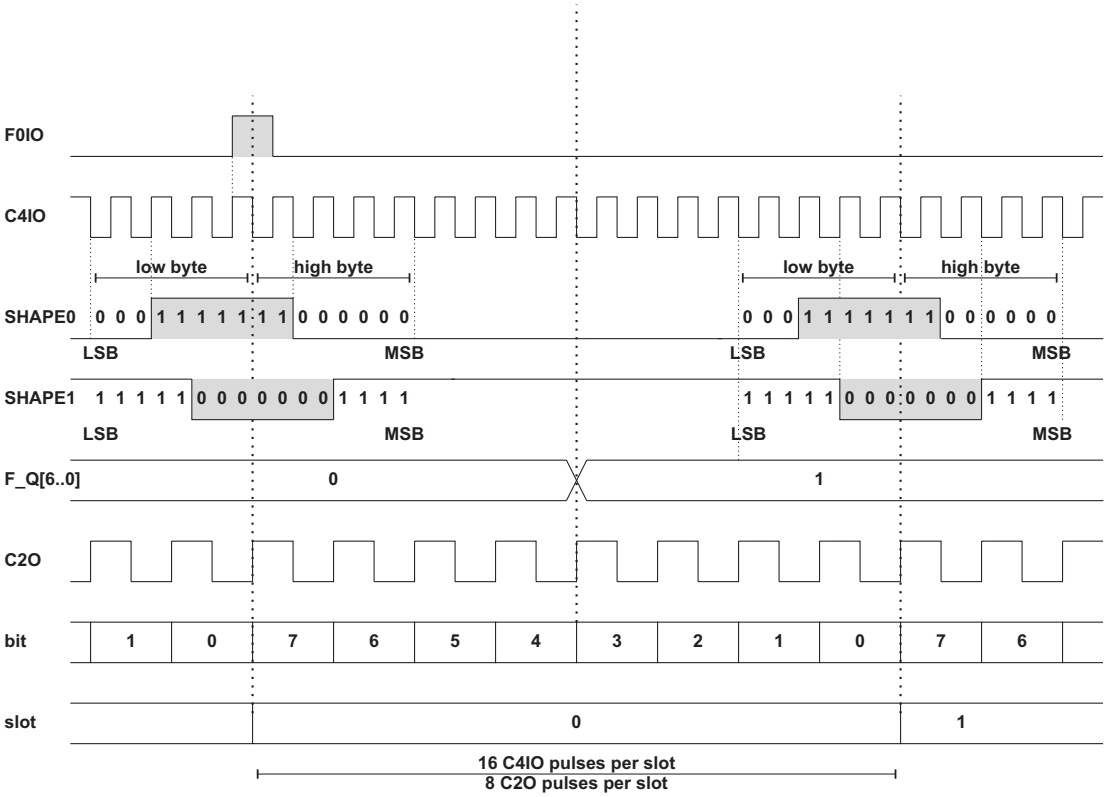

Figure 6.3: *Example for two CODEC enable signal shapes*

6.4 Register description

6.4.1 Write only register

R_PCM_MD0	(write only)	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 slave mode** '0' = active for one C4IO clock (244 ns at 4 MHz) '1' = active for two C4IO clocks (488 ns at 4 MHz)
7..4	0	V_PCM_ADDR	**Index value to select the register at address 15** At address 15 a so-called multi-register is accessible. 0 = R_SL_SEL0 register accessible 1 = R_SL_SEL1 register accessible 2 = R_SL_SEL2 register accessible 3 = R_SL_SEL3 register accessible 4 = R_SL_SEL4 register accessible 5 = R_SL_SEL5 register accessible 6 = R_SL_SEL6 register accessible 7 = R_SL_SEL7 register accessible 9 = R_PCM_MD1 register accessible 0xA = R_PCM_MD2 register accessible 0xC = R_SH0L register accessible 0xD = R_SH0H register accessible 0xE = R_SH1L register accessible 0xF = R_SH1H register accessible

R_SL_SEL0	(write only)	0x15

Slot selection register for pin F1_0

This multi-register is selected with bitmap V_PCM_ADDR = 0 of the register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_0 is disabled.

Bits	Reset Value	Name	Description
6..0	0x7F	**V_SL_SEL0**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_0. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL0**	**Shape selection** '0' = use shape 0 set by R_SH0L and R_SH0H registers '1' = use shape 1 set by R_SH1L and R_SH1H registers

Important !

For selecting slot 0 the value that has to be written to the bitmap V_SL_SEL0 ...V_SL_SEL7 of the register R_SL_SEL0 ... R_SL_SEL7 depends on the PCM data rate:

PCM data rate	Value
PCM30	0x1F
PCM64	0x3F
PCM128	0x7F

Please note that time slot 0 for PCM128 can only be used with V_SH_SEL0 ... V_SH_SEL7 = 0 (SHAPE 0) in the registers R_SL_SEL0 ... R_SL_SEL7.

R_SL_SEL1 **(write only)** 0x15

Slot selection register for pin F1_1

This multi-register is selected with bitmap V_PCM_ADDR = 1 of the register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_1 is disabled.

Bits	Reset Value	Name	Description
6..0	0x7F	**V_SL_SEL1**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_1. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL1**	**Shape selection** '0' = use shape 0 set by R_SH0L and R_SH0H registers '1' = use shape 1 set by R_SH1L and R_SH1H registers

R_SL_SEL2	(write only)	0x15

Slot selection register for pin F1_2

This multi-register is selected with bitmap V_PCM_ADDR = 2 of the register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_2 is disabled.

Bits	Reset Value	Name	Description
6..0	0x7F	**V_SL_SEL2**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_2. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL2**	**Shape selection** '0' = use shape 0 set by R_SH0L and R_SH0H registers '1' = use shape 1 set by R_SH1L and R_SH1H registers

R_SL_SEL3	(write only)	0x15

Slot selection register for pin F1_3

This multi-register is selected with bitmap V_PCM_ADDR = 3 of the register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_3 is disabled.

Bits	Reset Value	Name	Description
6..0	0x7F	**V_SL_SEL3**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_3. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL3**	**Shape selection** '0' = use shape 0 set by R_SH0L and R_SH0H registers '1' = use shape 1 set by R_SH1L and R_SH1H registers

R_SL_SEL4	(write only)	0x15

Slot selection register for pin F1_4

This multi-register is selected with bitmap V_PCM_ADDR = 4 of the register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_4 is disabled.

Bits	Reset Value	Name	Description
6..0	0x7F	**V_SL_SEL4**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_4. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL4**	**Shape selection** '0' = use shape 0 set by R_SH0L and R_SH0H registers '1' = use shape 1 set by R_SH1L and R_SH1H registers

R_SL_SEL5	(write only)	0x15

Slot selection register for pin F1_5

This multi-register is selected with bitmap V_PCM_ADDR = 5 of the register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_5 is disabled.

Bits	Reset Value	Name	Description
6..0	0x7F	**V_SL_SEL5**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_5. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL5**	**Shape selection** '0' = use shape 0 set by R_SH0L and R_SH0H registers '1' = use shape 1 set by R_SH1L and R_SH1H registers

R_SL_SEL6	(write only)	0x15

Slot selection register for pin F1_6

This multi-register is selected with bitmap V_PCM_ADDR = 6 of the register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_6 is disabled.

Bits	Reset Value	Name	Description
6..0	0x7F	**V_SL_SEL6**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_6. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL6**	**Shape selection** '0' = use shape 1 set by R_SH0L and R_SH0H registers '1' = use shape 1 set by R_SH1L and R_SH1H registers

R_SL_SEL7	(write only)	0x15

Slot selection register for pin F1_7

This multi-register is selected with bitmap V_PCM_ADDR = 7 of the register R_PCM_MD0.

Note: By setting all 8 bits to '1' pin F1_7 is disabled.

Bits	Reset Value	Name	Description
6..0	0x7F	**V_SL_SEL7**	**PCM time slot selection** The selected slot number is V_SL_SEL1 +1 for F1_7. Slot number 0 is selected with the maximum slot number of the selected PCM speed.
7	1	**V_SH_SEL7**	**Shape selection** '0' = use shape 0 set by R_SH0L and R_SH0H registers '1' = use shape 1 set by R_SH1L and R_SH1H registers

R_PCM_MD1	(write only)	0x15

PCM mode, register 1

This multi-register is selected with bitmap V_PCM_ADDR = 9 of the register R_PCM_MD0.

Bits	Reset Value	Name	Description
0	0	**V_CODEC_CON**	**CODEC connection scheme** '0' = CODEC enable signals on F1_0 ...F1_7 '1' = SHAPE 0 pulse on pin SHAPE0, SHAPE 1 pulse on pin SHAPE1 and CODEC count on F_Q0 ...F_Q6 for up to 128 external CODECs.
1	0	**(reserved)**	Must be '0'.
3..2	0	**V_PLL_ADJ**	**DPLL adjust speed** '00' = C4IO clock is adjusted in the last time slot of PCM frame 4 times by one half clock cycle of PCM clock '01' = C4IO clock is adjusted in the last time slot of PCM frame 3 times by one half clock cycle of PCM clock '10' = C4IO clock is adjusted in the last time slot of PCM frame twice by one half clock cycle of PCM clock '11' = C4IO clock is adjusted in the last time slot of PCM frame once by one half clock cycle of PCM clock **Note:** Internal PCM clock is 16.384 MHz nominell
5..4	0	**V_PCM_DR**	**PCM data rate** '00' = 2 MBit/s (C4IO is 4.096 MHz, 32 time slots) '01' = 4 MBit/s (C4IO is 8.192 MHz, 64 time slots) '10' = 8 MBit/s (C4IO is 16.384 MHz, 128 time slots) '11' = unused
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.
7		**(reserved)**	Must be '0'.

R_PCM_MD2	(write only)	0x15

PCM mode, register 2

This multi-register is selected with bitmap V_PCM_ADDR = 0xA of the register R_PCM_MD0.

Bits	Reset Value	Name	Description
0		**(reserved)**	Must be '0'.
1	0	**V_SYNC_PLL**	**SYNC_O with internal PLL output** '0' = V_SYNC_OUT is used for synchronization '1' = SYNC_O has a frequency of the internal PLL output signal C4O divided by 8 (512 kHz, 1024 kHz or 2048 kHz depending on the PCM data rate)
2	0	**V_SYNC_SRC**	**PCM PLL synchronization source selection** '0' = E1 interface (see R_SYNC_CTRL for further sync configuration) '1' = SYNC_I input 8 kHz
3	0	**V_SYNC_OUT**	**SYNC_O output selection** '0' = E1 interface '1' = SYNC_I is connected to SYNC_O
5..4		**(reserved)**	Must be '00'.
6	0	**V_ICR_FR_TIME**	**Increase PCM frame time** This bit is only valid if V_EN_PLL is set. '0' = PCM frame time is reduced as selected by the bitmap V_PLL_ADJ of the R_PCM_MD1 register '1' = PCM frame time is increased as selected by the bitmap V_PLL_ADJ of the R_PCM_MD1 register
7	0	**V_EN_PLL**	**PLL enable** '0' = normal operation '1' = enable PCM PLL adjustment (can be used to make synchronization by software if no sync source is available)

R_SH0L	(write only)	0x15

CODEC enable signal SHAPE0, low byte

This multi-register is selected with bitmap V_PCM_ADDR = 0xC of the register R_PCM_MD0.

Bits	Reset Value	Name	Description
7..0	0	**V_SH0L**	**Shape bits 7 . . . 0** Every bit is used for 1/2 C4IO clock cycle.

R_SH0H	(write only)	0x15

CODEC enable signal SHAPE0, high byte

This multi-register is selected with bitmap V_PCM_ADDR = 0xD of the register R_PCM_MD0.

Bits	Reset Value	Name	Description
7..0	0	**V_SH0H**	**Shape bits 15 . . . 8** Every bit is used for 1/2 C4IO clock cycle. Bit 7 of V_SH0H defines the value for the rest of the period.

R_SH1L	(write only)	0x15

CODEC enable signal SHAPE1, low byte

This multi-register is selected with bitmap V_PCM_ADDR = 0xE of the register R_PCM_MD0.

Bits	Reset Value	Name	Description
7..0	0	**V_SH1L**	**Shape bits 7 . . . 0** Every bit is used for 1/2 C4IO clock cycle.

R_SH1H	(write only)	0x15

CODEC enable signal SHAPE1, high byte

This multi-register is selected with bitmap V_PCM_ADDR = 0xF of the register R_PCM_MD0.

Bits	Reset Value	Name	Description
7..0	0	V_SH1H	**Shape bits 15 … 8** Every bit is used for 1/2 C4IO clock cycle. Bit 7 of V_SH1H defines the value for the rest of the period.

6.4.2 Read only register

R_F0_CNTL	(read only)	0x18

F0IO pulse counter, low byte

Bits	Reset Value	Name	Description
7..0	0x00	V_F0_CNTL	**Low byte (bits 7 . . . 0) of the 125 μs time counter** This register should be read first to 'lock' the value of the R_F0_CNTH register until R_F0_CNTH has also been read.

R_F0_CNTH	(read only)	0x19

F0IO pulse counter, high byte

Bits	Reset Value	Name	Description
7..0	0	V_F0_CNTH	**High byte (bits 15 . . . 8) of the 125 μs time counter** The low byte must be read first (see register R_F0_CNTL)

Chapter 7

Pulse width modulation (PWM) outputs

Table 7.1: *Overview of the HFC-E1 PWM pins*

Number	Name	Description
95	PWM1	Pulse Width Modulator Output 1
96	PWM0	Pulse Width Modulator Output 0

Table 7.2: *Overview of the HFC-E1 PWM registers*

Address	Name	Page
0x38	R_PWM0	197
0x39	R_PWM1	197
0x46	R_PWM_MD	198

The HFC-E1 has two PWM output lines PWM0 and PWM1 with programmable output characteristic.

The output lines can be configured as open drain, open source and push / pull by setting V_PWM0_MD respectively V_PWM1_MD in the register R_PWM_MD.

7.1 Standard PWM usage

The duty cycle of the output signals can be set in the registers R_PWM0 and R_PWM1. The register value 0 generates an output signal which is permanently low. The register value defines the number of clock periods where the output signal is high during the cycle time

$$T = 256 \cdot \frac{1}{24.576\,\mathrm{MHz}} = 256 \cdot 40.69\,\mathrm{ns} = 10.42\,\mu\mathrm{s}$$

for the normal system clock 24.576 MHz.

The ouput signal of the PWM unit can be used for analog settings by using an external RC filter which generates a voltage that can be adapted by changing the PWM register value.

7.2 Alternative PWM usage

The PWM output lines can be programmed to generate a 16 kHz signal. This signal can be used as analog metering pulse for POTS interfaces. Each PWM output line can be switched to 16 kHz signal by setting V_PWM0_16KHZ or V_PWM1_16KHZ in the register R_RAM_MISC. In this case the output characteristic is also determined by the R_PWM_MD register settings.

7.3 Register description

7.3.1 Write only register

R_PWM0	(write only)	0x38

Modulator register for pin PWM0

Bits	Reset Value	Name	Description
7..0	0	V_PWM0	**PWM duty cycle** The value specifies the number of clock periods where the output signal of PWM0 is high during a 256 clock periods cycle, e.g. 0x00 = no pulse, always low 0x80 = 1/1 duty cycle 0xFF = 1 clock period low after 255 clock periods high

R_PWM1	(write only)	0x39

Modulator register for pin PWM1

Bits	Reset Value	Name	Description
7..0	0	V_PWM1	**PWM duty cycle** The value specifies the number of clock periods where the output signal of PWM1 is high during a 256 clock periods cycle, e.g. 0x00 = no pulse, always low 0x80 = 1/1 duty cycle 0xFF = 1 clock period low after 255 clock periods high

R_PWM_MD	(write only)	0x46

PWM output mode register

Bits	Reset Value	Name	Description
2..0	0	(reserved)	Must be '000'.
3	0	V_EXT_IRQ_EN	**External interrupt enable** '0' = normal operation '1' = external interrupt from GPI24 ...GPI31 enable **Note:** The GPI pins must be connected to a pull-up resistor to VDD. Any low input signal on one of the lines will generate an external interrupt.
5..4	0	V_PWM0_MD	**Output buffer configuration for pin PWM0** '00' =PWM output tristate (disable) '01' = PWM push / pull output '10' = PWM push to 0 only '11' = PWM pull to 1 only
7..6	0	V_PWM1_MD	**Output buffer configuration for pin PWM1** '00' = PWM output tristate (disable) '01' = PWM push / pull output '10' = PWM push to 0 only '11' = PWM pull to 1 only

Chapter 8

Multiparty audio conferences

Table 8.1: *Overview of the HFC-E1 conference registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x18	R_CONF_EN	204	0x14	R_CONF_OFLOW	206
0xD1	A_CONF	205			

8.1 Conference unit description

The HFC-E1 has a built in conference unit which allows up to 8 conferences with an arbitrary number of members each. The conference unit is located in the data stream going out to the PCM interface. So the normal outgoing data is replaced by the conference data. The number of conference members that can be combined to one conference is only limited by the number of the PCM time slots (maximum 64 members with 128 PCM time slots). Each time slot can only be part of one conference.

All PCM values combined to a conference are added in one 125 μs time intervall. Then for every conference member the added value for this member is substracted so that every member of a conference hears all the others but not himself. This is done on a alternating buffer scheme for every 125 μs time intervall.

To enable the conference unit the bit V_CONF_EN in the register R_CONF_EN must be set. If this is done there are additional accesses to the SRAM of HFC-E1 which reduces performance of the on-chip processor on the other hand. Thus conference cannot be used with 8 Mbit/s PCM data rate where 128 slots are used, except the chip operates with doubled input frequency.

To add a PCM time slot to a conference the slot number must be written into the register R_SLOT. If the time slot has not yet been linked to a HFC-channel this can be done by writing the HFC-channel number and the channels source / destination (input / output pins) to the A_SL_CFG register. Afterwards the conference number must be written into the A_CONF register. Noise suppression threshold and input attenuation level can be configured independently for each time slot.

To remove a time slot from a conference the time slot must be selected by writing its number to the R_SLOT register. Then 0x00 must be written into the A_CONF register.

8.2 Overflow handling

The data summation of the conference HFC-channels can cause signal overflows. The conference unit internally works with signed 16 bit words. In case of an overflow the amplitude value is limited to the maximum amplitude value.

Overflow conditions can be checked with the R_CONF_OFLOW register. Every bit of this register indicates that an overflow has occured in one of the eight corresponding conferences.

The more conference members are involved in a conference, the higher is the probability of signal overflows. In this case the signal attenuation can be reduced by the bitmap V_ATT_LEV in the register A_CONF. This can be done on-the-fly to improve the signal quality of a conference.

8.3 Conference including the E1 interface

As the conference unit is located in the PCM transmit data path, some additional explanations for conference members on the E1 interface have to be made.

Conference members can also be time slots of the E1 interface. In this case, a pair of transmit / receive PCM time slots have to be configured to loop back the data.

In detail, the conference signal on E1-channel[n,RX] gets assigned to PCM time slot[i,TX] and the signal is looped-back from slot[j,RX] to HFC-channel[m,TX]. The data transmission on HFC-channel[n,RX] and HFC-channel[m,TX] require one transmit and one receive FIFO to be enabled, although the FIFOs are not used to store data (see Section 3.4).

8.4 Conference setup example for CSM

The following example shows the register settings for a conference with three members. Two members are located on the PCM interface side while the other one is located on the E1 interface side. The example uses conference number 2. It is specified in Table 8.2.

Table 8.2: *Conference example specification*

Conference member		Connection		
E1 member	:	S/T interf. #1, RX B1	→	PCM slot[6,TX]
	:	S/T interf. #1, TX B1	←	PCM slot[6,RX]
1st PCM member	:	PCM slot[5,RX]	→	HFC-channel[6,TX]
	:	PCM slot[5,TX]	←	HFC-channel[6,TX]
2nd PCM member	:	PCM slot[20,RX]	→	HFC-channel[6,RX]
	:	PCM slot[20,TX]	←	HFC-channel[6,RX]

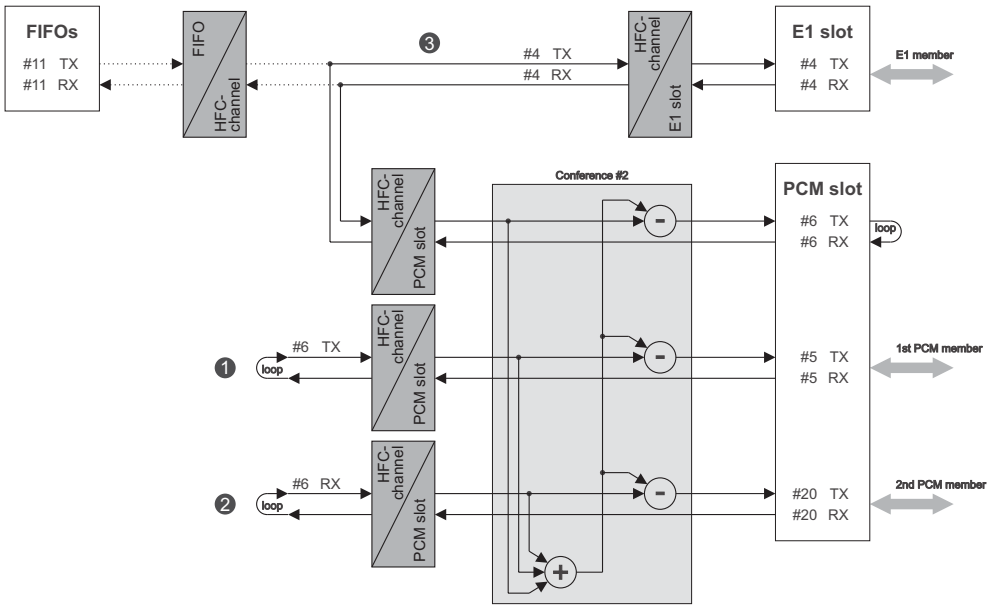

Figure 8.1: *Conference example*

Only two FIFOs are used in this example. Channel select mode should be selected to avoid unnecessary FIFO usage [1]. A PCM member allocates a single HFC-channel to establish the data loop via the switching buffer (see Fig. 3.3 and 3.3).

❶ A PCM conference member can be looped over an arbitrary HFC-channel. In this example HFC-channel[6,TX] is used for the first PCM conference member. The conference is enabled only on the transmit time slot of the PCM interface.

R_SLOT	: V_SL_DIR	= 0	(transmit slot)
	: V_SL_NUM	= 5	(slot #5)
A_SL_CFG[5,TX]	: V_CH_DIR1	= 0	(transmit HFC-channel)
	: V_CH_NUM1	= 6	(HFC-channel #6)
A_CONF[5,TX]	: V_CONF_NUM	= 2	(conference #2)
	: V_CONF_SL	= 1	(enable conference)
R_SLOT	: V_SL_DIR	= 1	(receive slot)
	: V_SL_NUM	= 5	(slot #5)
A_SL_CFG[5,RX]	: V_CH_DIR1	= 0	(transmit HFC-channel)
	: V_CH_NUM1	= 6	(HFC-channel #6)
A_CONF[5,RX]	: V_CONF_SL	= 0	(disable conference)

❷ The settings for the second PCM conference member is quite similar.

R_SLOT	: V_SL_DIR	= 0	(transmit slot)
	: V_SL_NUM	= 20	(slot #20)
A_SL_CFG[20,TX]	: V_CH_DIR1	= 1	(receive HFC-channel)
	: V_CH_NUM1	= 6	(HFC-channel #6)
A_CONF[20,TX]	: V_CONF_NUM	= 2	(conference #2)
	: V_CONF_SL	= 1	(enable conference)
R_SLOT	: V_SL_DIR	= 1	(receive slot)
	: V_SL_NUM	= 20	(slot #20)
A_SL_CFG[20,RX]	: V_CH_DIR1	= 1	(receive HFC-channel)
	: V_CH_NUM1	= 6	(HFC-channel #6)
A_CONF[20,RX]	: V_CONF_SL	= 0	(disable conference)

❸ Finally the E1 conference member must loop back its data via the PCM interface. This is normally done internally, i.e. the PCM output buffers are both disabled (see Chapter 6 for details). A pair of FIFOs is used to configure the PCM-to-E1 connection but no data is stored in these FIFOs.

[1] Remember that in *Simple Mode* FIFO numbers are equal to HFC-channel numbers. In the example four HFC-channels are enabled, so that in *Simple Mode* all FIFOs with the same number are blocked.

R_FIFO	: V_FIFO_DIR	=	0	(transmit FIFO)
	: V_FIFO_NUM	=	11	(FIFO #11)
A_CON_HDLC[11,TX]	: V_DATA_FLOW	=	'110'	(E1 → PCM)
A_CHANNEL[11,TX]	: V_CH_DIR0	=	0	(transmit HFC-channel)
	: V_CH_NUM0	=	4	(HFC-channel #4)
R_SLOT	: V_SL_DIR	=	1	(receive slot)
	: V_SL_NUM	=	6	(slot #6)
A_SL_CFG[6,RX]	: V_CH_DIR1	=	0	(transmit HFC-channel)
	: V_CH_NUM1	=	4	(HFC-channel #4)
A_CONF[6,RX]	: V_CONF_SL	=	0	(disable conference)
R_FIFO	: V_FIFO_DIR	=	1	(receive FIFO)
	: V_FIFO_NUM	=	11	(FIFO #11)
A_CON_HDLC[11,RX]	: V_DATA_FLOW	=	'110'	(E1 ← PCM)
A_CHANNEL[11,RX]	: V_CH_DIR0	=	1	(receive HFC-channel)
	: V_CH_NUM0	=	4	(HFC-channel #4)
R_SLOT	: V_SL_DIR	=	0	(transmit slot)
	: V_SL_NUM	=	6	(slot #6)
A_SL_CFG[6,TX]	: V_CH_DIR1	=	1	(receive HFC-channel)
	: V_CH_NUM1	=	4	(HFC-channel #4)
A_CONF[6,TX]	: V_CONF_NUM	=	2	(conference #2)
	: V_CONF_SL	=	1	(enable conference)

8.5 Register description

8.5.1 Write only registers

R_CONF_EN	(write only)	0x18
Conference mode register		

Bits	Reset Value	Name	Description
0	0	V_CONF_EN	**Global conference enable** '0' = disable '1' = enable
6..1		(reserved)	Must be '000000'.
7	0	V_ULAW	**Data coding of the conference unit** '0' = A-Law '1' = μ-Law

A_CONF [SLOT]	(write only)	0xD1

Conference parameter register for the selected PCM time slot

Before writing this array register the PCM time slot must be selected by register R_SLOT.

Bits	Reset Value	Name	Description
2..0	0	V_CONF_NUM	**Conference number** (0 . . . 7)
4..3	0	V_NOISE_SUPPR	**Noise suppression threshold** '00' = no noise suppression '01' = data values less or equal to 5 are set to 0 '10' = data values less or equal to 9 are set to 0 '11' = data values less or equal to 16 are set to 0
6..5	0	V_ATT_LEV	**Input attenuation level** '00' = 0 dB '01' = -3 dB '10' = -6 dB '11' = -9 dB
7		V_CONF_SL	**Conference enable for the selected PCM time slot** '0' = slot is not added to the conference '1' = slot is added to the conference

8.5.2 Read only registers

R_CONF_OFLOW	(read only)	0x14

Conference overflow indication register

Specifies the conference numbers where an overflow has occured. Reading this register clears the bits.

Bits	Reset Value	Name	Description
0	0	V_CONF_OFLOW0	Overflow occured in conference 0
1	0	V_CONF_OFLOW1	Overflow occured in conference 1
2	0	V_CONF_OFLOW2	Overflow occured in conference 2
3	0	V_CONF_OFLOW3	Overflow occured in conference 3
4	0	V_CONF_OFLOW4	Overflow occured in conference 4
5	0	V_CONF_OFLOW5	Overflow occured in conference 5
6	0	V_CONF_OFLOW6	Overflow occured in conference 6
7	0	V_CONF_OFLOW7	Overflow occured in conference 7

Chapter 9

DTMF controller

Table 9.1: *Overview of the HFC-E1 DTMF registers*

Write only registers:		
Address	**Name**	**Page**
0x1C	R_DTMF0	211
0x1D	R_DTMF1	212

9.1 DTMF detection engine

The transmission of dialed numbers on analog lines is normaly done by DTMF (Dual Tone Multi-Frequency). This means that pairs of two frequencies are used to determine one key of a keypad like shown in Table 9.2.

Table 9.2: *DTMF tones on a 16 keys keypad*

Keypad				Frequencies	
1	2	3	A	697	
4	5	6	B	770	low tones
7	8	9	C	852	(f/Hz)
*	0	#	D	941	
1209	1336	1477	1633	high tones (f/Hz)	

Thus there are 4 low tones and 4 high tones and therefore 16 combinations of 2 tones. Because the ISDN network has several interfaces to the old-fashioned POTS analog network, in-band number dialing with DTMF can take place. To decode this DTMF information the HFC-E1 has a built in DTMF detection engine.

The detection is done by the digital processing of the PCM input data by the so-called Goerzel Algorithm

$$W_{n+1} = K \cdot W_n - W_{n-1} + x \,, \tag{9.1}$$

where W_{n+1} is a coefficient calculated from the 2 previous coefficients W_n and W_{n-1}. The factor

$$K = 2\cos\left(2\pi \cdot \frac{f}{8000\,\mathrm{Hz}}\right)$$

is a constant for each frequency and x is a new PCM value every 125 μs. Equation (9.1) is calculated every 125 μs for 16 or 32 W_{n+1} values.

The start condition is $W_0 = W_{-1} = 0$.

After processing equation (9.1) for N times the real power amplitude is

$$A^2 = W_N^2 + W_{N-1}^2 - K \cdot W_N \cdot W_{N-1} \,. \tag{9.2}$$

The calculation of equation (9.1) is done for every new PCM sample value (for all 8 frequencies) every 125 μs. Optionally also the second harmonic (double frequency) is also investigated. The K factors are values concerning to the DTMF frequencies. If the DTMF calculation is implemented in integer arithmetic, it is useful to multiply K with 2^{14} to exploit the whole 16 bit value range. These K values are listed in Table 9.3.

The DTMF engine must be enabled by setting bit V_DTMF_EN in register R_DTMF0. How many iterations are calculated with the Goerzel algorithm is determined by the register

Table 9.3: *16-bit K factors for the DTMF calculation*

1st harmonic		2nd harmonic	
f/Hz	**$K \cdot 2^{14}$**	**f/Hz**	**$K \cdot 2^{14}$**
697	27 980	1406 *	14 739
770	26 956	1555 *	11 221
852	25 701	1704	7 549
941	24 219	1882	3 032
1209	19 073	2418	-10 565
1336	16 325	2672	-16 503
1477	13 085	2954	-22 318
1633	9 315	3266	-27 472

(*: These frequencies are modified to achieve a better detection compared with the high fundamental tones.)

value V_DTMF1 in the register R_DTMF1. A good compromise between bandwith of the Goerzel filter and the length of the investigation is a value of 102. A DTMF detection can be done on a continuous base. However then the reading of the calculated coefficients has to be done in a very short time intervall before the coefficients are cleared to zero for a new calculation. Is more convenient to set the V_DTMF_STOP bit of the register R_DTMF0. The DTMF engine is stopped then after each calculation of a set of coefficients and the V_DTMF_IRQ bit is set in the register R_IRQ_MISC. Then a software routine has time to read the coefficients out of HFC-E1. After this, a new calculation can be started. However some PCM samples (x values) can be lost.

The host processor should read the two W_N and W_{N-1} 16-bit coefficients for 8 or 16 frequencies for the desired channels. The coefficients are located in the SRAM memory of HFC-E1. The memory address is calculated by

$$\text{address} = \text{base address} + \text{frequency offset} + \text{channel offset} + \text{W-byte offset}\,. \qquad (9.3)$$

The individual address components are shown in Table 9.4.

If 32 channels are used, only the 8 fundamental frequencies can be detected. If only 16 channels are used, all 16 frequencies (1st and 2nd harmonic) can be detected.

For every frequency and every channel the power amplitude can be calculated with equation (9.2). This calculation is not implemented in the chip and has to take place in the host processor.

After a discrimination process and a balance check between 2 frequency candidates with the maximum power, the software can determine if there was a DTMF signal on the line or not. If there was a DTMF signal the tone pair is detected and so the dialed digit is decoded.

In case the existence of DTMF tones in an arbitrary voice signal has to be detected, it is helpfull to investigate not only the 8 DTMF tones but also their second harmonics. For DTMF tones the second harmonics should have no significant amplitude.

Table 9.4: *Memory address calculation for DTMF coefficients related to equation (9.3)*

base address	RAM size	address	RAM size	address
	32k	0x1000	128k	0x2000
			512k	0x2000

frequency offset	low tones	offset	high tones	offset
(1st harmonic)	697 Hz	0x00	1406 Hz	0x40
	770 Hz	0x80	1555 Hz	0xC0
	852 Hz	0x100	1704 Hz	0x140
	941 Hz	0x180	1882 Hz	0x1C0
(2nd harmonic)	1209 Hz	0x200	2418 Hz	0x240
	1336 Hz	0x280	2672 Hz	0x2C0
	1477 Hz	0x300	2954 Hz	0x340
	1633 Hz	0x380	3266 Hz	0x3C0

channel offset	number	offset	number	offset
	0	0x00	16	0x40
	1	0x04	17	0x44
	2	0x08	18	0x48
	3	0x0C	19	0x4C
	4	0x10	20	0x50
	5	0x14	21	0x54
	6	0x18	22	0x58
	7	0x1C	23	0x5C
	8	0x20	24	0x60
	9	0x24	25	0x64
	10	0x28	26	0x68
	11	0x2C	27	0x6C
	12	0x30	28	0x70
	13	0x34	29	0x74
	14	0x38	30	0x78
	15	0x3C	31	0x7C

W-byte offset	W_{N-1}	offset	W_N	offset
	low byte	0	low byte	2
	high byte	1	high byte	3

9.2 Register description

R_DTMF0	(write only)	0x1C
DTMF configuration register		

Bits	Reset Value	Name	Description
0	0	V_DTMF_EN	**Global DTMF enable** '0' = disable DTMF unit '1' = enable DTMF unit
1	0	V_HARM_SEL	**Harmonics selection** 2nd harmonics of the DTMF frequencies can be enabled to improve the detection algorithm. '0' = 8 frequencies in 32 channels (only 1st harmonics are processed) '1' = 16 frequencies in 16 channels (1st and 2nd harmonics are processed)
2	0	V_DTMF_RX_CH	**DTMF data source** '0' = transmit buffer of the flow controller (HFC-channels to PCM time slot) are used for DTMF detection '1' = receive buffer of the flow controller (HFC-channels from PCM time slot) are used for DTMF detection
3	0	V_DTMF_STOP	**Stop DTMF unit** '0' = continuous DTMF processing '1' = DTMF processing stops after n processed samples
4	0	V_CHBL_SEL	**HFC-Channel block selection** HFC-Channel block selection (only if 32 channels are used) '0' = lower 16 channels (0 . . . 15) '1' = upper 16 channels (16 . . . 31)
5		(reserved)	Must be '0'.
6	0	V_RESTART_DTMF	**Restart DTMF prosessing** '0' = no action '1' = enables new DTMF calculation phase after stop, automatically cleared
7	0	V_ULAW_SEL	**Data coding for DTMF detection** '0' = A-Law code '1' = μ-Law code

R_DTMF1	(write only)	0x1D

Number of samples

This register defines the number of samples which are calculated in the recursive part of the Goertzel filter.

Bits	Reset Value	Name	Description
7..0	0	**V_DTMF1**	**Number of samples** V_DTMF1 +1 PCM values generate 1 pair of DTMF coefficients (1 PCM value every 125 μs).

Chapter 10

BERT

Table 10.1: *Overview of the HFC-E1 BERT registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x1B	R_BERT_WD_MD	215	0x17	R_BERT_STA	216
0xFF	A_IRQ_MSK	239	0x1A	R_BERT_ECL	216
			0x1B	R_BERT_ECH	217

10.1 BERT functionality

Bit Error Rate Test (BERT) is a very important test for communication lines. The bit error rate should be as low as possible. Increasing bit error rate is an early indication of a malfunction of components or the communication wire link itself.

HFC-E1 includes a high performance pseudo random bit generator (PRBG) and a pseudo random bit receiver with automatic synchronization capability. Error rate can be checked by the also implemented Bit Error counter (BERT counter).

The PRBG can be set to a variety of different pseudo random bit patterns. With the bit pattern V_PAT_SEQ in register R_BERT_WD_MD the transmit and receive detector can be set to the trivial always '0' or always '1' pattern as well to well known patterns described in ITU-T O.150 and O.151 specifications.

In every transmit HFC-channel the HDLC or transparent data is overwritten by bits from the PRBG if V_BERT_EN in the register A_IRQ_MSK[FIFO] is set to '1'. The random data is only generated when the FIFO is processing data. So if subchannel processing is enabled the PRBG is only enabled for less than 8 bits. Next PRGB bits are generated in the next FIFO where a HFC-channel is processed and V_BERT_EN is set. The receive detector can function properly only when the same receive FIFOs connected to the same E1 time slots are enabled for BERT in receive direction as on the transmit FIFOs of the remote E1 interface side.

The receive detector has an auto synchonization capability and also is enabled to automatic detect an inverted BERT pattern. The auto synchronization only works with bit error rates of less than $4 \cdot 10^{-2}$. If the error rate is higher synchronization will not be achieved. A found synchronization is reported by V_BERT_SYNC $= 1$ in register R_BERT_STA. If the received pattern is inverted also V_BERT_INV_DATA is set.

A 16 bit BERT error count is available by reading the registers R_BERT_ECL and R_BERT_ECH. The counter is reset when the R_BERT_ECL register is read.

To test a connection and the error detection of the BERT error counter on the receiver side of an E1 link a BERT error can be generated. Setting the V_BERT_ERR generates one wrong BERT bit in the outgoing data stream.

10.2 Register description

10.3 Write only register

R_BERT_WD_MD	(write only)	0x1B
Bit error rate test (BERT) and watchdog mode		

Bits	Reset Value	Name	Description
2..0	0	V_PAT_SEQ	**Pattern for BERT** '000' = continuous '0' pattern '001' = continuous '1' pattern '010' = pseudo random pattern seq. $2^9 - 1$ '011' = pseudo random pattern seq. $2^{10} - 1$ '100' = pseudo random pattern seq. $2^{15} - 1$ '101' = pseudo random pattern seq. $2^{20} - 1$ '110' = pseudo random pattern seq. $2^{20} - 1$, but maximal 14 bits are zero '111' = pseudo random pattern seq. $2^{23} - 1$ **Note:** This sequences are defined in ITU-T O.150 and O.151 specifications.
3	0	V_BERT_ERR	**BERT error** Generates 1 error bit in the BERT data stream '0' = no error generation '1' = generates one error bit This bit is cleared automatically.
4		(reserved)	Must be '0'.
5	0	V_AUTO_WD_RES	**Automatically watchdog timer reset** '0' = watchdog is only reset by V_WD_RES '1' = watchdog is reset after every access to the chip
6		(reserved)	Must be '0'.
7	0	V_WD_RES	**Watchdog timer reset** '0' = no action '1' = manual watchdog timer reset This bit is automatically cleared.

10.4 Read only register

R_BERT_STA	(read only)	0x17

Bit error rate test status

Bits	Reset Value	Name	Description
3..0	0	(reserved)	
4	0	V_BERT_SYNC	**BERT synchronization status** '0' = BERT not synchronized to input data '1' = BERT sync to input data
5	0	V_BERT_INV_DATA	**BERT data inversion** '0' = BERT receives normal data '1' = BERT receives inverted data
7..6	0	(reserved)	

R_BERT_ECL	(read only)	0x1A

BERT error counter, low byte

Bits	Reset Value	Name	Description
7..0	0	V_BERT_ECL	**Bits 7 . . . 0 of the BERT error counter** This register should be read first to 'lock' the value of the R_BERT_ECH register until R_BERT_ECH has also been read. **Note:** The BERT counter is cleared after reading this register.

R_BERT_ECH	(read only)	0x1B

BERT error counter, high byte

Bits	Reset Value	Name	Description
7..0	0	V_BERT_ECH	**Bits 15 … 8 of the BERT error counter** **Note:** Low byte must be read first (see register R_BERT_ECL).

Chapter 11

Auxiliary interface

(For an overview of the auxiliary interface pins see the comparison of first and second pin function in Table 11.2 on page 220.)

Table 11.1: *Overview of the HFC-E1 auxiliary bridge registers*

Write only registers:

Address	Name	Page
0x02	R_BRG_PCM_CFG	226
0x45	R_BRG_CTRL	227
0x47	R_BRG_MD	228
0x48	R_BRG_TIM0	229
0x49	R_BRG_TIM1	229
0x4A	R_BRG_TIM2	230
0x4B	R_BRG_TIM3	230
0x4C	R_BRG_TIM_SEL01	231
0x4D	R_BRG_TIM_SEL23	231
0x4E	R_BRG_TIM_SEL45	232
0x4F	R_BRG_TIM_SEL67	232

The HFC-E1 has an auxiliary interface which is designed for connecting up to 8 external devices with the universal bus interface. This bridge functionality supports 8 bit data bus and up to 12 address lines. The auxiliary-to-host bridge is typically used to realize a PCI bridge or a PCMCIA bridge for external devices. The auxiliary interface is implemented parallel to the optional external SRAM interface, so it can only be used if no external SRAM is connected to the HFC-E1.

11.1 Interface pins

The auxiliary bridge must be switched on with V_BRG_EN = 1 in the register V_BRG_EN. Table 11.2 shows that the bridge functionality uses some HFC-E1 pins in their second function. As the first pin functions are associated to the SRAM interface, the external SRAM must be disabled when the bridge functionality is switched on.

Table 11.2: *HFC-E1 pins of the auxiliary bridge*

Pin	1st function	2nd function
54 . . . 61	SRA0 . . . SRA7	BRG_A0 . . . BRG_A7
63 . . . 66	SRA8 . . . SRA11	BRG_A8 . . . BRG_A11
67 . . . 73	SRA12 . . . SRA18	/BRG_CS0 . . . /BRG_CS6
74	NC	/BRG_CS7
77 . . . 84	SRD0 . . . SRD7	BRG_D0 . . . BRG_D7
85	/SR_WR	/BRG_WR
87	/SR_OE	/BRG_RD

External devices can be accessed by an address bus with up to 12 lines, an 8 bit data bus, up to 8 chip select signals and two control lines supporting Motorola- or Siemens/Intel-Style interfaces.

Important !

As the auxiliary interface and the external SRAM use the same chip pins, it is strongly recommended not to enable the external SRAM and the bridge functionality at the same time!

Extract from the register descriptions:

Register	Bit	Description
R_CTRL	V_EXT_RAM	The internal SRAM is switched off when external SRAM is used. '0' = internal SRAM is used in lower 32 kByte address space '1' = external SRAM is used
R_BRG_PCM_CFG	V_BRG_EN	'0' = disable (external SRAM can be used) '1' = enable (external SRAM is disabled)

Both register bits are zero by default.

11.2 Various mode selections

The host-to-auxiliary bridge can be configured into various modes which define the behavior of the bridge. The overview of these modes is illustrated in Figure 11.1 and will be described in the following sections.

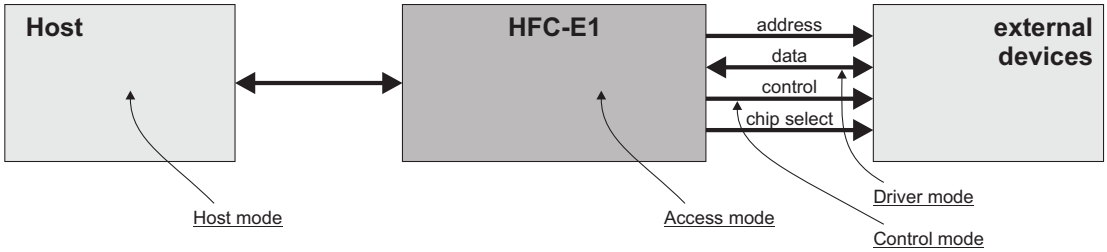

Figure 11.1: *Points of contact of the various bridge modes*

11.2.1 Driver mode

The behavior of the data bus of the auxiliary bridge can be modified by V_BRG_MD of the register R_BRG_PCM_CFG. A '0' defines that the bus BRG_D0 . . . BRG_D7 is tristated when no bridge access is performed and a '1' defines that the bus is only tristated when a read access is performed.

11.2.2 Control mode

The register R_BRG_MD defines for each chip select the style of the access.

The bit value '0' executes an access to the external device in Siemens/Intel style. Alternatively an access in Motorola style can be selected with '1'.

Table 11.3: *Control mode*

/IOR /DS	/IOW R/$\overline{W}$	/CS	ALE	Operation	Access style
0	1	0	1	read data	Motorola
0	0	0	1	write data	Motorola
0	1	0	0	read data	Siemens/Intel
1	0	0	0	write data	Siemens/Intel

11.2.3 Access mode

The access mode is controlled by the two bit M0 and M1. A normal chip access is done with M[1..0] = '00'.

The CIP must be written with one 16 bit access to use the auxiliary interface.

Data write

Data write requires M[1..0] = '01' and is always a posted write. An internal write register is written by the host write access. Then the data is transferred to the auxiliary interface.

Data read

For read operations the auxiliary bridge uses an internal data buffer. The read access can be performed in three different modes.

Normal read: (M[1..0] = '01') In *normal read* mode a host read access is immediately transferred to the auxiliary interface. The host read access must be long enough to pass the data from the auxiliary interface to the host data bus. Big delays may be involved.

Posted read: (M[1..0] = '10') Depending on the selected timing for the desired bridge read operation, the *normal read* may not meet the timing requirements of the selected host interface. To ensure timing constraints when using slow devices the *posted read* mode can be selected. In this mode the data of the internal buffer is immediately read by the host interface. Afterwards a read on the auxiliary interface is initiated to fill the buffer again. So the data of the first host read access should be ignored.

Last read: (M[1..0] = '11') The last buffered data byte can be read in *last read* mode. The buffered data is transferred to the host interface and no read access is performed by the auxiliary bridge afterwards.

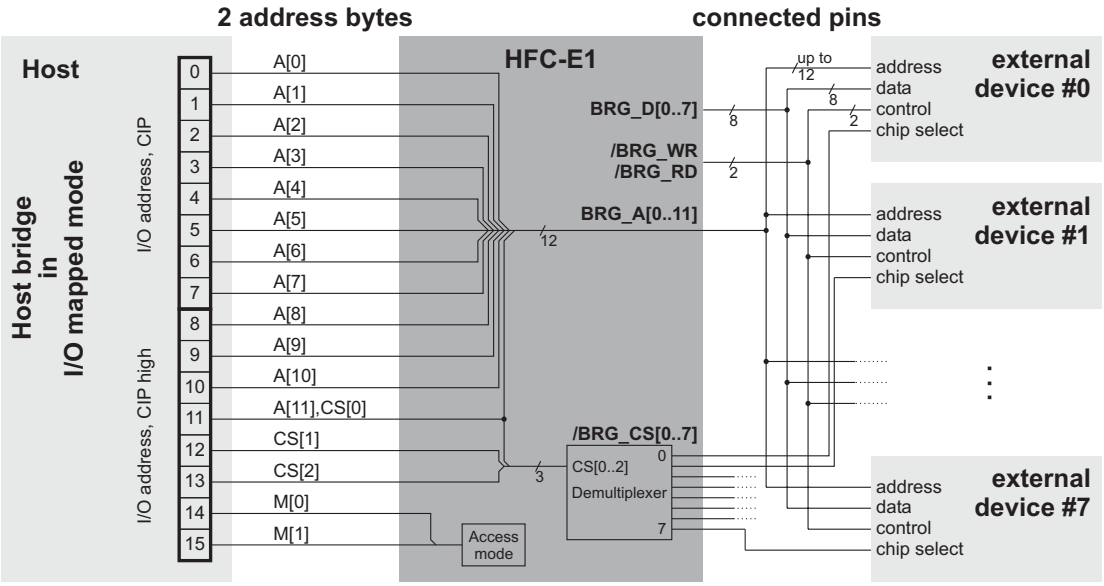

Figure 11.2: *Host bridge structure in I/O mapped mode*

It is possible to perfom byte, word or double word accesses. Word or double word are splitted into two or four consecutive byte accesses. The accesses are all executed on the same address. Thus word and double word accesses are useful for FIFO style buffered data transfers from or to an external device.

11.2.4 Host mode

Auxiliary-to-host accesses can be performed in two ways. In I/O mapped mode two CIP bytes must be programmed to execute read and write accesses. The second way uses the memory mapped mode and the register R_BRG_CTRL.

Bridge access in I/O mapped mode

This mode is supported for PCI I/O mode, PCMCIA, ISA PnP and SPI modes.

The host-to-auxiliary bridge uses two CIP bytes for read and write access control in I/O mapped mode. Figure 11.2 shows the bit mapping of these bytes. Please see Figure 11.2 on page 223 concerning the CIP bytes. If V_BRG_EN is set in the register R_BRG_PCM_CFG all CIP writes must be 16 bit writes.

As A[11] and CS[0] are located on the same CIP bit, it is either possible to use more than 4 external devices with 11 bit address bus width or to use up to 4 external devices with full 12 bit address bus width.

With 12 bit address space a small external circuitry is required to connect the external devices to the HFC-E1 chip select lines. In detail, /BRG_CS0 and /BRG_CS1 must be OR-ed to select the first device, /BRG_CS2 and /BRG_CS3 must be OR-ed to select the second device, and so on.

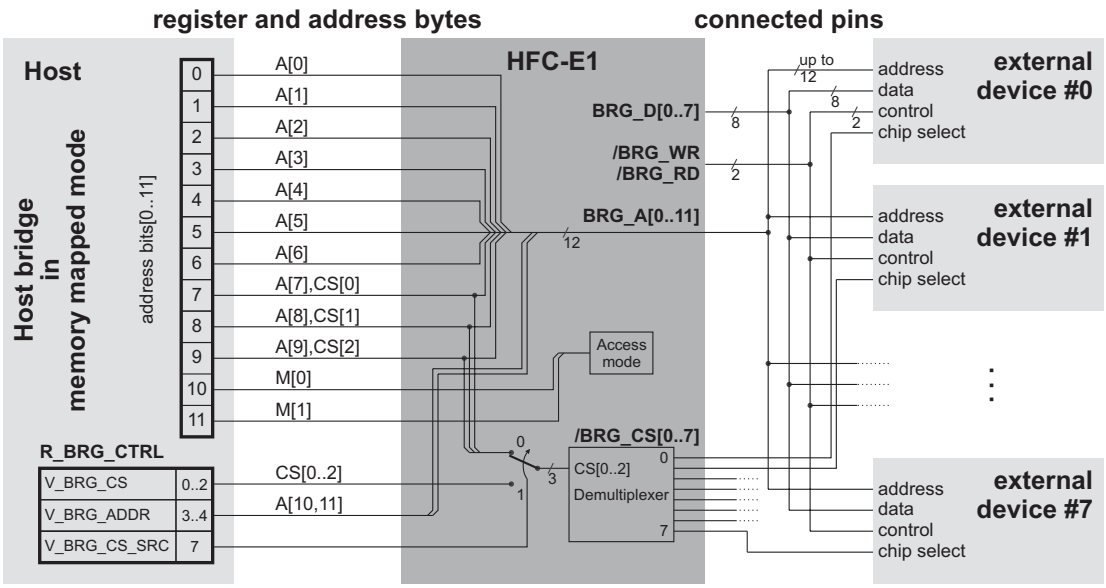

Figure 11.3: *Host bridge structure in memory mapped mode*

Bridge access in memory mapped mode

This mode is supported for PCI memory mapped mode and processor mode.

In memory mapped mode the control register R_BRG_CTRL can be used to perfom read and write accesses with a large address space. External devices with up to 10 address lines do not require this register. If R_BRG_CTRL is not used, the exact number of available address lines depends on the number of external devices. An overview of this functionality is given in Figure 11.3.

V_BRG_CS_SRC of the register R_BRG_CTRL selects the source of the chip select signals. By default the address lines 7 . . . 9 are taken.

1. If the external devices have not more than 7 address lines, the register R_BRG_CTRL is not necessary for bridge accesses. The bridge operation can be performed with 12 address bits as shown in Figure 11.3. Up to 8 external devices can be connected to the HFC-E1.

2. External devices with 8 . . . 10 address lines take one, two or even all chip select lines CS[0..2] from the address specification bits. The number of chip select output signals on the pins /BRG_CS0 . . . /BRG_CS7 is reduced appropriately. If A[7] . . . A[9] are used in parallel to chip select signals, the bit V_BRG_CS_SRC must be set in the register R_BRG_CTRL.

3. The full 12 bit address space can be used with the bitmap V_BRG_ADDR of the register R_BRG_CTRL. The address bits A[10] and A[11] have to be specified there.

11.3 Timing definitions

The timing requirements of the connected external devices can be fulfilled by programming different timing configurations. Four different read and write timings can be programmed in the registers R_BRG_TIM0 . . . R_BRG_TIM3.

The timings are defined by writing the number of idle clock cycles for an access to the bitmaps V_BRG_TIM0_IDLE . . . V_BRG_TIM3_IDLE of the registers R_BRG_TIM0 . . . R_BRG_TIM3. The number of active clock cycles are defined in the bitmaps V_BRG_TIM0_CLK . . . V_BRG_TIM3_CLK of the same registers.

The timing can be configured for each chip select and read / write operation independently by programming the registers R_BRG_TIM_SEL01 . . . R_BRG_TIM_SEL67.

11.4 Register description

R_BRG_PCM_CFG	(write only)	0x02
Auxiliary bridge and PCM configuration register		

Bits	Reset Value	Name	Description
0	0	V_BRG_EN	**Auxiliary bridge enable** '0' = disable (external SRAM can be used) '1' = enable (external SRAM is disabled)
1	0	V_BRG_MD	**Auxiliary bridge data lines mode** Mode of the data bus pins SRD0 SRD7. '0' = tristate when no bridge access '1' = only tristate when data is read
4..2		(reserved)	Must be '000'.
5	0	V_PCM_CLK	**Clock of the PCM module** '0' = system clock / 2 '1' = system clock / 4 PCM clock must be 16.384 MHz, system clock is normaly 24.576 MHz.
7..6	0	(reserved)	Must be '00'.

R_BRG_CTRL	(write only)	0x45

Access control register for the auxiliary brigde in memory mapped mode

Note: This register is not used in I/O mapped mode.

Bits	Reset Value	Name	Description
2..0	0	**V_BRG_CS**	**Chip select** This bitmap controls the chip select pins. '000' = /BRG_CS0 '001' = /BRG_CS1 ... '111' = /BRG_CS7
4..3	0	**V_BRG_ADDR**	**High bits of address** Address bits A[10] and A[11] of the auxiliary bridge (pins BRG_A10 and BRG_A11).
6..5		**(reserved)**	Must be '00'.
7	0	**V_BRG_CS_SRC**	**Chip select source** '0' = address bits A[9..7] are used for chip select CS[2..0] '1' = V_BRG_CS is used for chip select, address bits A[9..7] are used for address selection

R_BRG_MD	(write only)	0x47

Control mode

Select Siemens / Intel or Motorola style for external access ('0' = Siemens / Intel, '1' = Motorola).

Bits	Reset Value	Name	Description
0	0	V_BRG_MD0	Bridge access mode for the chip connected to pin /BRG_CS0
1	0	V_BRG_MD1	Bridge access mode for the chip connected to pin /BRG_CS1
2	0	V_BRG_MD2	Bridge access mode for the chip connected to pin /BRG_CS2
3	0	V_BRG_MD3	Bridge access mode for the chip connected to pin /BRG_CS3
4	0	V_BRG_MD4	Bridge access mode for the chip connected to pin /BRG_CS4
5	0	V_BRG_MD5	Bridge access mode for the chip connected to pin /BRG_CS5
6	0	V_BRG_MD6	Bridge access mode for the chip connected to pin /BRG_CS6
7	0	V_BRG_MD7	Bridge access mode for the chip connected to pin /BRG_CS7

R_BRG_TIM0	(write only)	0x48

Auxiliary bridge timing configuration register for timing 0

Bits	Reset Value	Name	Description
3..0	0	V_BRG_TIM0_IDLE	**Idle cycles** Number of idle system clock cycles for read / write signal
7..4	0	V_BRG_TIM0_CLK	**Active cycles** Number of active system clock cycles for read / write signal

R_BRG_TIM1	(write only)	0x49

Auxiliary bridge timing configuration register for timing 1

Bits	Reset Value	Name	Description
3..0	0	V_BRG_TIM1_IDLE	**Idle cycles** Number of idle clock cycles for read / write signal
7..4	0	V_BRG_TIM1_CLK	**Active cycles** Number of active clock cycles for read / write signal

R_BRG_TIM2	(write only)	0x4A

Auxiliary bridge timing configuration register for timing 2

Bits	Reset Value	Name	Description
3..0	0	V_BRG_TIM2_IDLE	**Idle cycles** Number of idle clock cycles for read / write signal
7..4	0	V_BRG_TIM2_CLK	**Active cycles** Number of active clock cycles for read / write signal

R_BRG_TIM3	(write only)	0x4B

Auxiliary bridge timing configuration register for timing 3

Bits	Reset Value	Name	Description
3..0	0	V_BRG_TIM3_IDLE	**Idle cycles** Number of idle clock cycles for read / write signal
7..4	0	V_BRG_TIM3_CLK	**Active cycles** Number of active clock cycles for read / write signal

R_BRG_TIM_SEL01	(write only)	0x4C

Timing selection for bridge device connected to /BRG_CS0 and /BRG_CS1

Every selection uses a timing defined in R_BRG_TIM0 . . . R_BRG_TIM3.

Bits	Reset Value	Name	Description
1..0	0	V_BRG_WR_SEL0	WR-timing selection for the chip connected to pin /BRG_CS0
3..2	0	V_BRG_RD_SEL0	RD-timing selection for the chip connected to pin /BRG_CS0
5..4	0	V_BRG_WR_SEL1	WR-timing selection for the chip connected to pin /BRG_CS1
7..6	0	V_BRG_RD_SEL1	RD-timing selection for the chip connected to pin /BRG_CS1

R_BRG_TIM_SEL23	(write only)	0x4D

Timing selection for bridge device connected to /BRG_CS2 and /BRG_CS3

Every selection uses a timing defined in R_BRG_TIM0 . . . R_BRG_TIM3.

Bits	Reset Value	Name	Description
1..0	0	V_BRG_WR_SEL2	WR-timing selection for the chip connected to pin /BRG_CS2
3..2	0	V_BRG_RD_SEL2	RD-timing selection for the chip connected to pin /BRG_CS2
5..4	0	V_BRG_WR_SEL3	WR-timing selection for the chip connected to pin /BRG_CS3
7..6	0	V_BRG_RD_SEL3	RD-timing selection for the chip connected to pin /BRG_CS3

R_BRG_TIM_SEL45	(write only)	0x4E

Timing selection for bridge device connected to /BRG_CS4 and /BRG_CS5

Every selection uses a timing defined in R_BRG_TIM0 . . . R_BRG_TIM3.

Bits	Reset Value	Name	Description
1..0	0	V_BRG_WR_SEL4	WR-timing selection for the chip connected to pin /BRG_CS4
3..2	0	V_BRG_RD_SEL4	RD-timing selection for the chip connected to pin /BRG_CS4
5..4	0	V_BRG_WR_SEL5	WR-timing selection for the chip connected to pin /BRG_CS5
7..6	0	V_BRG_RD_SEL5	RD-timing selection for the chip connected to pin /BRG_CS5

R_BRG_TIM_SEL67	(write only)	0x4F

Timing selection for bridge device connected to /BRG_CS6 and /BRG_CS7

Every selection uses a timing defined in R_BRG_TIM0 . . . R_BRG_TIM3.

Bits	Reset Value	Name	Description
1..0	0	V_BRG_WR_SEL6	WR-timing selection for the chip connected to pin /BRG_CS6
3..2	0	V_BRG_RD_SEL6	RD-timing selection for the chip connected to pin /BRG_CS6
5..4	0	V_BRG_WR_SEL7	WR-timing selection for the chip connected to pin /BRG_CS7
7..6	0	V_BRG_RD_SEL7	RD-timing selection for the chip connected to pin /BRG_CS7

Chapter 12

Clock, reset, interrupt, timer and watchdog

Table 12.1: *Overview of the HFC-E1 clock pins*

Number	Name	Description
90	OSC_IN	Oscillator Input Signal
91	OSC_OUT	Oscillator Output Signal
92	CLK_MODE	Clock Mode

Table 12.2: *Overview of the HFC-E1 reset, timer and watchdog registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x11	R_IRQMSK_MISC	236	0x10	R_IRQ_OVIEW	240
0x13	R_IRQ_CTRL	237	0x11	R_IRQ_MISC	241
0x1A	R_TI_WD	238	0x1C	R_STATUS	242
0xFF	A_IRQ_MSK	239	0xC8	R_IRQ_FIFO_BL0	243
			0xC9	R_IRQ_FIFO_BL1	244
			0xCA	R_IRQ_FIFO_BL2	245
			0xCB	R_IRQ_FIFO_BL3	246
			0xCC	R_IRQ_FIFO_BL4	247
			0xCD	R_IRQ_FIFO_BL5	248
			0xCE	R_IRQ_FIFO_BL6	249
			0xCF	R_IRQ_FIFO_BL7	250

12.1 Clock

The clock generation circuitry of the HFC-E1 is shown in Figure 12.1. Two different crystal frequencies can be used. Pin CLK_MODE must be set as shown in Table 12.3 to ensure a system clock of 32,768 MHz.

E1 applications need exactly 32,768 MHz . It is recommended to ensure an accuracy of ± 50 ppm.

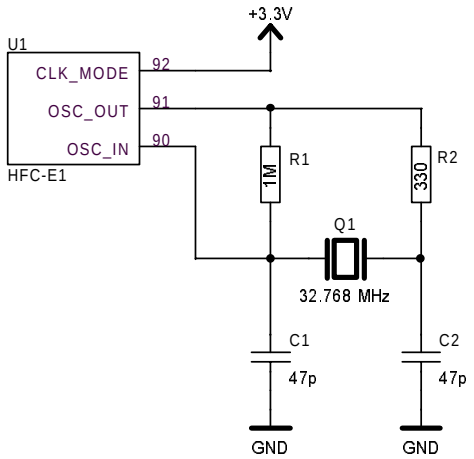

Figure 12.1: *Standard HFC-E1 quartz circuitry*

Table 12.3: *Quartz selection*

Crystal frequency	CLK_MODE	System clock f_{CLKI}
32,768 MHz	'1'	32,768 MHz
65,536 MHz	'0'	32,768 MHz

12.2 Reset

HFC-E1 has a level sensitive RESET input. This is low active in PCI mode (pin name RST#) and high active in all other modes (pin name RESET). The MODE0 / MODE1 pins must be valid during RESET and /SPISEL must be '1' (inactive). After RESET HFC-E1 enters an initialization sequence.

The HFC-E1 has 4 different software resets. The FIFO registers, PCM registers and E1 registers can be reset independently with the bits of the register R_CIRM which are listed in Table 12.4. The reset bits must be cleared by software.

Information about the registers reset by the different resets can be found in the register list on pages 16 and 14.

Table 12.4: *HFC-E1 reset groups*

Reset name	Reset group	Register bit	Description
Soft Reset	0	V_SRES	Reset for FIFO, PCM and E1 registers of the HFC-E1. Soft reset is the same as reset of all partial reset registers.
HFC Reset	1	V_HFCRES	Reset for all FIFO registers of the HFC-E1.
PCM Reset	2	V_PCMRES	Reset for all PCM registers of the HFC-E1.
E1 Reset	3	V_E1RES	Reset for all E1 registers of the HFC-E1.
Hardware reset	H	–	Hardware reset initiated by RESET input pin

12.3 Interrupt

HFC-E1 is equipped with a maskable interrupt engine. A big variety of interrupt sources can be enabled and disabled. All interrupts except FIFO interrupts are reported independently of masking the interrupt or not. Only mask enabled interrupts are used to generate an interrupt on the interrupt pin of the HFC-E1. Reading the interrupt status register resets the bits. Interrupt bits set during the reading are reported at the next reading of the interrupt status registers.

FIFO interrupts can be enabled or disabled by setting the bit V_IRQ in register A_IRQ_MSK[FIFO]. Because there are 64 interrupts there are 8 interrupt status registers for FIFO interrupts. To determine which interrupt register must be read in an interrupt routine there is an interrupt overview register which shows in which status register at least one interrupt bit is set (R_IRQ_OVIEW). Reading this register does not clear any interrupt. The following reading of an interrupt register (R_IRQ_FIFO_BL0 … R_IRQ_FIFO_BL7) clears the reported interrupts.

There are some other conditions which also can generate an interrupt. These are reported in the register R_IRQ_MISC and can be masked in the register R_IRQMSK_MISC.

The R_IRQ_CTRL register sets the behavior of the interrupt output pin. V_GLOB_IRQ_EN enables the interrupt pin. V_FIFO_IRQ enables the mask enabled FIFO interrupts.

12.4 Watchdog and Timer

The HFC-E1 includes a watchdog and a timer with interrupt capability.

The timer counts F0IO pulses. So the timer is incremented every 125 μs. The watchdog counter is incremented every 2 ms.

The timer values for timer and watchdog can be selected by the R_TI_WD register. 16 different timer and watchdog values can be selected.

The watchdog can be manually reset by setting bit V_WD_RES of the R_BERT_WD_MD register. Furthermore the watchdog is reset at every access to the HFC-E1 if bit V_AUTO_WD_RES of the R_BERT_WD_MD register is set.

12.5 Register description

12.5.1 Write only register

R_IRQMSK_MISC	(write only)	0x11

Miscellaneous interrupt status mask register

'0' means that the interrupt is not used for generating an interrupt on the interrupt pin 197.

Bits	Reset Value	Name	Description
0	0	V_STA_IRQMSK	**State of state machine changed interrupt mask bit**
1	0	V_TI_IRQMSK	**Timer elapsed interrupt mask bit**
2	0	V_PROC_IRQMSK	**Processing / nonprocessing transition interrupt mask bit** (every 125 μs)
3	0	V_DTMF_IRQMSK	**DTMF detection interrupt mask bit**
4	0	V_IRQ1S_MSK	**1 second interrupt mask bit**
5	0	V_SA6_IRQMSK	**SA6 pattern changed or external interrupt mask bit**
6	0	V_RX_EOMF_MSK	**Receive end of multiframe mask bit**
7	0	V_TX_EOMF_MSK	**Transmit end of multiframe mask bit**

R_IRQ_CTRL	(write only)	0x13

Interrupt control register

Bits	Reset Value	Name	Description
0	0	**V_FIFO_IRQ**	**FIFO interrupt** '0' = FIFO interrupts disabled '1' = FIFO interrupts enabled
2..1		**(reserved)**	Must be '00'.
3	0	**V_GLOB_IRQ_EN**	**Global interrupt signal enable (pin 197)** '0' = disable '1' = enable
4	0	**V_IRQ_POL**	**Polarity of interrupt signal** '0' = low active signal '1' = high active signal
7..5		**(reserved)**	Must be '000'.

R_TI_WD	(write only)	0x1A

Timer and watchdog control register

Bits	Reset Value	Name	Description
3..0	0	V_EV_TS	**Timer event after** $2^n \cdot 250\ \mu s$ 0 = 250 μs 1 = 500 μs 2 = 1 ms 3 = 2 ms 4 = 4 ms 5 = 8 ms 6 = 16 ms 7 = 32 ms 8 = 64 ms 9 = 128 ms 0xA = 256 ms 0xB = 512 ms 0xC = 1.024 s 0xD = 2.048 s 0xE = 4.096 s 0xF = 8.192 s
7..4	0	V_WD_TS	**Watchdog event after** $2^n \cdot 2\ \mathrm{ms}$ 0 = 2 ms 1 = 4 ms 2 = 8 ms 3 = 16 ms 4 = 32 ms 5 = 64 ms 6 = 128 ms 7 = 256 ms 8 = 512 ms 9 = 1.024 s 0xA = 2.048 s 0xB = 4.096 s 0xC = 8.192 s 0xD = 16.384 s 0xE = 32.768 s 0xF = 65.536 s

A_IRQ_MSK [FIFO]	(write only)	0xFF

Interrupt register for the selected FIFO

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

Bits	Reset Value	Name	Description
0	0	**V_IRQ**	**Interrupt mask for the selected FIFO** '0' = disabled '1' = enabled
1	0	**V_BERT_EN**	**BERT output enable** '0' = BERT disabled, normal data is transmitted '1' = BERT enabled, output of BERT generator is transmitted
2	0	**V_MIX_IRQ**	**Mixed interrupt generation** '0' = disabled (normal operation) '1' = frame interrupts and transparent interrupts are both generated in HDLC mode
7..3		**(reserved)**	Must be '00000'.

12.5.2 Read only register

R_IRQ_OVIEW	(read only)	0x10

FIFO interrupt overview register

Every bit with value '1' indicates that an interrupt has occured in the FIFO block. A FIFO block consists of 4 transmit and 4 receive FIFOs. The exact FIFO can be determined by reading the R_IRQ_FIFO_BL0 ...R_IRQ_FIFO_BL7 registers that belong to the specified FIFO block.

Reading any R_IRQ_FIFO_BL0 ...R_IRQ_FIFO_BL7 registers clear the corresponding bit in this register. Reading this overview register does not clear any interrupt bit.

Bits	Reset Value	Name	Description
0		V_IRQ_FIFO_BL0	**Interrupt overview of FIFO block 0** (FIFOs 0 ... 3)
1		V_IRQ_FIFO_BL1	**Interrupt overview of FIFO block 1** (FIFOs 4 ... 7)
2		V_IRQ_FIFO_BL2	**Interrupt overview of FIFO block 2** (FIFOs 8 ... 11)
3		V_IRQ_FIFO_BL3	**Interrupt overview of FIFO block 3** (FIFOs 12 ... 15)
4		V_IRQ_FIFO_BL4	**Interrupt overview of FIFO block 4** (FIFOs 16 ... 19)
5		V_IRQ_FIFO_BL5	**Interrupt overview of FIFO block 5** (FIFOs 20 ... 23)
6		V_IRQ_FIFO_BL6	**Interrupt overview of FIFO block 6** (FIFOs 24 ... 27)
7		V_IRQ_FIFO_BL7	**Interrupt overview of FIFO block 7** (FIFOs 28 ... 31)

R_IRQ_MISC	(read only)	0x11

Miscellaneous interrupt status register

All bits of this register are cleared after a read access.

Bits	Reset Value	Name	Description
0	0	V_STA_IRQ	**State change** '1' = state of HFC-E1 interface state machine has changed
1	0	V_TI_IRQ	**Timer interrupt** '1' = timer elapsed
2	0	V_IRQ_PROC	**Processing / non processing transition interrupt status** '1' = The HFC-E1 has changed from processing to non processing phase (every 125 μs).
3	0	V_DTMF_IRQ	**DTMF detection interrupt** '1' = DTMF detection has been finished. The results can be read from the RAM.
4	0	V_IRQ1S	**1 second interrupt** '1' = 1 second elapsed
5	0	V_SA6_IRQ	**SA6 pattern has changed or external interrupt**
6	0	V_RX_EOMF	**End of multiframe received**
7	0	V_TX_EOMF	**End of multiframe transmited**

R_STATUS	(read only)	0x1C

HFC-E1 status register

Bits	Reset Value	Name	Description
0	0	V_BUSY	**BUSY / NOBUSY status** '1' = the HFC-E1 is BUSY after initialising Reset FIFO, increment F-counter or change FIFO '0' = the HFC-E1 is not busy, all accesses are allowed
1	1	V_PROC	**Processing / non processing status** '1' = the HFC-E1 is in processing phase (every 125 μs) '0' = the HFC-E1 is not in processing phase
2	0	V_DTMF_IRQSTA	**DTMF interrupt** DTMF interrupt has occured
3	0	V_LOST_STA	**LOST error (frames have been lost)** This means the HFC-E1 did not process all data in 125 μs. So data may be corrupted. Bit V_RES_LOST of the R_INC_RES_FIFO register must be set to reset this bit.
4	0	V_SYNC_IN	**Synchronization input** Value of the SYNC_I input pin
5	0	V_EXT_IRQSTA	**External interrupt** External interrupt has occured
6	0	V_MISC_IRQSTA	**Any miscellaneous interrupt** All enabled miscellaneous interrupts of the register R_IRQ_MISC are 'ored'.
7	0	V_FR_IRQSTA	**Any FIFO interrupt** All enabled FIFO interrupts in the registers R_IRQ_FIFO_BL0 ... R_IRQ_FIFO_BL7 are 'ored'.

R_IRQ_FIFO_BL0	(read only)	0xC8

FIFO interrupt register for FIFO block 0

In HDLC mode the *end of frame* is signaled, while in transparent mode the frequency of interrupts is set in the bitmap V_TRP_IRQ of the register A_CON_HDLC.

The bit value '1' indicates that the corresponding FIFO generated an interrupt. If a bit is '0', no interrupt occured in the corresponding FIFO.

Reading this register clears all set bits and the corresponding bit of the register R_IRQ_OVIEW.

Bits	Reset Value	Name	Description
0	0	V_IRQ_FIFO0_TX	Interrupt occured in transmit FIFO 0
1	0	V_IRQ_FIFO0_RX	Interrupt occured in receive FIFO 0
2	0	V_IRQ_FIFO1_TX	Interrupt occured in transmit FIFO 1
3	0	V_IRQ_FIFO1_RX	Interrupt occured in receive FIFO 1
4	0	V_IRQ_FIFO2_TX	Interrupt occured in transmit FIFO 2
5	0	V_IRQ_FIFO2_RX	Interrupt occured in receive FIFO 2
6	0	V_IRQ_FIFO3_TX	Interrupt occured in transmit FIFO 3
7	0	V_IRQ_FIFO3_RX	Interrupt occured in receive FIFO 3

R_IRQ_FIFO_BL1	(read only)	0xC9

FIFO interrupt register for FIFO block 1

In HDLC mode the *end of frame* is signaled, while in transparent mode the frequency of interrupts is set in the bitmap V_TRP_IRQ of the register A_CON_HDLC.

The bit value '1' indicates that the corresponding FIFO generated an interrupt. If a bit is '0', no interrupt occured in the corresponding FIFO.

Reading this register clears all set bits and the corresponding bit of the register R_IRQ_OVIEW.

Bits	Reset Value	Name	Description
0	0	V_IRQ_FIFO4_TX	Interrupt occured in transmit FIFO 4
1	0	V_IRQ_FIFO4_RX	Interrupt occured in receive FIFO 4
2	0	V_IRQ_FIFO5_TX	Interrupt occured in transmit FIFO 5
3	0	V_IRQ_FIFO5_RX	Interrupt occured in receive FIFO 5
4	0	V_IRQ_FIFO6_TX	Interrupt occured in transmit FIFO 6
5	0	V_IRQ_FIFO6_RX	Interrupt occured in receive FIFO 6
6	0	V_IRQ_FIFO7_TX	Interrupt occured in transmit FIFO 7
7	0	V_IRQ_FIFO7_RX	Interrupt occured in receive FIFO 7

R_IRQ_FIFO_BL2	(read only)	0xCA

FIFO interrupt register for FIFO block 2

In HDLC mode the *end of frame* is signaled, while in transparent mode the frequency of interrupts is set in the bitmap V_TRP_IRQ of the register A_CON_HDLC.

The bit value '1' indicates that the corresponding FIFO generated an interrupt. If a bit is '0', no interrupt occured in the corresponding FIFO.

Reading this register clears all set bits and the corresponding bit of the register R_IRQ_OVIEW.

Bits	Reset Value	Name	Description
0	0	V_IRQ_FIFO8_TX	Interrupt occured in transmit FIFO 8
1	0	V_IRQ_FIFO8_RX	Interrupt occured in receive FIFO 8
2	0	V_IRQ_FIFO9_TX	Interrupt occured in transmit FIFO 9
3	0	V_IRQ_FIFO9_RX	Interrupt occured in receive FIFO 9
4	0	V_IRQ_FIFO10_TX	Interrupt occured in transmit FIFO 10
5	0	V_IRQ_FIFO10_RX	Interrupt occured in receive FIFO 10
6	0	V_IRQ_FIFO11_TX	Interrupt occured in transmit FIFO 11
7	0	V_IRQ_FIFO11_RX	Interrupt occured in receive FIFO 11

R_IRQ_FIFO_BL3	(read only)	0xCB

FIFO interrupt register for FIFO block 3

In HDLC mode the *end of frame* is signaled, while in transparent mode the frequency of interrupts is set in the bitmap V_TRP_IRQ of the register A_CON_HDLC.

The bit value '1' indicates that the corresponding FIFO generated an interrupt. If a bit is '0', no interrupt occured in the corresponding FIFO.

Reading this register clears all set bits and the corresponding bit of the register R_IRQ_OVIEW.

Bits	Reset Value	Name	Description
0	0	V_IRQ_FIFO12_TX	Interrupt occured in transmit FIFO 12
1	0	V_IRQ_FIFO12_RX	Interrupt occured in receive FIFO 12
2	0	V_IRQ_FIFO13_TX	Interrupt occured in transmit FIFO 13
3	0	V_IRQ_FIFO13_RX	Interrupt occured in receive FIFO 13
4	0	V_IRQ_FIFO14_TX	Interrupt occured in transmit FIFO 14
5	0	V_IRQ_FIFO14_RX	Interrupt occured in receive FIFO 14
6	0	V_IRQ_FIFO15_TX	Interrupt occured in transmit FIFO 15
7	0	V_IRQ_FIFO15_RX	Interrupt occured in receive FIFO 15

R_IRQ_FIFO_BL4	(read only)	0xCC

FIFO interrupt register for FIFO block 4

In HDLC mode the *end of frame* is signaled, while in transparent mode the frequency of interrupts is set in the bitmap V_TRP_IRQ of the register A_CON_HDLC.

The bit value '1' indicates that the corresponding FIFO generated an interrupt. If a bit is '0', no interrupt occured in the corresponding FIFO.

Reading this register clears all set bits and the corresponding bit of the register R_IRQ_OVIEW.

Bits	Reset Value	Name	Description
0	0	V_IRQ_FIFO16_TX	Interrupt occured in transmit FIFO 16
1	0	V_IRQ_FIFO16_RX	Interrupt occured in receive FIFO 16
2	0	V_IRQ_FIFO17_TX	Interrupt occured in transmit FIFO 17
3	0	V_IRQ_FIFO17_RX	Interrupt occured in receive FIFO 17
4	0	V_IRQ_FIFO18_TX	Interrupt occured in transmit FIFO 18
5	0	V_IRQ_FIFO18_RX	Interrupt occured in receive FIFO 18
6	0	V_IRQ_FIFO19_TX	Interrupt occured in transmit FIFO 19
7	0	V_IRQ_FIFO19_RX	Interrupt occured in receive FIFO 19

R_IRQ_FIFO_BL5	(read only)	0xCD

FIFO interrupt register for FIFO block 5

In HDLC mode the *end of frame* is signaled, while in transparent mode the frequency of interrupts is set in the bitmap V_TRP_IRQ of the register A_CON_HDLC.

The bit value '1' indicates that the corresponding FIFO generated an interrupt. If a bit is '0', no interrupt occured in the corresponding FIFO.

Reading this register clears all set bits and the corresponding bit of the register R_IRQ_OVIEW.

Bits	Reset Value	Name	Description
0	0	V_IRQ_FIFO20_TX	Interrupt occured in transmit FIFO 20
1	0	V_IRQ_FIFO20_RX	Interrupt occured in receive FIFO 20
2	0	V_IRQ_FIFO21_TX	Interrupt occured in transmit FIFO 21
3	0	V_IRQ_FIFO21_RX	Interrupt occured in receive FIFO 21
4	0	V_IRQ_FIFO22_TX	Interrupt occured in transmit FIFO 22
5	0	V_IRQ_FIFO22_RX	Interrupt occured in receive FIFO 22
6	0	V_IRQ_FIFO23_TX	Interrupt occured in transmit FIFO 23
7	0	V_IRQ_FIFO23_RX	Interrupt occured in receive FIFO 23

R_IRQ_FIFO_BL6	(read only)	0xCE

FIFO interrupt register for FIFO block 6

In HDLC mode the *end of frame* is signaled, while in transparent mode the frequency of interrupts is set in the bitmap V_TRP_IRQ of the register A_CON_HDLC.

The bit value '1' indicates that the corresponding FIFO generated an interrupt. If a bit is '0', no interrupt occured in the corresponding FIFO.

Reading this register clears all set bits and the corresponding bit of the register R_IRQ_OVIEW.

Bits	Reset Value	Name	Description
0	0	V_IRQ_FIFO24_TX	Interrupt occured in transmit FIFO 24
1	0	V_IRQ_FIFO24_RX	Interrupt occured in receive FIFO 24
2	0	V_IRQ_FIFO25_TX	Interrupt occured in transmit FIFO 25
3	0	V_IRQ_FIFO25_RX	Interrupt occured in receive FIFO 25
4	0	V_IRQ_FIFO26_TX	Interrupt occured in transmit FIFO 26
5	0	V_IRQ_FIFO26_RX	Interrupt occured in receive FIFO 26
6	0	V_IRQ_FIFO27_TX	Interrupt occured in transmit FIFO 27
7	0	V_IRQ_FIFO27_RX	Interrupt occured in receive FIFO 27

R_IRQ_FIFO_BL7	(read only)	0xCF

FIFO interrupt register for FIFO block 7

In HDLC mode the *end of frame* is signaled, while in transparent mode the frequency of interrupts is set in the bitmap V_TRP_IRQ of the register A_CON_HDLC.

The bit value '1' indicates that the corresponding FIFO generated an interrupt. If a bit is '0', no interrupt occured in the corresponding FIFO.

Reading this register clears all set bits and the corresponding bit of the register R_IRQ_OVIEW.

Bits	Reset Value	Name	Description
0	0	V_IRQ_FIFO28_TX	Interrupt occured in transmit FIFO 28
1	0	V_IRQ_FIFO28_RX	Interrupt occured in receive FIFO 28
2	0	V_IRQ_FIFO29_TX	Interrupt occured in transmit FIFO 29
3	0	V_IRQ_FIFO29_RX	Interrupt occured in receive FIFO 29
4	0	V_IRQ_FIFO30_TX	Interrupt occured in transmit FIFO 30
5	0	V_IRQ_FIFO30_RX	Interrupt occured in receive FIFO 30
6	0	V_IRQ_FIFO31_TX	Interrupt occured in transmit FIFO 31
7	0	V_IRQ_FIFO31_RX	Interrupt occured in receive FIFO 31

Chapter 13

General purpose I/O pins (GPIO) and input pins (GPI)

(For an overview of the GPIO and GPI pins see Table 13.2 on page 253.)

Table 13.1: *Overview of the HFC-E1 general purpose I/O registers*

Write only registers:			Read only registers:		
Address	**Name**	**Page**	**Address**	**Name**	**Page**
0x40	R_GPIO_OUT0	254	0x40	R_GPIO_IN0	259
0x41	R_GPIO_OUT1	255	0x41	R_GPIO_IN1	260
0x42	R_GPIO_EN0	256	0x44	R_GPI_IN0	261
0x43	R_GPIO_EN1	257	0x45	R_GPI_IN1	262
0x44	R_GPIO_SEL	258	0x46	R_GPI_IN2	263
			0x47	R_GPI_IN3	264

13.1 GPIO and GPI functionality

Most of the interface signals can be used as general purpose I/O pins (GPIOs) or those who are only inputs as general purpose input pins (GPIs). This functionality can be used if the pins are not used as dedicated E1 interface.

GPIOs must be switched to GPIO mode in the register R_GPIO_SEL if they should be used as outputs. The input functionality of all GPIOs and GPIs is allways enabled. The output values for the GPIOs are set in the registers R_GPIO_OUT0 and R_GPIO_OUT1. The tristate function can be enabled in the registers R_GPIO_EN0 and R_GPIO_EN1.

The input values for the GPIO[0..15] can be read in the registers R_GPIO_IN0 and R_GPIO_IN1. The input values for GPI[0..31] can be read in the registers R_GPI_IN0, R_GPI_IN1, R_GPI_IN2 and R_GPI_IN3.

13.2 GPIO output voltage adjustment

The GPIO output high voltage can be influenced for each set of 4 GPIOs by connecting the appropiate VDD_E1 pin to a voltage different from VDD. The voltage must not exceed 3.6 V. See Table 13.2 for details.

Table 13.2: *Adjustable pin groups of the HFC-E1*

Power supply pin		Adjustable amplitude pins		Power supply pin		Adjustable amplitude pins	
129	VDD_E1	124	GPI31	164	VDD_E1	159	GPI15
		125	GPI30			160	GPI14
		126	GPI29			161	GPI13
		127	GPI28			162	GPI12
		130	GPIO15			165	GPIO7
		131	GPIO14			166	GPIO6
		132	GPIO13			167	GPIO5
		133	GPIO12			168	GPIO4
		136	GPI27			171	GPI11
		137	GPI26			172	GPI10
		138	GPI25			173	GPI9
		139	GPI24			174	GPI8
147	VDD_E1	142	GPI23	181	VDD_E1	176	GPI7
		143	GPI22			177	GPI6
		144	GPI21			178	GPI5
		145	GPI20			179	GPI4
		148	GPIO11			182	GPIO3
		149	GPIO10			183	GPIO2
		150	GPIO9			184	GPIO1
		151	GPIO8			185	GPIO0
		154	GPI19			188	GPI3
		155	GPI18			189	GPI2
		156	GPI17			190	GPI1
		157	GPI16			191	GPI0

13.3 Register description

> **Please note !**
>
> For using a port as GPIO the R_GPIO_SEL register must be programmed.

13.3.1 Write only register

R_GPIO_OUT0	(write only)	0x40
GPIO data output bits 7 . . . 0		

Bits	Reset Value	Name	Description
0	0	V_GPIO_OUT0	Output data for pin GPIO0
1	0	V_GPIO_OUT1	Output data for pin GPIO1
2	0	V_GPIO_OUT2	Output data for pin GPIO2
3	0	V_GPIO_OUT3	Output data for pin GPIO3
4	0	V_GPIO_OUT4	Output data for pin GPIO4
5	0	V_GPIO_OUT5	Output data for pin GPIO5
6	0	V_GPIO_OUT6	Output data for pin GPIO6
7	0	V_GPIO_OUT7	Output data for pin GPIO7

R_GPIO_OUT1	(write only)	0x41

GPIO data output bits 15 … 8

Bits	Reset Value	Name	Description
0	0	V_GPIO_OUT8	Output data for pin GPIO8
1	0	V_GPIO_OUT9	Output data for pin GPIO9
2	0	V_GPIO_OUT10	Output data for pin GPIO10
3	0	V_GPIO_OUT11	Output data for pin GPIO11
4	0	V_GPIO_OUT12	Output data for pin GPIO12
5	0	V_GPIO_OUT13	Output data for pin GPIO13
6	0	V_GPIO_OUT14	Output data for pin GPIO14
7	0	V_GPIO_OUT15	Output data for pin GPIO15

R_GPIO_EN0	(write only)	0x42

GPIO data output enable bits 7 . . . 0

Bits	Reset Value	Name	Description
0	0	V_GPIO_EN0	Output enable for pin GPIO0
1	0	V_GPIO_EN1	Output enable for pin GPIO1
2	0	V_GPIO_EN2	Output enable for pin GPIO2
3	0	V_GPIO_EN3	Output enable for pin GPIO3
4	0	V_GPIO_EN4	Output enable for pin GPIO4
5	0	V_GPIO_EN5	Output enable for pin GPIO5
6	0	V_GPIO_EN6	Output enable for pin GPIO6
7	0	V_GPIO_EN7	Output enable for pin GPIO7

R_GPIO_EN1	(write only)	0x43

GPIO data output enable bits 15 … 8

Bits	Reset Value	Name	Description
0	0	V_GPIO_EN8	Output enable for pin GPIO8
1	0	V_GPIO_EN9	Output enable for pin GPIO9
2	0	V_GPIO_EN10	Output enable for pin GPIO10
3	0	V_GPIO_EN11	Output enable for pin GPIO11
4	0	V_GPIO_EN12	Output enable for pin GPIO12
5	0	V_GPIO_EN13	Output enable for pin GPIO13
6	0	V_GPIO_EN14	Output enable for pin GPIO14
7	0	V_GPIO_EN15	Output enable for pin GPIO15

R_GPIO_SEL	(write only)	0x44

GPIO selection register

This register allows to select first or second function of some pins.

Bits	Reset Value	Name	Description
0	0	V_GPIO_SEL0	**GPIO0 and GPIO1** '0' = pins T_A and T_B enabled '1' = pins GPIO0 and GPIO1 enabled
1	0	V_GPIO_SEL1	**GPIO2 and GPIO3** '0' = pins GPIO2 and GPIO3 disabled '1' = pins GPIO2 and GPIO3 enabled
2	0	V_GPIO_SEL2	**GPIO4 and GPIO5** '0' = pins GPIO4 and GPIO5 disabled '1' = pins GPIO4 and GPIO5 enabled
3	0	V_GPIO_SEL3	**GPIO6 and GPIO7** '0' = pins GPIO6 and GPIO7 disabled '1' = pins GPIO6 and GPIO7 enabled
4	0	V_GPIO_SEL4	**GPIO8 and GPIO9** '0' = pins GPIO8 and GPIO9 disabled '1' = pins GPIO8 and GPIO9 enabled
5	0	V_GPIO_SEL5	**GPIO10 and GPIO11** '0' = pins GPIO10 and GPIO11 disabled '1' = pins GPIO10 and GPIO11 enabled
6	0	V_GPIO_SEL6	**GPIO12 and GPIO13** '0' = pins GPIO12 and GPIO13 disabled '1' = pins GPIO12 and GPIO13 enabled
7	0	V_GPIO_SEL7	**GPIO14 and GPIO15** '0' = pins GPIO14 and GPIO15 disabled '1' = pins GPIO14 and GPIO15 enabled

13.3.2 Read only register

R_GPIO_IN0	(read only)	0x40
GPIO data input bits 7 … 0		

Bits	Reset Value	Name	Description
0	0	V_GPIO_IN0	Input data from pin GPIO0
1	0	V_GPIO_IN1	Input data from pin GPIO1
2	0	V_GPIO_IN2	Input data from pin GPIO2
3	0	V_GPIO_IN3	Input data from pin GPIO3
4	0	V_GPIO_IN4	Input data from pin GPIO4
5	0	V_GPIO_IN5	Input data from pin GPIO5
6	0	V_GPIO_IN6	Input data from pin GPIO6
7	0	V_GPIO_IN7	Input data from pin GPIO7

R_GPIO_IN1	(read only)	0x41

GPIO data input bits 15 … 8

Bits	Reset Value	Name	Description
0	0	V_GPIO_IN8	Input data from pin GPIO8
1	0	V_GPIO_IN9	Input data from pin GPIO9
2	0	V_GPIO_IN10	Input data from pin GPIO10
3	0	V_GPIO_IN11	Input data from pin GPIO11
4	0	V_GPIO_IN12	Input data from pin GPIO12
5	0	V_GPIO_IN13	Input data from pin GPIO13
6	0	V_GPIO_IN14	Input data from pin GPIO14
7	0	V_GPIO_IN15	Input data from pin GPIO15

R_GPI_IN0	(read only)	0x44

GPI data input bits 7 . . . 0

Note: Unused GPI pins must be connected to ground.

Bits	Reset Value	Name	Description
0	0	V_GPI_IN0	Input data from pin GPI0
1	0	V_GPI_IN1	Input data from pin GPI1
2	0	V_GPI_IN2	Input data from pin GPI2
3	0	V_GPI_IN3	Input data from pin GPI3
4	0	V_GPI_IN4	Input data from pin GPI4
5	0	V_GPI_IN5	Input data from pin GPI5
6	0	V_GPI_IN6	Input data from pin GPI6
7	0	V_GPI_IN7	Input data from pin GPI7

R_GPI_IN1	(read only)	0x45

GPI data input bits 15 . . . 8

Note: Unused GPI pins must be connected to ground.

Bits	Reset Value	Name	Description
0	0	V_GPI_IN8	Input data from pin GPI8
1	0	V_GPI_IN9	Input data from pin GPI9
2	0	V_GPI_IN10	Input data from pin GPI10
3	0	V_GPI_IN11	Input data from pin GPI11
4	0	V_GPI_IN12	Input data from pin GPI12
5	0	V_GPI_IN13	Input data from pin GPI13
6	0	V_GPI_IN14	Input data from pin GPI14
7	0	V_GPI_IN15	Input data from pin GPI15

R_GPI_IN2	(read only)	0x46

GPI data input bits 23 . . . 16

Note: Unused GPI pins must be connected to ground.

Bits	Reset Value	Name	Description
0	0	V_GPI_IN16	Input data from pin GPI16
1	0	V_GPI_IN17	Input data from pin GPI17
2	0	V_GPI_IN18	Input data from pin GPI18
3	0	V_GPI_IN19	Input data from pin GPI19
4	0	V_GPI_IN20	Input data from pin GPI20
5	0	V_GPI_IN21	Input data from pin GPI21
6	0	V_GPI_IN22	Input data from pin GPI22
7	0	V_GPI_IN23	Input data from pin GPI23

R_GPI_IN3	(read only)	0x47

GPI data input bits 31 … 24

Note: Unused GPI pins must be connected to ground.

Bits	Reset Value	Name	Description
0	0	V_GPI_IN24	Input data from pin GPI24
1	0	V_GPI_IN25	Input data from pin GPI25
2	0	V_GPI_IN26	Input data from pin GPI26
3	0	V_GPI_IN27	Input data from pin GPI27
4	0	V_GPI_IN28	Input data from pin GPI28
5	0	V_GPI_IN29	Input data from pin GPI29
6	0	V_GPI_IN30	Input data from pin GPI30
7	0	V_GPI_IN31	Input data from pin GPI31

Chapter 14

Electrical characteristics

Absolute maximum ratings

Parameter	Symbol	Min.	Max.
Power supply	V_{DD}	$-0.3\,\mathrm{V}$	$+4.6\,\mathrm{V}$
Input voltage	V_I	$-0.3\,\mathrm{V}$	$5.5\,\mathrm{V}$
Operating temperature	T_{opr}	$0\,^\circ\mathrm{C}$	$+70\,^\circ\mathrm{C}$
Junction temperature	T_{jnc}	$0\,^\circ\mathrm{C}$	$+100\,^\circ\mathrm{C}$
Storage temperature	T_{stg}	$-55\,^\circ\mathrm{C}$	$+125\,^\circ\mathrm{C}$

Recommended operating conditions

Parameter	Symbol	Min.	Typ.	Max	Conditions
Power supply	V_{DD}	$3.0\,\mathrm{V}$	$3.3\,\mathrm{V}$	$3.6\,\mathrm{V}$	
Operating temperature	T_{opr}	$0\,^\circ\mathrm{C}$		$+70\,^\circ\mathrm{C}$	

Electrical characteristics for 3.3 V power supply

Parameter	Symbol	Min.	Typ.	Max	Conditions
Low input voltage	V_{IL}	$-0.3\,\mathrm{V}$		$0.2V_{DD}$	
High input voltage	V_{IH}	$0.7V_{DD}$		V_{DD}	
Low output voltage	V_{OL}	$0\,\mathrm{V}$		$0.4\,\mathrm{V}$	
High output voltage	V_{OH}	$2.4\,\mathrm{V}$		V_{DD}	

Appendix A

HFC-E1 package dimensions

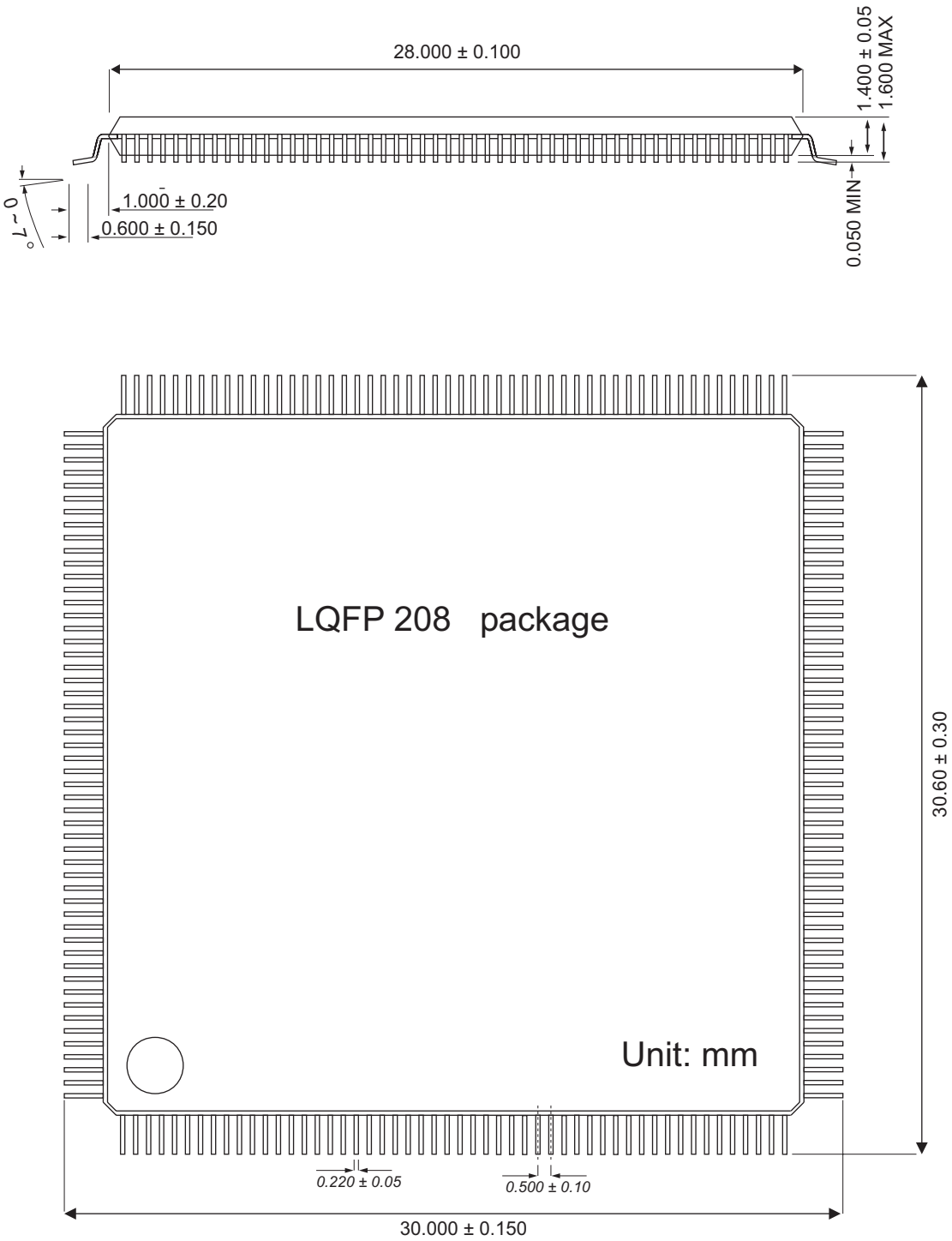

Figure A.1: *HFC-E1 package dimensions*

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.

A	A bit
ADDR	address
ADDR0	address (byte 0)
ADDR1	address (byte 1)
ADDR2	address (byte 2)
ADJ	adjust
AIS	alarm indication signal
ALT	alternate
ATT	attenuation
ATX	analog transmitter
AUTO	automatic
BERT	bit error rate test
BIT	bit
BL	block
BRG	bridge
BUSY	busy
C4	C4IO clock
CFG	configuration
CH	HFC-channel
CHANNEL	HFC-channel
CHG	changed
CHIP	chip
CLK	clock
CMI	code mark inversion
CNT	counter
CNTH	counter, high byte
CNTL	counter, low byte
CODE	code
CON	connection settings
COND	condition
CONF	conference
CRC	cyclic redundancy check
CS	chip select
CSM	channel select mode
CTRL	control
DATA	data
DEC	decoder
DEL	deletion
DIR	direction
DR	data rate
DTMF	dual tone multiple frequency
E	CRC error indication bits
E1	E1 bit
E1RES	E1 interface reset
E2	E2 bit
ECH	error counter, high byte
ECL	error counter, low byte
EN	enable
END	end
EOMF	end of multiframe
EPR	EEPROM
ERR	error
EV	event
EXCHG	exchange
EXT	external
F	F-counter
F0	frame syncronization signal
F1	F1-counter
F12	F1- and F2-counter
F2	F2-counter
FAS	frame alignment signal
FBAUD	full bauded
FG	F/G state
FIFO	FIFO
FIRST	first
FLOW	flow
FOSLIP	force frequency slip warning
FOSTA	force state
FR	frame
FSM	FIFO sequence mode
GLOB	global
GPI	general purpose input
GPIO	general purpose input/output
HARM	harmonic
HCLK	half clock (frequency)
HDLC	high-level data link control
HFCRES	HFC reset

ICR	increase
ID	identifier
IDLE	idle
IDX	index
IFF	inter frame fill
IN	input
INC	increment
INIT	buffer initialization
INSYNC	synchronization to input data
INT	internal
INV	invert
IPATS	IPATS test
IRQ	interrupt
IRQ1S	one-second interrupt
IRQMSK	interrupt mask
IRQSTA	interrupt status
ITU	ITU-T G.775
JATT	jitter attenuator
LEN	length
LEV	level
LI	line
LOOP	loop
LOS	loss of signal
LOSS	loss of synchronization signal
LOST	frame data lost
LPRIO	low priority
MD	mode
MF	multiframe
MFA	multiframe alignment
MISC	miscellaneous
MIX	mixed
MSK	mask
NEG	negative
NEXT	next
NFAS	no frame alignment signal
NMF	no multiframe
NO	no
NOINC	no increment
NOISE	noise
NTRI	no tristate
NUM	number
OFF	off
OFFS	offset
OFLOW	overflow
OK	ok
ON	on
OUT	output
OVIEW	overview
PAT	pattern
PCM	PCM
PCMRES	PCM reset
PLL	phase locked loop
PNP	plug and play
POL	polarity
PROC	processing
PWM	pulse width modulation
RAL	remote alarm
RAM	RAM
RD	read
RECO	recovery
RES	reset
RESTART	restart
RESYNC	resynchronization
REV	reverse
RLD	reload
ROUT	routing (of PCM buffer)
RV	revision
RX	receive
SA	spare bits
SA13	spare bit S_{a13}
SA23	spare bit S_{a23}
SA6	spare bit S_{a6}
SA84	spare bits $S_a[8:4]$
SCI	state change interrupt
SEL	select
SEQ	sequence
SH	shape
SH0H	shape 0, high byte
SH0L	shape 0, low byte
SH1H	shape 1, high byte
SH1L	shape 1, low byte
SI	SI bit
SIG	signal
SIM	simulation
SIZE	buffer size
SL	time slot
SL0	time slot 0
SLIP	frequency slip
SLOT	PCM time slot
SLOW	slow
SPEED	speed
SRAM	SRAM
SRC	source
SRES	soft reset
STA	state, status
START	start
STATE	state, status
STATUS	status
STOP	stop
SUBCH	subchannel
SUPPR	suppression (threshold)
SWORD	service word
SYNC	synchronize
SZ	size
TI	timer
TIM	timing
TIME	time
TRANS	transition
TRP	transparent
TS	timestep
TX	transmit
ULAW	μ-law
use	usage
VIO	code violation
WD	watchdog timer
WR	write
XCRC	extended CRC4
XS13	Spare bit of frame 13

XS15	Spare bit of frame 15	**Z1H**	Z1-counter, high byte	**Z2H**	Z2-counter, high byte
Z1	Z1-counter	**Z1L**	Z1-counter, low byte	**Z2L**	Z2-counter, low byte
Z12	Z1- and Z2-counter	**Z2**	Z2-counter		

Cologne Chip AG

Data Sheet of HFC-E1