



DIN EN ISO 9001:2000
certified



ADDI-DATA GmbH
Dieselstraße 3
D-77833 OTTERSWEIER



Technical support:
+49 (0)7223 / 9493 – 0

Function description

ADDICOUNT APCI-/CPCI-1710

Edge Time Measurement (ETM)

Edition: 02.01-08/2005

Product information

This manual contains the technical installation and important instructions for correct commissioning and usage, as well as production information according to the current status before printing. The content of this manual and the technical product data may be changed without prior notice. ADDI-DATA GmbH reserves the right to make changes to the technical data and the materials included herein.

Warranty and liability

The user is not permitted to make changes to the product beyond the intended use, or to interfere with the product in any other way.

ADDI-DATA shall not be liable for obvious printing and phrasing errors. In addition, ADDI DATA, if legally permissible, shall not be liable for personal injury or damage to materials caused by improper installation and/or commissioning of the board by the user or improper use, for example, if the board is operated despite faulty safety and protection devices, or if notes in the operating instructions regarding transport, storage, installation, commissioning, operation, thresholds, etc. are not taken into consideration. Liability is further excluded if the operator changes the board or the source code files without authorisation and/or if the operator is guilty of not monitoring the permanent operational capability of working parts and this has led to damage.

Copyright

This manual, which is intended for the operator and its staff only, is protected by copyright. Duplication of the information contained in the operating instructions and of any other product information, or disclosure of this information for use by third parties, is not permitted, unless this right has been granted by the product licence issued. Non-compliance with this could lead to civil and criminal proceedings.

ADDI-DATA software product licence

Please read this licence carefully before using the standard software. The customer is only granted the right to use this software if he/she agrees with the conditions of this licence.

The software must only be used to set up the ADDI-DATA boards.

Reproduction of the software is forbidden (except for back-up and for exchange of faulty data carriers). Disassembly, decompilation, decryption and reverse engineering of the software are forbidden. This licence and the software may be transferred to a third party if this party has acquired a board by purchase, has agreed to all the conditions in this licence contract and the original owner does not keep any copies of the software.

Trademarks

- ADDI-DATA is a registered trademark of ADDI-DATA GmbH.
- Turbo Pascal, Delphi, Borland C, Borland C++ are registered trademarks of Borland Insight Company.
- Microsoft C, Visual C++, Windows XP, 98, Windows 2000, Windows 95, Windows NT, EmbeddedNT and MS DOS are registered trademarks of Microsoft Corporation.
- LabVIEW, LabWindows/CVI, DasyLab, Diadem are registered trademarks of National Instruments Corp.
- CompactPCI is a registered trademark of PCI Industrial Computer Manufacturers Group.
- VxWorks is a registered trademark of Wind River Systems Inc.

WARNING

The following risks result from improper implementation and from use of the board contrary to the regulations:



- ◆ Personal injury
- ◆ Damage to the board, PC and peripherals
- ◆ Pollution of the environment

◆ **Protect yourself, the others and the environment!**

◆ **Read carefully the safety precautions (yellow leaflet).**

If this leaflet is not with the documentation, please contact us and ask for it.

◆ **Observe the instructions of the manual.**

Make sure that you do not forget or skip any step. We are not liable for damages resulting from a wrong use of the board.

◆ **Used symbols:**



IMPORTANT!

designates hints and other useful information.



WARNING!

It designates a possibly dangerous situation.

If the instructions are ignored the board, PC and/or peripheral may be destroyed.

1	DEFINITION OF APPLICATION	6
1.1	Intended use	6
1.2	Usage restrictions.....	6
1.3	Technical description	6
1.4	Function description	7
1.5	Used abbreviations	7
2	ETM.....	8
2.1	General description	8
2.1.1	Block diagram of the ETM function	8
2.1.2	Typical applications	9
2.2	Used signals	9
2.3	Pin assignment of the ETM.....	9
2.4	Connection example	10
2.5	I/O mapping	11
2.6	Description of the I/O functions.....	11
2.6.1	Division factor Register (Base +0)	11
2.6.2	Module register (command/status; Base + 4)	12
2.6.3	Command register ETM counter 0 (Base + 8).....	12
2.6.4	State register ETM counter 0 (Base + 20).....	12
2.6.5	Command register ETM counter 0 (Base+ 24)	12
2.6.6	Version register (Base + 60).....	13
2.7	Working with the ETM function.....	13
3	STANDARD SOFTWARE	14
3.1	Introduction	14
3.2	Interrupt mask	14
3.3	ETM initialisation.....	16
	1) i_APCI1710_InitETM (...)	16
	2) i_APCI1710_EnableETM (...).....	18
	3) i_APCI1710_DisableETM (...).....	20
	4) i_APCI1710_GetETMInitialisation (...).....	21
3.4	Read ETM	23
	1) i_APCI1710_GetETMProgressStatus (...)	23
	2) i_APCI1710_ReadETMValue (...)	25
	3) i_APCI1710_ReadETMTotalTime	27
	4) i_APCI1710_ConvertETMValue (...)	29
3.5	Functions in the kernel mode.....	31
	1) i_APCI1710_KRNL_ReadETMValue (...)	31

Figures

Fig. 2-1: Block diagram of the ETM function	8
Fig. 2-2: Pin assignment of the front connector	10
Fig. 2-3: Connection example	10

Tables

Table 1-1: Delivered manuals	7
Table 2-1: Used signals.....	9
Table 2-2: I/O mapping of the ETM function.....	11
Table 3-1: Define value	14
Table 3-2: Interrupt mask of the function "ETM"	14
Table 3-3: Return table for the counter value.....	15
Table 3-4: Value of the time base.....	17

1 DEFINITION OF APPLICATION

1.1 Intended use

The board **APCI-1710** must be inserted in a PC with PCI 5V/32-bit slots, which is used as electrical equipment for measurement, control and laboratory pursuant to the norm IEC 61010-1.

The board **CPCI-1710** must be inserted in a CompactPCI system with PCI 5V/32-bit slots, which is used as electrical equipment for measurement, control and laboratory pursuant to the norm IEC 61010-1

1.2 Usage restrictions

The board **APCI-/CPCI-1710** must not be used as safety related part for securing emergency stop functions

The board **APCI-/CPCI-1710** must not be used in potentially explosive atmospheres.

1.3 Technical description

This manual refers to the **APCI-1710** as well as to the **CPCI-1710** board. Make sure that you have received the following items:

- The CD 1 "Standard Software Drivers" with the ADDISET parameterizing program and the required software drivers.
- The CD 2 "Technical Manuals". This CD contains the following:

- 1) The technical description **ADDICOUNT APCI-1710 / CPCI-1710: Function-programmable counter board for the PCI bus** (containing general information on the operation of the board)
- 2) A function description for each function which you want to program on the board
- 3) The yellow leaflet "Safety precautions"

According to the function used you will find the required assignment and programming functions in the different manuals for each function:

Table 1-1: Delivered manuals

Function	PDF file (CD2 technical manuals)		Function description in SET1710	CFG file
	German	English		
Incremental counter	Inkr_zähler_d.pdf	Incr_counter_e.pdf	Incremental counter	inc_cpt.cfg
SSI	SSI_d.pdf	ssi_e.pdf	SSI	ssi.cfg
SSI monitor	SSI-Monitor_d	SSIMonitor_e.pdf	SSI_Monitor	ssi_mon.cfg
Chronos	chronos_d.pdf	chronos_e.pdf	Chronos	chronos.cfg
Counter/timer	Zähler_timer_d.pdf	Counter_timer_e.pdf	counter/timer	82x54.cfg
TOR	TOR_d.pdf	TOR_e.pdf	TOR	tor.cfg
PWM	PWM_d.pdf	PWM_e.pdf	Pulse width modulation	PWM.cfg
TTL	TTL_IO_d.pdf	TTL_IO_e.pdf	TTL I/O	ttl_io.cfg
Digital I/O	dig_EA_d.pdf	dig_IO_e.pdf	Digital I/O	dig_IO.cfg
Pulse counter	Impulszähler_d.pdf	pulseCounter_e.pdf	Pulse counter	imp_cpt.cfg
ETM (Edge time measurement)	ETM_d.pdf	ETM_e.pdf	Edge time measurement	etm.cfg

Please note:

The board CPCI-1710 is compatible with the board APCI-1710 as far as the installation of the software is concerned. The ADDIREG and SET1710 programs make no difference between PCI and CompactPCI boards. The API functions of the standard software are also identical.

1.4 Function description

Apart from a global description of the functions this manual contains:

- the pin assignment of the front connector
- a list of the used signals
- the I/O mapping
- a chapter about the API software functions of the standard software.

1.5 Used abbreviations

The signals on the 50 pin SUB-D connector refer always to one function module.

Please note the used abbreviations:

- UAS: Interference signal
- CLK: Clock
- REF: Reference point logic
- ENA: Enable

C1+ is a signal for **function module 1**.

2 ETM

2.1 General description

The function "ETM" is a timer interface which allows to measure the time of a period and at the same time the high or low level time of this period.

The following 2 functions are implemented:

- 1 x 32-bit timer, in order to create a reference time
- 2 x 32-bit measurement timers, which measure the period time and the time of the high or low level.

Properties:

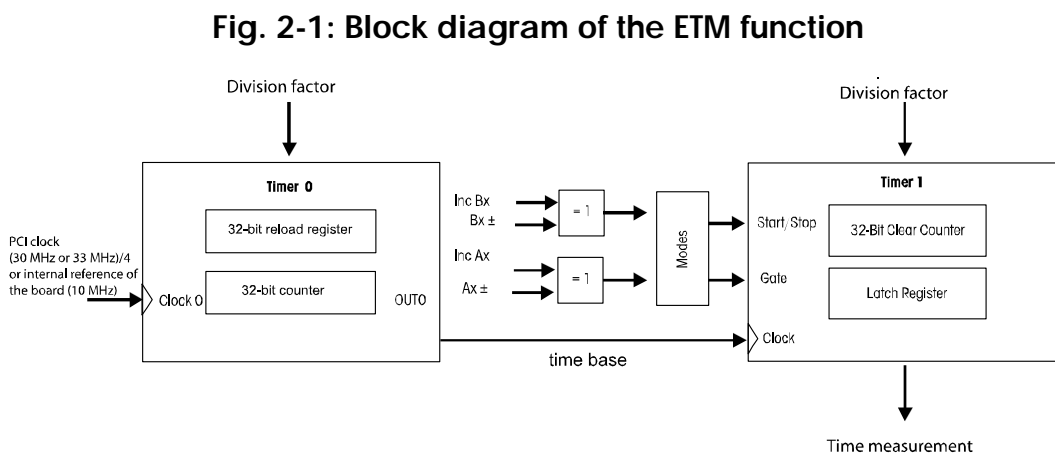
Complete isolation through optical couplers for the input and output channels to avoid earth circuit

- Interrupt possibility at the end of measurement
- Signals up to 5 MHz can be processed
- Timer is rereadable
- Inputs and outputs can be inverted through software
- Software GATE possible.

2.1.1 Block diagram of the ETM function

The interface contains:

- 1 gate input
- 2 independent from each other 32-bit timer, which can be read or written via the data bus.



2.1.2 Typical applications

- Period time measurement
- Level time measurement

2.2 Used signals

The function ETM occupies **4 inputs** (A to D) of the respecting function module of the **APCI-/CPCI-1710**.

On one board you can use max. 8 ETM (2 per module).

Table 2-1: Used signals

AM STECKER	POLARITÄT	FUNKTION
Ax +/-	Diff./TTL/optional 24 V	Gate input of the ETM counter 0
Bx +/-	Diff./TTL/optional 24 V	Input of the ETM counter 0
Cx +/-	Diff./TTL/optional 24 V	Gate of the ETM counter 1
Dx +/-	Diff./TTL/optional 24 V	Input of the ETM counter 1

2.3 Pin assignment of the ETM



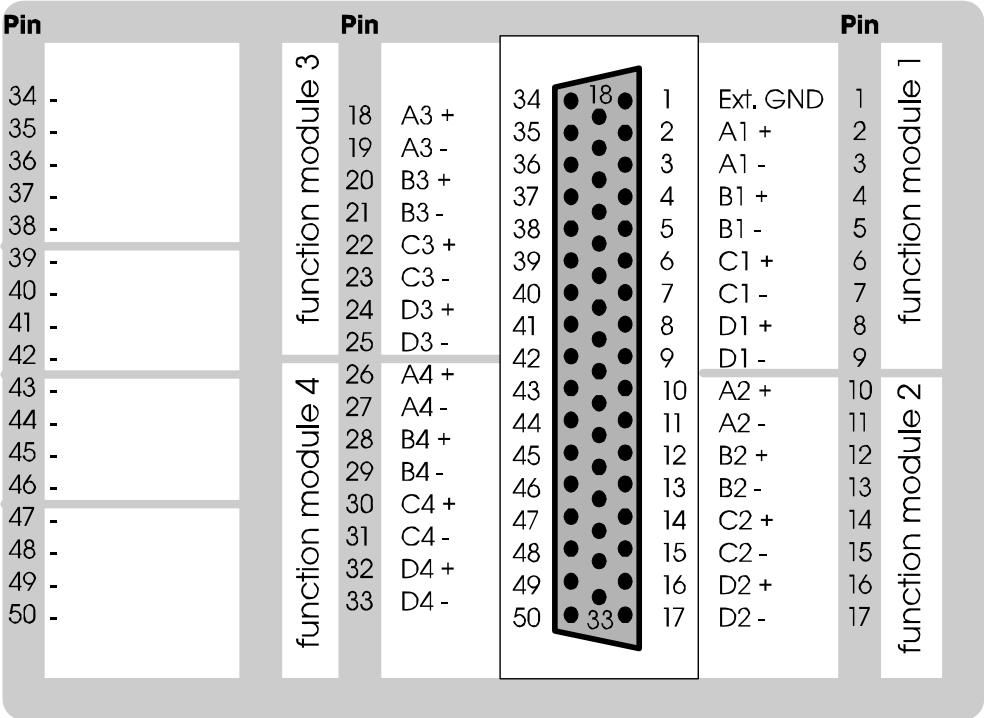
IMPORTANT!

The function modules are defined differently in the hardware and software descriptions.

For the pin assignment (hardware) the modules from 1 to 4 are numbered. For the SET1710 program or the software functions (Software) the module numbering **BEGINS** with 0.

The figure below is a connection example. The function “ETM” is implemented on all function modules.

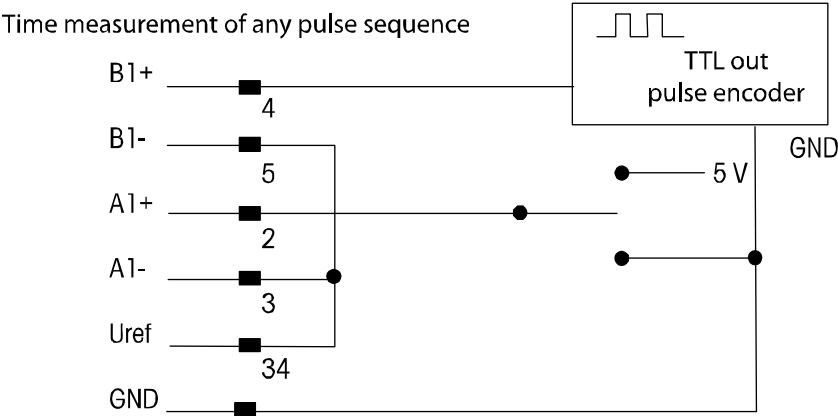
Fig. 2-2: Pin assignment of the front connector



:- not connected

2.4 Connection example

Fig. 2-3: Connection example



2.5 I/O mapping

Table 2-2: I/O mapping of the ETM function

			D31...D24	D23...D16	D15.....D8	D7.....D0
BYTES	Rd	Wr				
BASE x + 0	☑	☑		Division factor		
BASE x + 4	☑	☑				Command/status module x
BASE x + 8	☑	☑				Command ETM counter 0
BASE x + 12	☑			Edge time evaluation ETM counter 0		
BASE x + 16	☑			Total time counter 0		
BASE x + 20	☑					Status ETM counter 0
BASE x + 24	☑	☑				Command ETM counter1
BASE x + 28	☑			Edge time evaluation ETM counter1		
BASE x + 32	☑			Total time counter 1		
...			-	-	-	-
BASE x + 60	☑		FUNKNBR2	FUNKNBR1	REVBYTE2	REVBYTE1

-: No function

x: Number of the function module

The accesses are always written or read in 32-bit depth.

2.6 Description of the I/O functions

The ETM function contains 2 time gauges. Each time gauge has 2 signals:

- The gate signal (A or C channel)
- The signal to be measured (B or D channel)

At the trigger signal the time gauge is latched, reset to 0 and a new continuous measurement is started.

The time base of the time gauge can be defined through the software. It is the same for both counters. This time is between 25 ns and 0.4 s. As time base the 40 MHz quartz on the board is used.

2.6.1 Division factor Register (Base + 0)

Division factor for the time base

2.6.2 Module register (command/status; Base + 4)

D0:	0	PCI bus clock is used as time base.
	1	Internal quartz is used as time base (40 MHz).
D1:	0	Division factor for the time base is not yet initialised.
	1	Division factor for the time base is initialised.
D2:	0	No interrupt available on the first ETM counter
	1	Interrupt available on the first ETM counter
D3:	0	No interrupt available on the second ETM counter
	1	Interrupt available on the second ETM counter
D4:	0	Do not generate a software reset
	1	Generates a software reset on all 3 ETM counters

2.6.3 Command register ETM counter 0 (Base + 8)

D0:	0	Measures the low time
	1	Measures the high time
D1:	0	The low level starts/ends the measurement
	1	The high level starts/ends the measurement
D2:	0	Only one measurement is started
	1	Cyclic measurement
D3:	0	The measurement is started right after the start function
	1	The measurement is started after the next trigger signal
D4:	0	There is no interrupt after the measurement
	1	An interrupt is generated after each measurement
D5:	0	Do not start the measurement
	1	Start the measurement

2.6.4 State register ETM counter 0 (Base + 20)

D0:	0	Measurement is not yet started
	1	Measurement is started (trigger signal available)
D1:	0	Measurement is not yet ended
	1	Measurement is ended (trigger signal available)
D2:	0	No transition
	1	Transition

2.6.5 Command register ETM counter 0 (Base + 24)

D0:	0	Measures the low time
	1	Measures the high time
D1:	0	The low level starts/ends the measurement
	1	The high level starts/ends the measurement
D2:	0	Only one measurement will be started
	1	Cyclic measurement
D3:	0	The measurement is started right after the start function
	1	The measurement is started after the next trigger signal
D4:	0	There is no interrupt after the measurement
	1	An interrupt is generated after each measurement
D5:	0	Does not start the measurement
	1	Starts the measurement

2.6.6 Version register (Base +60)

Contains the function description and the revision (reading command, ASCII format)

BASE + 60 "E" "T" "1" "0"

Meaning: ETM, Revision 1.0

2.7 Working with the ETM function

1. Connection of the signal encoder to the board
2. Parametrization of the API function (signal level selection, time reference, single or continuous mode)
3. Evaluate the status of the measurement via polling or interrupt
4. Read out time measurement timer
5. Value from the time measurement timer and time reference determines the level and period time

3 STANDARD SOFTWARE

3.1 Introduction



IMPORTANT!

Note the following style conventions in the text:

Function: *"i_APCI1710_SetBoardInformation"*

Variable *ui_Address*

Table 3-1: Define value

Define name	Decimal value	Hexadecimal value
DLL_COMPILER_C	1	1
DLL_COMPILER_VB	2	2
DLL_COMPILER_PASCAL	3	3
DLL_LABVIEW	4	4
APCI1710_30MHZ	30	1E
APCI1710_33MHZ	33	21
APCI1710_40MHZ	40	28

3.2 Interrupt mask

Each ETM counter can generate an interrupt. In order to get this interrupt, you shall enable the interrupt and the interrupt routine with the function *"i_APCI1710_SetBoardIntRoutineX"* .

Table 3-2: Interrupt mask of the function "ETM"

b_ModuleMask	ul_InterruptMask	Meaning
0000 0001	00010 0000 0000 0000 0000	Interrupt on ETM counter 0, module 0
0000 0001	00100 0000 0000 0000 0000	Interrupt on ETM counter 1, module 0
0000 0010	00010 0000 0000 0000 0000	Interrupt on ETM counter 0, module 1
0000 0010	00100 0000 0000 0000 0000	Interrupt on ETM counter 1, module 1
0000 0100	00010 0000 0000 0000 0000	Interrupt on ETM counter 0, module 2
0000 0100	00100 0000 0000 0000 0000	Interrupt on ETM counter 1, module 2
0000 1000	00010 0000 0000 0000 0000	Interrupt on ETM counter 0, module 3
0000 1000	00100 0000 0000 0000 0000	Interrupt on ETM counter 1, module 3

Table 3-3: Return table for the counter value

b_ModuleMask	ul_InterruptMask	Source	ul_CounterLatchValue
b_ModuleMask = 1	ul_InterruptMask = 20000Hex	Interrupt generated through ETM counter 0 module 0	Measured edge time (24-bit)
b_ModuleMask = 1	ul_InterruptMask = 40000Hex	Interrupt generated through ETM counter 1 module 0	Measured edge time (24-bit)
b_ModuleMask = 2	ul_InterruptMask = 20000Hex	Interrupt generated through ETM counter 0 module 1	Measured edge time (24-bit)
b_ModuleMask = 2	ul_InterruptMask = 40000Hex	Interrupt generated through ETM counter 1 module 1	Measured edge time (24-bit)
b_ModuleMask = 4	ul_InterruptMask = 20000Hex	Interrupt generated through ETM counter 0 module 2	Measured edge time (24-bit)
b_ModuleMask = 4	ul_InterruptMask = 40000Hex	Interrupt generated through ETM counter 1 module 2	Measured edge time (24-bit)
b_ModuleMask = 8	ul_InterruptMask = 20000Hex	Interrupt generated through ETM counter 0 module 3	Measured edge time (24-bit)
b_ModuleMask = 8	ul_InterruptMask = 40000Hex	Interrupt generated through ETM counter 1 module 3	Measured edge time (24-bit)

3.3 ETM initialisation

1) i_APCI1710_InitETM (...)

Syntax:

```
<Return Wert> = i_APCI1710_InitETM
                    (BYTE      b_BoardHandle,
                     BYTE      b_ModulNbr,
                     BYTE      b_ClockSelection,
                     BYTE      b_TimingUnit,
                     ULONG     ul_Timing,
                     PULONG    pul_RealTiming)
```

Parameter:

-Input:

BYTE	b_BoardHandle	Handle of the board APCI-/CPCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_ClockSelection	Selection of the PCI bus clock - APCI1710_30MHZ: The board uses a PCI bus clock of 30 MHz - APCI1710_33MHZ: The board uses a PCI bus clock of 33 MHz - APCI1710_40 MHz: The board uses 40 MHz quartz clock.
BYTE	b_TimingUnit	Unit of the time base (0 to 2) 0: ns 1: µs 2: ms
ULONG	ul_Timing	Value of the time base See table "Value of the time base"

-Output:

PULONG	pul_RealTiming	Correct value of the time base. Returns the value that corresponds mostly with the value entered in the ul_Timing
--------	----------------	---

Table 3-4: Value of the time base

PCI bus clock	<i>b_TimingUnit</i>	<i>ul_Timing</i> Min. value	<i>ul_Timing</i> Max. value
APCI1710_30MHz	ns (0)	33	559240500
	μs (1)	1	559240
	ms (2)	1	559
APCI1710_33MHz	ns (0)	30	508400454
	μs (1)	1	508400
	ms (2)	1	508
APCI1710_40MHz	ns (0)	25	419430375
	μs (1)	1	419430
	ms (2)	1	419

Task:

Configures all ETM counters of the selected module (*b_ModulNbr*). The parameters *ul_Timing* und *ul_TimingUnit* determine the time base for the measurement. *pul_RealTiming* returns the correct time value.

Call this function before you call another function, which access the ETM counter.

Calling convention:

ANSI C:

```
int          i_ReturnValue;
unsigned char b_BoardHandle;
unsigned long ul_RealTiming;
```

```
i_ReturnValue = i_APCI1710_InitETM    (b_BoardHandle,
                                         0,
                                         APCI1710_40MHZ,
                                         1,
                                         100,
                                         &ul_RealTiming);
```

Return value:

0: No error

-1: The handle parameter of the board is wrong

-2: The selected module number is wrong.

-3: The selected module is no "ETM" module.

-4: The selected input clock is wrong

-5: The selected time unit is wrong.

-6: The selected time base is wrong

-7: The 40 MHz cannot be configured on your board.

2) i_APCI1710_EnableETM (...)

Syntax:

<Return Wert> = i_APCI1710_EnableETM

(BYTE	b_BoardHandle,
BYTE	b_ModulNbr,
BYTE	b_ETM,
BYTE	b_EdgeLevel,
BYTE	b_TriggerLevel,
BYTE	b_CycleMode,
BYTE	b_FirstTriggerMode,
BYTE	b_InterruptEnable)

Parameter:

-Input

BYTE	b_BoardHandle	Handle of the board APCI-/CPCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_ETM	Selection of the ETM counter (0 or 1)
BYTE	b_EdgeLevel	Edge selection for time measurement 0: Measures the low level time 1: Measures the high level time
BYTE	b_TriggerLevel	Selection of the trigger level 0: Trigger at low level 1: Trigger at high level
BYTE	b_CycleMode	Mode selection 0: Single mode 1: Continuous Mode. Each trigger stops the measurement and starts a new cycle.
BYTE	b_FirstTriggerMode	Mode of the first trigger 0: The measurement of the first edge time starts after calling the function "i_APCI1710_EnableETM". 1: The measurement of the first edge time starts after the next trigger signal.
BYTE	b_InterruptEnable	Enables or disables the interrupt function. An interrupt is generated after disabling APCI1710_DISABLE: Interrupt.

-Output:

There is no output

Task:

Enables the ETM counter of the selected module (*b_ModulNbr*). The function "i_APCI1710_InitETM" has to be called first. If the interrupt is enabled, the ETM counter generates an interrupt after each trigger signal. See function "i_APCI1710_SetBoardIntRoutineXX" and Table 3-4: Value of the time base

Calling convention:

ANSI C:

```
int          i_ReturnValue;  
unsigned char b_BoardHandle;  
  
i_ReturnValue = i_APCI1710_EnableETM  
                (b_BoardHandle,  
                 0,  
                 0,  
                 0,  
                 0,  
                 APCI1710_DISABLE);
```

Return value:

0: No error

-1: Handle parameter of the board is wrong

-2: The selected module number is wrong.

-3: The selected module is no "ETM" module.

-4: Selected ETM counter is wrong

-5: ETM is not initialised. See function "i_APCI1710_InitETM"

-6: Selection of the edge level is wrong

-7: Selection of the trigger level is wrong.

-8: Mode selection is wrong.

-9: Mode selection for the first trigger is wrong

-10: Interrupt parameter is wrong

-11: Interrupt function is not initialised.

See function "i_APCI1710_SetBoardIntRoutineXX".

3) i_APCI1710_DisableETM (...)**Syntax:**

```
<Return Wert> = i_APCI1710_DisableETM
                                (BYTE      b_BoardHandle,
                                BYTE      b_ModulNbr
                                BYTE      b_ETM)
```

Parameter:**-Input:**

BYTE	b_BoardHandle	Handle of the APCI-/CPCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_ETM	Selection of the ETM counter (0 or 1)

-Output:

There is no output.

Task:

Disables the ETM counter of the selected module (*b_ModulNbr*).

Calling convention:ANSI C:

```
int          i_ReturnValue;
unsigned char b_BoardHandle;
```

```
i_ReturnValue = i_APCI1710_DisableETM
                                (b_BoardHandle,
                                0,
                                0);
```

Return value:

0: No error

-1: Handle parameter of the board is wrong

-2: The selected module number is wrong.

-3: The selected module is no "ETM" module.

-4: ETM not initialised. See function "i_APCI1710_InitETM".

4) i_APCI1710_GetETMInitialisation (...)

Syntax:

<Return Wert> = i_APCI1710_GetETMInitialisation

(BYTE	b_BoardHandle,
BYTE	b_ModulNbr,
BYTE	b_ETM,
PBYTE	pb_TimingUnit,
PULONG	pul_Timing,
PBYTE	pb_EdgeLevel,
PBYTE	pb_TriggerLevel,
PBYTE	pb_CycleMode,
PBYTE	pb_FirstTriggerMode,
PBYTE	pb_InterruptEnable,
PBYTE	pb_Enable)

Parameter:

-Input:

BYTE	b_BoardHandle	Handle of the APCI-/CPCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_ETM	Selection of the ETM counter (0 or 1)

-Output:

PBYTE	pb_TimingUnit	Unit of the time base (0 to 2) 0: ns 1: µs 2: ms
PULONG	pul_Timing	Value of the time base. See table "Value of the time base"
PBYTE	pb_EdgeLevel	Edge selection for time measurement 0: Measures the low level time 1: Measures the high level time
PBYTE	pb_TriggerLevel	Selection of the trigger level 0: Trigger at the low level 1: Trigger at the high level
PBYTE	pb_CycleMode	Mode selection 0: Single mode 1: Continuous mode. Each trigger stops the measurement and starts a new cycle
PBYTE	pb_FirstTriggerMode	Mode of the first trigger 0: The measurement of the first edge time starts after calling the function "i_APCI1710_EnableETM" 1: The measurement of the first edge time starts after the next trigger signal.
PBYTE	pb_InterruptEnable	Enables or disables the interrupt function. APCI1710_ENABLE: Interrupt enables. An

PBYTE pb_Enable	interrupt occurs after the trigger APCI1710_DISABLE: Interrupt disabled. Returns whether the ETM is enabled or disabled. 0: ETM disabled 1: ETM enabled
-----------------	---

Task:

Returns the information about the ETM (*b_ETM*) initialisation of the selected module (*b_ModulNbr*).

First call the function "i_APCI1710_InitETM" before calling another function

Calling convention:

ANSI C:

```

unsigned char  b_TimingUnit;
unsigned long  ul_Timing;
unsigned char  b_EdgeLevel;
unsigned char  b_TriggerMode;
unsigned char  b_InterruptEnable;
unsigned char  b_Enable;

```

```

i_ReturnValue = i_APCI1710_GetETMInitialisation
                (b_BoardHandle,
                 0,
                 0,
                 &b_TimingUnit,
                 &ul_Timing,
                 &b_EdgeLevel,
                 &b_TriggerMode,
                 &b_InterruptEnable,
                 &b_Enable);

```

Return value

0: No error

- 1: Handle parameter of the board is wrong
- 2: The selected module number is wrong
- 3: The selected module is no "ETM" module.
- 4: Selection of the ETM counter is wrong.
- 5: ETM not initialised. See function "i_APCI1710_InitETM".

3.4 Read ETM

1) i_APCI1710_GetETMProgressStatus (...)

Syntax:

```
<Return Wert> = i_APCI1710_GetETMProgressStatus
                                     (BYTE      b_BoardHandle,
                                     BYTE      b_ModulNbr,
                                     BYTE      b_ETM
                                     PBYTE     pb_ETMStatus)
```

Parameter:

-Input:

BYTE	b_BoardHandle	Handle of the APCI-/CPCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_ETM	Selection of the ETM counter (0 or 1)

-Output:

PULONG	pb_ETMStatus	Return of the ETM state. 0: Measurement not started. No start trigger arrived. 1: Measurement started. A start trigger has arrived. 2: Measurement stopped. A stop trigger has arrived. The measurement is stopped. 3: Transition has been realised. Please change the time base with the function "i_APCI1710_InitETM".
--------	--------------	--

Task:

Returns the ETM state (*pb_ETMStatus*) of the selected module (*b_ModulNbr*).

Calling convention:

ANSI C:

```
int          i_ReturnValue;
unsigned char b_BoardHandle;
unsigned char b_ETMStatus;
```

```
i_ReturnValue = i_APCI1710_GetETMProgressStatus
               (b_BoardHandle,
               0,
               0,
               &pb_ETMStatus);
```

Return value

0: No error
-1: Handle parameter of the board is wrong
-2: The selected module number is wrong.
-3: The selected module is no "ETM" module.

- 4: Selected ETM counter is wrong
- 5: ETM not initialised. See function "i_APCI1710_InitETM".

2) i_APCI1710_ReadETMValue (...)

Syntax:

<Return Wert> = i_APCI1710_ReadETMValue
 (BYTE b_BoardHandle,
 BYTE b_ModulNbr,
 BYTE b_ETM
 UINT ui_TimeOut,
 PBYTE pb_ETMStatus,
 PULONG pul_ETMValue)

Parameter:

-Input:

BYTE	b_BoardHandle	Handle of the APCI-/CPCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_ETM	Selection of the ETM counter (0 or 1)
UINT	ui_TimeOut	Selection of the timeout (0 to 65535) 0: Timeout not used. The function returns the ETM state and measures the counter value when a stop signal has arrived. 1 to 65535: Determines the timeout in ms. The function is stopped after a timeout or a stop signal.

-Output:

PBYTE	pb_ETMStatus	Return of the ETM state. 0: Measurement not started. No start trigger has arrived. 1: Measurement started. A start trigger has arrived. 2: Measurement stopped. A stop trigger has arrived. The measurement is stopped and <i>pul_ETMValue</i> returns the ETM time. 3: Transition has been realised. Please change the time base with the function "i_APCI1710_InitETM" 4: Timeout has been realised.
PULONG	pul_ETMValue	ETM time value.

Task:

Return of the ETM state (*pb_ETMStatus*) and of the time value (*pul_ETMValue*) after a stop signal on the selected module (*b_ModulNbr*). This function is only available when you disable the interrupt function. See function "i_APCI1710_EnableETM" and Table 3-4). The ETM state can be tested with the function "i_APCI1710_GetETMProgressStatus".

The value returned by *pul_ETMValue* is not the correct time value.

Use the "i_APCI1710_ConvertETMValue" function.

Otherwise, the following formula is applied in order to calculate the correct time value

Time value = *pul_ETMValue* x
pul_RealTiming.

pul_RealTiming is the returned value of "i_APCI1710_InitETM". The time base is the variable *b_TimingUnit* of the function "i_APCI1710_InitETM".

Calling convention:

ANSI C:

```
int          i_ReturnValue;
unsigned char b_BoardHandle;
unsigned char b_ETMStatus;
unsigned long ul_ETMValue;

i_ReturnValue = i_APCI1710_ReadETMValue
                (b_BoardHandle,
                 0,
                 0,
                 0
                 &pb_ETMStatus,
                 &ul_ETMValue);
```

Return value:

- 0: No error
- 1: Handle parameter of the board is wrong
- 2: The selected module number is wrong
- 3: The selected module is no "ETM" module.
- 4: Selected ETM counter is wrong.
- 5: ETM not initialised. See function "i_APCI1710_InitETM".
- 6: The timeout parameter is wrong (0 to 65535).
- 7: Interrupt routine installed. The measured ETM cannot be read directly.

3) i_APCI1710_ReadETMTotalTime

Syntax:

<Return value> = i_APCI1710_ReadETMTotalTime

(BYTE_ b_BoardHandle,
 BYTE_ b_ModulNbr,
 BYTE_ b_ETM,
 UINT_ ui_TimeOut,
 PBYTE_ pb_ETMstatus,
 PULONG_ pul_ETMValue)

Parameter:

-Input:

BYTE	b_BoardHandle	: Handle of the APCI-1710
BYTE	b_ModulNbr	: Number of the module to be configured (0 to 3)
BYTE	b_ETM	: Selection of the ETM (0 or 1)
UINT	ui_TimeOut	: Selection of the timeout (0 to 65535) If you use this function of the interrupt function, this parameter is not used. 0: Timeout not used. The function returns the ETM status and if a stop signal occurs the measured time value. 1 to 65535: Determines the timeout in ms. The function returns after a timeout or a stop signal occurred

-Output:

PULONG pul_ETMValue : ETM total time value.

Task:

Returns the ETM status (pb_ETMstatus) and the total time value (pul_ETMValue) after a stop signal occurred on the selected ETM module (b_ModulNbr). This function is only available if you have disabled the interrupt function. See function "i_APCI1710_EnableETM" and table 3-1. You can test the ETM status with "i_APCI1710_GetETMProgressStatus". The returned value of pul_ETMValue is not the actual measured time value. You must use the function "i_APCI1710_ConvertETMValue" or execute this operation in order to calculate the time value:

pul_ETMValue x pul_RealTiming.

pul_RealTiming is the returned parameter of "i_APCI1710_InitETM" and the time unit is the b_TimingUnit of the "i_APCI1710_InitETM" function.

Return value:

0: No error
 -1: Handle parameter of the board is wrong

- 2: The selected module number is wrong
- 3: The selected module is no ETM module
- 4: The selected ETM is wrong
- 5: ETM is not initialised. See function "i_APCI1710_InitETM"
- 6: Timeout parameter is wrong

4) i_APCI1710_ConvertETMValue (...)

Syntax:

```
<Return Wert> = i_APCI1710_ConvertETMValue
                                (BYTE      b_BoardHandle,
                                BYTE      b_ModulNbr,
                                ULONG     ul_ETMValue,
                                PULONG    pul_Hour,
                                PBYTE     pb_Minute,
                                PBYTE     pb_Second,
                                PUINT     pui_MilliSecond,
                                PUINT     pui_MicroSecond,
                                PUINT     pui_NanoSecond)
```

Parameter:

-Input:

BYTE	b_BoardHandle	Handle of the APCI-/CPCI-1710
BYTE	b_ModulNbr	Number of the module to be configured
ULONG	ul_ETMValue	ETM time value. See "i_APCI1710_ReadETMValue"

-Output:

PULONG	pul_Hour	Time measurement in hours.
PBYTE	pb_Minute	Time measurement in minutes
PBYTE	pb_Second	Time measurement in seconds.
PUINT	pui_MilliSecond	Time measurement in milliseconds.
PUINT	pui_MicroSecond	Time measurement in microseconds.
PUINT	pui_NanoSecond	Time measurement in nanoseconds

Task:

Conversion of the measured ETM time (*ul_ETMValue*) in h, mn, s, ms, μ s and ns.

Calling convention:

ANSI C:

```
int          i_ReturnValue;
unsigned char b_BoardHandle;
unsigned int  ui_MilliSecond;
unsigned int  ui_MicroSecond;
unsigned int  ui_NanoSecond;
unsigned char b_Second;
unsigned char b_Minute;
i_ReturnValue = i_APCI1710_ConvertETMValue
                (b_BoardHandle,
                0,
                0,
                &b_Minute,
                &b_Second,
                &ui_MilliSecond,
                &ui_MicroSecond,
                &ui_NanoSecond);
```

Return value:

- 0: No error
- 1: Handle parameter of the board is wrong
- 2: The selected module number is wrong
- 3: The selected module is no "ETM" module.
- 4: ETM is not initialised. See function "i_APCI1710_InitETM".

3.5 Functions in the kernel mode

i

IMPORTANT!

These functions are only available for the user of the interrupt routine under Windows NT and Windows 95 in the synchronous mode. See function "i_APCI1710_SetBoardIntRoutineWin32"

1) i_APCI1710_KRNL_ReadETMValue (...)

Syntax:

```
<Return Wert> = i_APCI1710_KRNL_ReadETMValue
                                (UINT          ui_BaseAddress,
                                 BYTE           b_ModulNbr,
                                 BYTE           b_ETM
                                 PBYTE          pb_ETMStatus,
                                 PULONG        pul_ETMValue)
```

Parameter:

-Input:

UINT	ui_BaseAddress	Base address of the APCI-/CPCI-1710 board
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_ETM	Selection of the ETM counter (0 or 1)

-Output:

PBYTE	pb_ETMStatus	Return of the ETM state. 0: Measurement is not started. No start trigger is arrived. 1: Measurement is started. A start trigger is arrived. 2: Measurement stopped. A stop trigger is arrived. The measurement is stopped and <i>pul_ETMValue</i> returns the ETM time. 3: Transition has been realised. Please change the time base with the function "i_APCI1710_InitETM"
PULONG	pul_ETMValue	ETM time value

Task:

Return of the ETM state (*pb_ETMStatus*) and of the time value (*pul_ETMValue*) after a stop signal onto the selected module (*b_ModulNbr*). This function is only available if you have disabled the interrupt function. See function "i_APCI1710_EnableETM" and Table 3-4.

The ETM state can be tested with the function "i_APCI1710_KRNL_GetETMProgressStatus" .

The through *pul_ETMValue* returned value is not the correct time value. Use the "i_APCI1710_ConvertETMValue" function.

Otherwise, the following formula applies in order to calculate the correct time value:

Time value = *pul_ETMValue* x *pul_RealTiming*.

pul_RealTiming is the returned value of "i_APCI1710_InitETM". The time unit is the variable *b_TimingUnit* of the function "i_APCI1710_InitETM".

Calling convention:

ANSI C:

```
int          i_ReturnValue;  
unsigned int  ui_BaseAddress;  
unsigned char b_ETMStatus;  
unsigned long ul_ETMValue;
```

```
i_ReturnValue = i_APCI1710_KRNL_ReadETMValue  
                (ui_BaseAddress,  
                 0,  
                 &pb_ETMStatus,  
                 &ul_ETMValue);
```

Return value:

0: No error

-1: The selected module number is wrong.

-2: The selected module is no "ETM" module.

-3: Selected ETM counter is wrong.

-4: ETM not initialised. See function "i_APCI1710_InitETM".

-5: Interrupt routine installed. The measured ETM time cannot be read directly.