PMC-FIFO

2 * 16-bit FIFO (4k, 16k or Compatible)
8-bit TTL Input with IRQ
8-bit TTL Input Static
8-bit TTL Output

Hardware Manual

N O T E

The information in this document has been carefully checked and is believed to be entirely reliable. **esd** makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. **esd** reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

esd assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of **esd** gmbh.

esd does not convey to the purchaser of the product described herein any license under the patent rights of **esd** gmbh nor the rights of others.

esd electronic system design gmbh
Vahrenwalder Str. 205
D-30165 Hannover
Germany

Tel: +49-511-372-980
Fax: +49-511-37298-68
E-mail: info@esd-electronics.com
Internet: http://www.esd-electronics.com

Document file:	I:\TEXTE\DOKU\MANUALS\SONSTIGE\PMC_FIFO\PMFIF6HE.MA6
Date of print:	1.12.99

PCB version:	PMCFIFO-10

Changes in the chapters

The changes in the user's manual listed below affect changes in the hardware as well as changes in the description of the facts only.

Chapter	Changes versus previous version
1.2,3	designation of the I/O-Connector P1 corrected
-	

Technical details are subject to change without notice.

Content

Page

1. Overview

1.1 Module Description

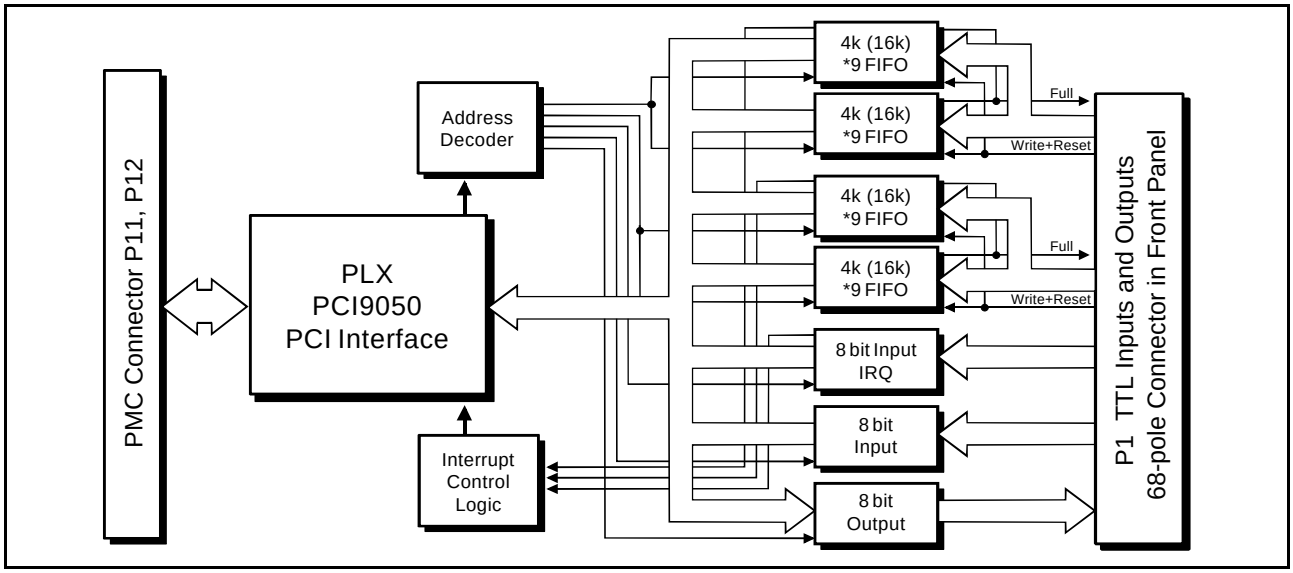


Fig. 1.1.1: Block diagram of PMC-FIFO

The module PMC-FIFO is supported with 2x 16 FIFO inputs for TTL level signals. The FIFOs can be equipped as 4k-size, 16k-size or compatible circuits. Reading-in the FIFO inputs is controlled by two externally available write inputs. By two RESET inputs the FIFOs can be deleted externally. For a flexible read-out the FIFOs can generate interrupts at 'Empty', ‘Half-Full’, and ‘Full’.

Furthermore the module has another 16 inputs and 8 outputs, which can be read or set by TTL-registers or TTL-drivers. 8 of the inputs can trigger an interrupt on the module.

After a power-on or when receiving the RESET signal, all FIFOs and interrupts are deleted and the 8 digital outputs are set to ‘high’.

The TTL-inputs and outputs are connected by a 68-pole male contact in the front panel.

1.2 General Technical Data

Module Interface	*PMC*
Temperature range	0...70EC ambient temperature
Humidity	max. 90%, non-condensing
Connectors	P11 - *PMC* 64-pole P12 - *PMC* 64-pole P1 - 68-pole, Typ: Thomas&Betts: 311-068072E (TTL-I/O signals via front panel) with Thomas&Betts: 311-068302 as necessary match
Board dimensions	74 mm x 149 mm
Weight	ca. 100 g
Power supply	via *PMC* : 5V / ca. 500mA

Table 1.2.1: General data of the PMC-FIFO

1.3 Technical Data of the I/O-Circuits

Number of FIFO inputs	2 * 16 (4k or 16k each) 2 * FIFO Write, 2 * FIFO Reset
Timing of the FIFO inputs (refer page 7)	data setup time: t_S $ 20 ns data hold time: t_H $ 20 ns data cycle time: t_P $ 50 ns burst read rate: # 132 Mbyte/s non burst read rate: # 4 Mbyte/s
Number of FIFO outputs	2 * FIFO-Full-Flag
Number of digital inputs with interrupt capability	8
Number of digital inputs	8
Number of digital outputs	8
Voltage level (all I/Os)	TTL-level
Receiver (IRQ-inputs, Write signals of FIFOs and FIFO reset)	74LS14 (Schmitt-trigger with hysteresis)
Receiver (all other inputs)	74LS244
Transmitter (FIFO-Full-Flag)	74LS244
Transmitter (8-bit output)	74ALS573
Interrupts	- FIFO IRQ at ‘empty’, ‘half full’ and ‘full’ - 8 digital inputs with interrupt (edge ‘high to low’)

Table 1.3.1: Technical data of the I/O circuits

1.4 Order Information

Type	Features	Order No.
PMC-FIFO	PMC-module with 2x 16-bit FIFO, 8-bit input (with IRQ), 8-bit input, 8-bit output	V.2010.01
PMC-FIFO-OS9	Driver as C-source for OS9	V.2010.50
PMC-FIFO-VxWorks	Driver as C-source for VxWorks	V.2010.56
PMC-FIFO-RTOS-UH	Driver for RTOS-UH	V.2010.54
PMC-FIFO-MD (*)	English user's manual	V.2010.20

1*) If ordered together with the module, the manual is free of charge.

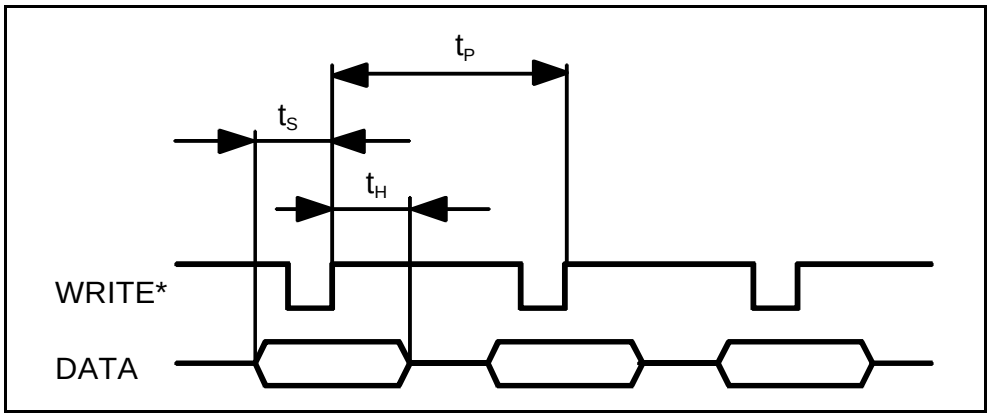
Table 1.4.1: Order information

2. FIFO Memory

The used FIFO memory components are of type Cypress CY7C433 (4k depth) or CY7C462 (16k depth) or compatible components.

The following figure displays the FIFO-write timing for reading data. Data is sampled at rising edge of the WRITE signal. Data has to be valid at least 20 ns before the rising edge appears (setup time t_S) and has to be hold at least 20 ns after the rising edge of the WRITE signal (hold time t_H)

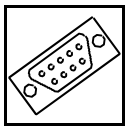
The minimum cycle time for reading data is 50 ns (t_P) resulting a write frequency of maximum 20 Mhz.



t_S $ 20 ns
t_H $ 20 ns
t_P $ 50 ns

Fig. 2.1: FIFO Write Timing

Reading of the FIFOs is done with a data rate of up to 132 Mbyte/s in burst mode (DMA controller). Without burst mode the maximum data rate is 4 Mbit/s.



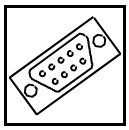
3. Appendix

3.1 Connector Pin Assignments

3.1.1 Assignment of the 64-pole PMC-Connector P11

Pin #	Signal Name	Signal Name	Pin #
1	TCK	-12V	2
3	GND	INTA*	4
5	INTB*	INTC*	6
7	PMCPRSNT*	+5V	8
9	INTD*	-	10
11	GND	-	12
13	CLK	GND	14
15	GND	PMCGNT*	16
17	PMCREQ*	+5V	18
19	+5V	AD31	20
21	AD28	AD27	22
23	AD25	GND	24
25	GND	C/BE3*	26
27	AD22	AD21	28
29	AD19	+5V	30
31	+5V	AD17	32
33	FRAME*	GND	34
35	GND	IRDY*	36
37	DEVSEL*	+5V	38
39	GND	LOCK*	40
41	SDONE*	SBO*	42
43	PAR	GND	44
45	+5V	AD15	46
47	AD12	AD11	48
49	AD09	+5V	50
51	GND	C/BE0*	52
53	AD06	AD05	54
55	AD04	GND	56
57	+5V	AD01	58
59	AD02	AD01	60
61	AD00	+5V	62
63	GND	REQ64*	64

PMC SPECIFICATION IEEE1386.1/Draft 2.0 - 04-APR-1995

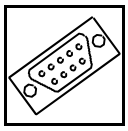


3.1.2 Assignment of the 64-pole PMC-Connector P12

Pin #	Signal Name	Signal Name	Pin #
1	+12V	TRST*	2
3	TMS	TDO	4
5	TDI	GND	6
7	GND	-	8
9	-	-	10
11	-	-	12
13	RST*	-	14
15	-	-	16
17	-	GND	18
19	AD30	AD29	20
21	GND	AD26	22
23	AD24	-	24
25	IDSEL	AD23	26
27	-	AD20	28
29	AD18	GND	30
31	AD16	C/BE2*	32
33	GND	-	34
35	TRDY*	-	36
37	GND	STOP*	38
39	PERR*	GND	40
41	-	SERR*	42
43	C/BE1*	GND	44
45	AD14	AD13	46
47	GND	AD10	48
49	AD08	-	50
51	AD07	-	52
53	-	-	54
55	-	GND	56
57	-	-	58
59	GND	-	60
61	ACK64*	-	62
63	GND	-	64

PMC SPECIFICATION IEEE1386.1/Draft 2.0 - 04-APR-1995

- This pin is not connected at the board.



3.1.3 Assignment of the 68-pole I/O-Connector P1

Pin #	Signal Name	Signal Name	Pin #
1	F1D0	F2D12	35
2	F1D1	F2D13	36
3	F1D2	F2D14	37
4	F1D3	F2D15	38
5	F1D4	F2WRITE*	39
6	F1D5	F2RESET*	40
7	F1D6	F2FULL*	41
8	F1D7	GND	42
9	GND	IRQ0	43
10	F1D8	IRQ1	44
11	F1D9	IRQ2	45
12	F1D10	IRQ3	46
13	F1D11	IRQ4	47
14	F1D12	IRQ5	48
15	F1D13	IRQ6	49
16	F1D14	IRQ7	50
17	F1D15	GND	51
18	F1WRITE*	IN0	52
19	F1RESET*	IN1	53
20	F1FULL*	IN2	54
21	GND	IN3	55
22	F2D0	IN4	56
23	F2D1	IN5	57
24	F2D2	IN6	58
25	F2D3	IN7	59
26	F2D4	GND	60
27	F2D5	OUT0	61
28	F2D6	OUT1	62
29	F2D7	OUT2	63
30	GND	OUT3	64
31	F2D8	OUT4	65
32	F2D9	OUT5	66
33	F2D10	OUT6	67
34	F2D11	OUT7	68

Type of I/O-Connector on the board (male contact): Thomas&Betts: 311-068072E
Type of necessary mating connector at the line (female contact): Thomas&Betts: 311-068302

PMC-FIFO

2 * 16bit FIFO (4k, 16k or compatible)
8bit TTL-Input with IRQ
8bit TTL-Input Static
8bit TTL-Output

Software Manual

N O T E

The information in this document has been carefully checked and is believed to be entirely reliable. **esd** makes no warranty of any kind with regard to the material in this document, and assumes no responsibility for any errors that may appear in this document. **esd** reserves the right to make changes without notice to this, or any of its products, to improve reliability, performance or design.

esd assumes no responsibility for the use of any circuitry other than circuitry which is part of a product of **esd** gmbh.

esd does not convey to the purchaser of the product described herein any license under the patent rights of **esd** gmbh nor the rights of others.

esd electronic system design gmbh
Vahrenwalder Str. 205
30165 Hannover
Germany

Phone: +49-511-372 98-0
Fax: +49-511-372 98-68
E-mail: info@esd-electronics.com
Internet: www.esd-electronics.com

USA / Canada
7667 W. Sample Road
Suite 127
Coral Springs, FL 33065
USA

Phone: +1-800-504-9856
Fax: +1-800-288-8235
E-mail: sales@esd-electronics.com

Software described	Revision/date
PMC-FIFO C-Interface	05/05/99

Changes in the software and/or documentation

Changes in this manual as compared with previous version	Changes in the software	Changes in the documentation
first english issue	-	-

Contents

Page

1. Programming Interface

The following chapter will describe the C-programming interface of the PMC-FIFO. The values returned in case of errors will be described in chapter 2.

1.1 Initialisation

The services described below are used to initialise the PMC-FIFO hardware.

---

pmcfifoOpen()

Name: **pmcfifoOpen()** - Initialising the PMC-FIFO board

Call:
```
int     pmcfifoOpen
        (
        )
```

Description: This function enables the memory area of the PMC-FIFO and initialises the PCI-components on the host CPU and the PMC-FIFO board. It has to be called before any other routine for communication with the PMC-FIFO.

Return: 0 or an error code described in the appendix.

---

pmcfifoClose()

Name: **pmcfifoClose()** - Terminating the communication with the PMC-FIFO board.

Call:
```
int     mcfifoClose
        (
        )
```

Description: This function switches off all connected interrupts and blocks all further access to the PMC-FIFO.

Return: 0 or an error code described in the appendix.

1.2 Reading the FIFOs

The services described below configure and communicate with the FIFO-components on the PMC-FIFO.

---

pmcfifoFIFOConfig()

Name: **pmcfifoFIFOConfig()** - Configuring the FIFO-memory.

Call:

```
int     pmcfifoFIFOConfig
        (
        int     i,          /* FIFO-number (1 or 2) */
        int     mode,       /* mode (0 ... 2) */
        int     depth,      /* FIFO depth (e.g. 4096) */
        void *  buf         /* pointer to free buffer */
        )
```

Description: This function configures the FIFO-buffer *i* as 8 (*mode*=0), 16 (*mode*=1) or 32 (*mode*=2) bit component. An interrupt service routine is connected which guarantees that the FIFOs are automatically read-out in status ‘half full’ or ‘full’. The parameter *depth* has got the depth of the FIFO-components used. This value has to be specified, because the components can be exchanged. In order to save the FIFO values a pointer to a buffer *buf* large enough is specified. The first longword in buffer *buf* contains the write pointer, which shows the number of values which have been read out of the FIFO.

In 8-bit mode the 8 MSB are ignored and only the 8 LSB are saved.

In 32-bit mode the two FIFO-buffers (each 16 bit) are combined to a 32-bit component and saved as a 32-bit value. FIFO number 1 has got the 16 LSB and FIFO number two has got the 16 MSB. 1 has to be specified for FIFO number *i*. In 32-bit mode it is not permissible to configure FIFO number 2. In this configuration the FIFO control signals (WRITE*, RESET*) of the first FIFO are used as control signals for both FIFOs.

The FIFO-buffer is deleted additionally.

Return: 0 or an error code described in the appendix.

pmcfifoFIFOClear()

Name: **pmcfifoFIFOClear()** - Deleting a FIFO-buffer.

Call:

```
int     pmcfifoFIFOClear
        (
        int     i               /* FIFO-number (1 or 2) */
        )
```

Description: This function deletes one of the two FIFO-memory components of the PMC-FIFO.

Return: 0 or an error code described in the appendix.

---

pmcfifoFIFOReset()

Name: **pmcfifoFIFOReset()** - Resetting a FIFO-buffer.

Call:

```
int     pmcfifoFIFOReset
        (
        int     i,              /* FIFO-number (1 or 2) */
        int     mode            /* mode (0 ... 2) */
        )
```

Description: This function deletes one of the two FIFO-memory components of the PMC-FIFO and resets the FIFO again with the mode specified (see **pmcfifoFIFO-Config**).

Return: 0 or an error code described in the appendix.

---

pmcfifoFIFORead()

Name: **pmcfifoFIFORead()** - Complete read-out of a FIFO-buffer.

Call:

```
int     pmcfifoFIFORead
        (
        int     i,              /* FIFO-number (1 or 2) */
        int     signal,         /* signal code */
        int     proc_id         /* process ID */
        )
```

Description: This function reads out the FIFO-buffer *i* completely (until the FIFO-component signals ‘empty’). Then the signal *signal* is transmitted to the process *proc_id* by calling *_os_send().*

Return: 0 or an error code described in the appendix.

pmcfifoFIFOStatus()

Name: **pmcfifoFIFOStatus()** - Reading the status of the FIFO-buffer.

Call:

```
char    pmcfifoFIFOStatus
        (
        )
```

Description: This function reads the status of the two FIFO-buffers and shows their value as return parameter.

Return: 8 bit with the following meaning:

Bit 0:	FIFO 1 empty	Bit 4:	FIFO 2 empty
Bit 1:	FIFO 1 half full	Bit 5:	FIFO 2 half full
Bit 2:	FIFO 1 full	Bit 6:	FIFO 2 full
Bit 3:	reserved	Bit 7:	reserved

1.3 Interrupt Handling

The services described below are used to configure the inputs which are suitable for interrupts.

---

pmcfifoIRQSignal()

Name: **pmcfifoIRQSignal()** - Configuring the inputs suitable for interrupts (OS-9).

Call:

```
int     pmcfifoIRQSignal
        (
        int     mode,           /* mode (0 or 1) */
        int     bit,            /* bit number (0 ... 7) */
        int     signal,         /* signal code */
        int     proc_id         /* process ID */
        )
```

Description: This function links the interrupt input *bit* (0...7) with *signal*.

If 1 is specified as *mode*, a *_os_send(proc_id, signal)* is once executed, provided the corresponding interrupt is triggered. This means that a process waiting for this signal can be ‘pushed’ once via an interrupt.

If 2 is specified as *mode*, the link is not established once but permanently. This link, however, can be cancelled again by calling *mode* 0.

If 0 is specified as *mode*, the link between *bit* and *signal* is cancelled again. No process is executed by the specified interrupt *bit*.

Return: 0 or an error code described in the appendix.

---

pmcfifoIRQWait()

Name: **pmcfifoIRQWait()** - Waiting for an interrupt (VxWorks / RTOS-UH).

Call:

```
int     pmcfifoIRQWait
        (
        int     bit             /* bit number (0 ... 7) */
        )
```

Description: After this function has been called the calling process waits until an interrupt is generated at the specified interrupt input *bit* (0...7).

Return: 0 or an error code described in the appendix.

pmcfifoIRQRead()

Name: **pmcfifoIRQRead()** - Reading the 8 interrupt inputs.

Call:

```
int     pmcfifoIRQRead
        (
        )
```

Description: This function reads the eight IRQ-inputs and gives their interrupt status as return parameters. A 1 corresponds to a previous ‘high-to-low’ level change.

Return: The 8 bits combined in a least significant byte in integer. Bit 0 corresponds to input number 0, etc.

1.4 Reading and Writing Data

The services described below are used to read the TTL-inputs or to write the TTL-outputs.

---

pmcfifoTTLRead()

Name: **pmcfifoTTLRead()** - Reading the 8 TTL-inputs.

Call:

```
int     pmcfifoTTLRead
        (
        )
```

Description: This function reads the 8 TTL-inputs and gives their value as return parameters. A 1 corresponds to a ‘high’ and a 0 to a ‘low’at the input.

Return: The 8 bits combined in a least significant byte in integer. Bit 0 corresponds to input number 0, etc.

---

pmcfifoTTLWrite()

Name: **pmcfifoTTLWrite()** - Writing one of the 8 TTL-outputs.

Call:

```
int     pmcfifoTTLWrite
        (
        int     bit_no,         /* bit number (0 ... 7) */
        int     bit_count,      /* bit counter (1 ... 8) */
        int     value           /* value (0 or 1) */
        )
```

Description: This function configures one or more of the 8 TTL-outputs (from bit number *bit_no bit_count*-bits) to the specified value *value*. A 1 corresponds to a ‘high’ and a 0 to a ‘low’ at the output. The other outputs remain unchanged.

Return: 0 or an error code described in the appendix.

2. Returned Values

A function with a returned value of type *int* is returned an error code according to the table below. Not every code is meaningful for every function.

Error	Description