



DIN EN ISO 9001:2000
certified



ADDI-DATA GmbH
Airpark Business Center
Airport Boulevard B210
77836 Rheinmünster
Germany



Technical support:
+49 7229 1847 - 0

Function description

APCI-1710

Multifunction counter board
- TTL inputs and outputs -

Edition: 01.02 - 05/2008

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 use6
 - 1.2 Usage restrictions.....6
 - 1.3 Technical documentation6
 - 1.4 Function description7

- 2 TTL INPUTS AND OUTPUTS 8**
 - 2.1 Function description8
 - 2.1.1 Typical applications8
 - 2.2 Used signals8
 - 2.3 Pin assignment of connector ST510
 - 2.4 Connection example12
 - 2.5 I/O mapping13
 - 2.6 Description of the I/O functions.....14
 - 2.6.1 Function description 14
 - 2.6.2 Description register (Base + 60) 14

- 3 STANDARD SOFTWARE 15**
 - 3.1 Introduction15
 - 3.2 Initialisation.....16
 - 1) i_APCI1710_InitTTLIODirection (...)16
 - 3.3 Read TTL I/O18
 - 1) i_APCI1710_ReadTTLIOChannelValue (...).....18
 - 2) i_APCI1710_ReadTTLIOPortValue (...)20
 - 3) i_APCI1710_ReadTTLIOAllPortValue (...)21
 - 3.4 Write TTL I/O22
 - 1) i_APCI1710_SetTTLIOChlOn (...).....22
 - 2) i_APCI1710_SetTTLIOChlOff (...)23
 - 3.5 Using the functions in the kernel module.....24
 - 1) i_APCI1710_KRNL_ReadTTLIOChannelValue (...).....24
 - 2) i_APCI1710_KRNL_ReadTTLIOPortValue (...)26
 - 3) i_APCI1710_KRNL_ReadTTLIOAllPortValue (...)27
 - 4) i_APCI1710_KRNL_SetTTLIOChlOn (...)28
 - 5) i_APCI1710_KRNL_SetTTLIOChlOff (...)29

Figures

Fig. 2-1: Pin assignment of the connector ST5 and the ribbon cable
 FB800010
 Fig. 2-2: Connection example12

Tables

Table 1-1: Delivered manuals7
 Table 2-1: Used signals.....9
 Table 2-2: Description of the pin assignment11
 Table 2-3: I/O mapping of the TTL I/O functions.....13
 Table 3-1: Define value15

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

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

1.3 Technical documentation

This manual refers to the **APCI-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: 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	tll_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

1.4 Function description

Besides a global description of the software functions this manual contains:

- The pin assignment of the front connector,
- A list of the signals used,
- The I/O mapping,
- A chapter on the API software functions of the supplied standard software.

2 TTL INPUTS AND OUTPUTS

2.1 Function description

On the **APCI-1710** board digital I/O, GND as well as Vcc of the PC are available via the connector **ST5**. The signals have to comply with the TTL level and be operated carefully: The board could be damaged if other signals were to be connected.

The ribbon cable **FB8000** connects the board to the peripheral through the connector **ST5**.

The signals PA0 to PA7, PB0 to PB7 and PC0 to PC7 are connected to all function modules. They can either be used for only one "function module" (e.g. "TTL I/O") or be distributed to each module.

Example: Function module 4 is programmed only with the function TTL I/O. The signals can be used either as input or as output channels (according to the function programmed).

This function stands out for its wide range of applications, its high precision and reliability for tough industrial applications.

Characteristics:

- Complete isolation through optical couplers for the input and output channels to avoid earth circuit.
- Inputs and outputs can be inverted by software.
- Hardware and software GATE are rereadable

2.1.1 Typical applications

- Digital "one-shot"
- Real-time clock
- Event counter
- Programmable rate generator
- Frequency generator
- Complex signal generator

2.2 Used signals

The signals PA0 to PA7, PB0 to PB7 and PC0 to PC7 are all connected to the function modules over the connector **ST5**. They can either be used for only one function module (e.g. "TTL I/O") or be distributed to each module.

2 TTL inputs and outputs are available (I and J) for each function module (FM1 to FM4). **These signals can only be used in connection with the TTL I/O function module.**

The function "TTL I/O" connects the **XX inputs and XX outputs** of the respecting function module of the **APCI-/CPCI-1710**.

Table 2-1: Used signals

AT CONNECTOR ST5	POLARITY	FUNCTION
PA 0 to PA 7	TTL	Input or output, after reset: Input
PB 0 to PB 7	TTL	Input or output, after reset: Input
PC 0 to PC 7	TTL	Input or output, after reset: Input
GND	PC GND	GND, not isolated
Kx	TTL	TTL output, same signal as Hx at the front connector
Ix	TTL	Input or output, after reset: Output
Jx	TTL	Input or output, after reset: Output
V. ext	PC +5 V	Voltage supply

x: Number of the function module.

2.3 Pin assignment of connector ST5

Fig. 2-1: Pin assignment of the connector ST5 and the ribbon cable FB8000

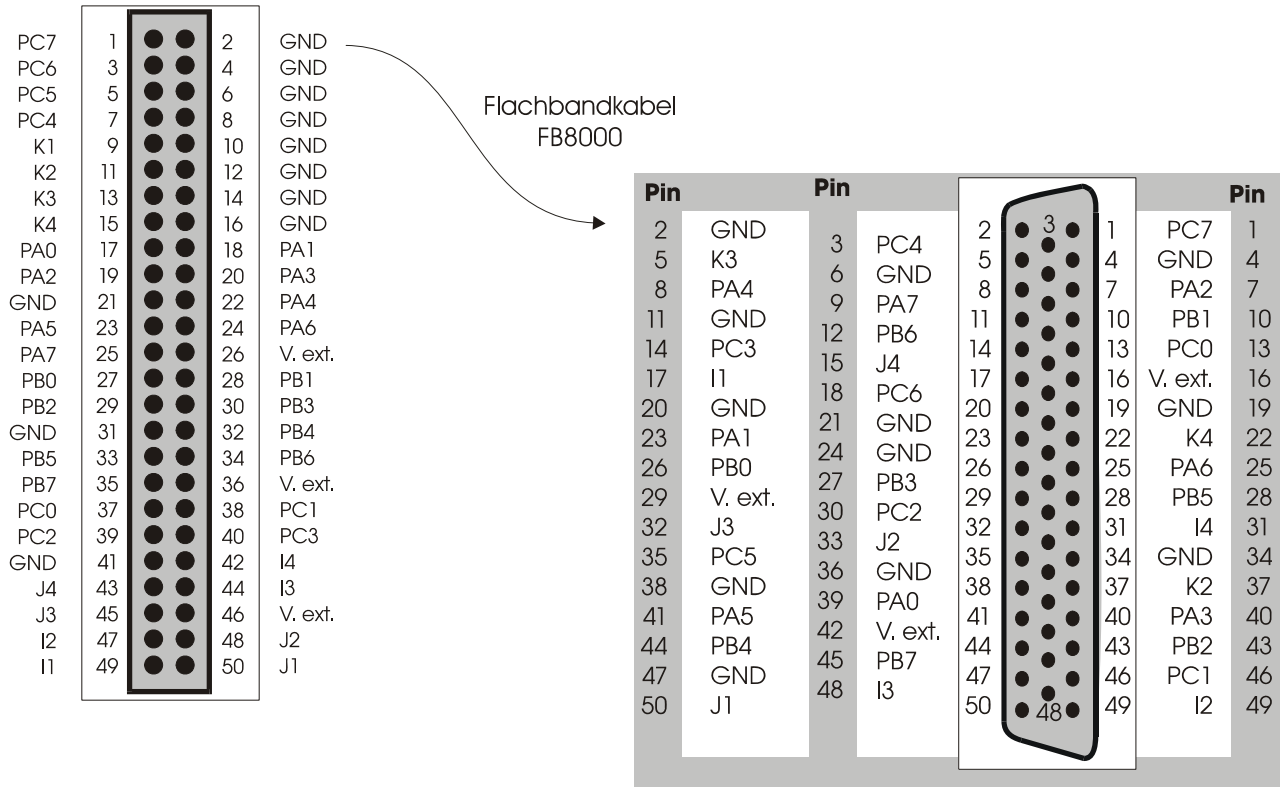


Table 2-2: Description of the pin assignment

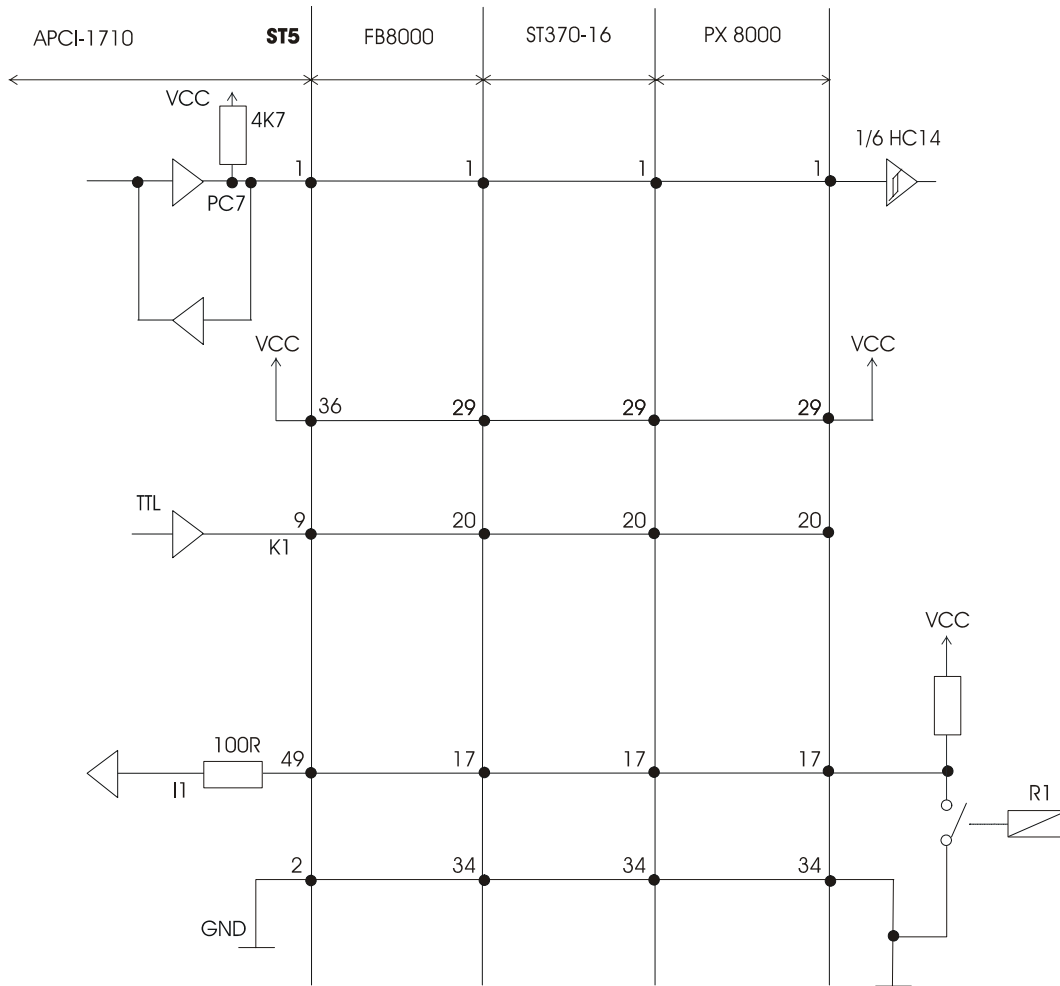
Pin number at the connector	Name	Pin number at FB8000	Pin number at the connector	Name	Pin number at FB8000
1	PC7 ¹	1	26	V. ext	42
2	GND	34	27	PB0	26
3	PC6	18	28	PB1	10
4	GND	2	29	PB2	43
5	PC5	35	30	PB3	27
6	GND	19	31	GND	11
7	PC4	3	32	PB4	44
8	GND	36	33	PB5	28
9	K1	20	34	PB6	12
10	GND	4	35	PB7	45
11	K2	37	36	V. ext	29
12	GND	21	37	PC0	13
13	K3	5	38	PC1	46
14	GND	38	39	PC2	30
15	K4	22	40	PC3	14
16	GND	6	41	GND	47
17	PA0	39	42	I4 ²	31
18	PA1	23	43	J4 ²	15
19	PA2	7	44	I3 ²	48
20	PA3	40	45	J3 ²	32
21	GND	24	46	V. ext	16
22	PA4	8	47	I2 ²	49
23	PA5	41	48	J2 ²	33
24	PA6	25	49	I1 ²	17
25	PA7	9	50	J1 ²	50

1 PA, PB and PC are equipped with pullup resistance on 5 V

2 Serial resistance 100 Ω

2.4 Connection example

Fig. 2-2: Connection example



ST5: Connector of the APCI-1710/APCI-1710-24V for connecting ribbon cable FB8000

2.5 I/O mapping

Table 2-3: I/O mapping of the TTL I/O functions

	IORD			
	D31...D24	D23...D16	D15.....D8	D7.....D0
BYTES				
BASEx + 0	Port D, J, I	Port C as input	Port B as input	Port A as input
BASEx + 4				
BASEx + 8				
BASEx + 12				
BASEx + 16				
				Control Word
.....				
BASEx + 60				

	IOWR			
	D31...D24	D23...D16	D15.....D8	D7.....D0
BYTES				
BASEx + 0				Output I
BASEx + 4				Output J
BASEx + 8				Port A as output
BASEx + 12				Port B as output
BASEx + 16				Port C as output
BASEx + 20				Control Word
.....				
BASEx + 60				

-: No function, y: Data of no importance, x: Number of the function module

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

2.6 Description of the I/O functions

2.6.1 Function description

Ports A, B, C are defined as inputs after reset, port D (I, J) is switched as output after the reset.

Ports A, B, C, D are all set on V_{CC} over 4K7 Pull Up resistances.

2.6.2 Description register (Base +60)

The function and the revision are identified (reading command, ASCII format)

BASE + 60 "T" "L" "2" "0"

Meaning: TTL E/A Revision 2.0

3 STANDARD SOFTWARE

3.1 Introduction



IMPORTANT!

Remember 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 Initialisation

1) i_APCI1710_InitTTLIODirection (...)

Syntax:

```
<Return Wert> = i_APCI1710_InitTTLIODirection
                    (BYTE b_BoardHandle,
                     BYTE b_ModulNbr,
                     BYTE b_PortAMode,
                     BYTE b_PortBMode,
                     BYTE b_PortCMode,
                     BYTE b_PortDMode)
```

Parameter:

-Input:

BYTE	b_BoardHandle	Handle parameter of the APCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_PortAMode	Sets the direction of port A 0: Used as input port 1: Used as output board.
BYTE	b_PortBMode	Sets the direction of port B 0: Used as input port 1: Used as output port
BYTE	b_PortCMode	Sets the direction of port C 0: Used as input port 1: Used as output port
BYTE	b_PortDMode	Sets the direction of port D 0: Used as input port 1: Used as output board

-Output:

There is no output

Task:

Configures the operation mode for the selected "TTL I/O" module (*b_ModulNbr*).
Call this function before you call other functions that access to TTL I/O.

Calling convention:

ANSI C:

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

```
i_ReturnValue = i_APCI1710_InitTTLIODirection
                (b_BoardHandle,
                 0,
                 0,
                 0,
```

0,
1);

Return value:

0: No error

-1: The handle parameter of the board is wrong.

-2: Wrong module selection.

-3: The selected module is no "TTL I/O" module

-4: Function is not available for this board version

-5: Selected mode for port A is wrong

-6: Selected mode for port B is wrong

-7: Selected mode for port C is wrong

-8: Selected mode for port D is wrong

3.3 Read TTL I/O

1) `i_APCI1710_ReadTTLIOChannelValue (...)`

Syntax:

```
<Return value> = i_APCI1710_ReadTTLIOChannelValue
                    (BYTE    b_BoardHandle,
                     BYTE    b_ModulNbr,
                     BYTE    b_SelectedPort,
                     BYTE    b_InputChannel,
                     PBYTE   pb_ChannelStatus)
```

Parameter:

-Input:

BYTE	b_BoardHandle	Handle parameter of the APCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_SelectedPort	Selection of the TTL I/O port (0 to 3) 0: Port A 1: Port B 2: Port C 3: Port D
BYTE	b_InputChannel	Selection of the TTL input channel 0 to 7 for port A,B,C 0 or 1 for port D

-Output:

PBYTE	pb_ChannelStatus	Status of the digital input 0: Channel is deactivated 1: Channel is activated.
-------	------------------	--

Task:

Returns the status of the TTL input (*b_InputChannel*) for the respecting module (*b_ModulNbr*).

Calling convention:

ANSI C:

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

```
i_ReturnValue = i_APCI1710_ReadTTLIOChannelValue
                (b_BoardHandle,
                 0, 0, 0,
                 b_ChannelStatus);
```

Return value:

0: No error

- 1: The handle parameter of the board is wrong
- 2: Wrong module selection
- 3: The selected module is no "TTL I/O" module
- 4: The selected TTL input port is wrong
- 5: The selected TTL input is wrong
- 6: TTL I/O is not initialised. See function: "i_APCI1710_InitTTLIO"

2) i_APCI1710_ReadTTLIOPortValue (...)

Syntax:

```
<Return value> = i_APCI1710_ReadTTLIOPortValue
                    (BYTE  b_BoardHandle,
                     BYTE  b_ModulNbr,
                     BYTE  b_SelectedPort,
                     PBYTE pb_PortValue)
```

Parameter:

-Input:

BYTE	b_BoardHandle	Handle parameter of the APCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_SelectedPort	Selection of the TTL I/O port (0 to 3) 0: Port A 1: Port B 2: Port C 3: Port D

-Output:

PBYTE	pb_PortValue	Status of the digital TTL input.
-------	--------------	----------------------------------

Task:

Returns the status of the digital input port (*b_SelectedPort*) for the selected TTL I/O module (*b_ModulNbr*).

Calling convention:

ANSI C:

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

```
i_ReturnValue = i_APCI1710_ReadTTLIOPortValue
                (b_BoardHandle,
                 0,
                 0,
                 &b_PortValue);
```

Return value:

0: No error
-1: The handle parameter of the board is wrong.
-2: Wrong module selection
-3: The selected module is no "TTL I/O" module
-4: The selected TTL input port is wrong.
-5: TTL I/O not initialised. See function: "i_APCI1710_InitTTLIO".

3) i_APCI1710_ReadTTLIOAllPortValue (...)**Syntax:**

```
<Return value> = i_APCI1710_ReadTTLIOAllPortValue
                    (BYTE b_BoardHandle,
                     BYTE    b_ModulNbr,
                     PULONG  pul_AllPortValue)
```

Parameter:**-Input:**

BYTE	b_BoardHandle	Handle parameter of the APCI-1710
BYTE	b_ModulNbr	Number of the module to be configured (0 to3)

-Output:

PULONG	pul_AllPortValue	Status of the digital TTL input port A, B, C and D.
--------	------------------	---

Task:

Returns the status of all digital inputs ports (port A, B, C and D) for the selected TTL I/O module (*b_ModulNbr*).

Calling convention:

ANSI C:

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

```
i_ReturnValue = i_APCI1710_ReadTTLIOAllPortValue
                (b_BoardHandle,
                 0,
                 & ul_AllPortValue);
```

Return value:

0: No error

-1: The handle parameter of the board is wrong.

-2: Wrong module selection.

-3: The selected module is no "TTL I/O" module

-4: TTL I/O is not initialised. See function: "i_APCI1710_InitTTLIO"

3.4 Write TTL I/O

1) i_APCI1710_SetTTLIOChlOn (...)

Syntax:

```
<Return Wert> = i_APCI1710_SetTTLIOChlOn
                                     (BYTE   b_BoardHandle,
                                     BYTE   b_ModulNbr,
                                     BYTE   b_OutputChannel)
```

Parameter:

-Input:

BYTE	b_BoardHandle	Handle parameter of the APCI-1710
BYTE	b_ModulNbr	Number of the selected module (0 to 3)
BYTE	b_OutputChannel	Selection of the digital output channel (0 to 25) 0: PD0 1: PD1 2 to 9: PA0 to PA7 10 to 17: PB0 to PB7 18 to 25: PC0 to PC7

-Output:

There is no output

Task:

Sets the output that is inserted via the parameter Parameter *b_Channel* . Setting an output means to set an output to "High".

Calling convention:

ANSI C:

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

```
i_ReturnValue = i_APCI1710_SetTTLIOChlOn
               (b_BoardHandle,
               0,
               0);
```

Return value:

0: No error
-1: The handle parameter of the board is wrong
-2: Wrong module selection
-3: The module is no "TTL I/O" module.
-4: The selection of the TTL output is wrong. .
-5: TTL I/O is not initialised. See function: "i_APCI1710_InitTTLIO".

2) i_APCI1710_SetTTLIOChlOff (...)

Syntax:

```
<Return Wert> = i_APCI1710_SetTTLIOChlOff
                    (BYTE b_BoardHandle,
                     BYTE      b_ModulNbr,
                     BYTE      b_OutputChannel)
```

Parameter:

-Input:

BYTE	b_BoardHandle	Handle parameter of the APCI-1710
BYTE	b_ModulNbr	Number of the selected module (0 to 3)
BYTE	b_OutputChannel	Selection of the digital output channel (0 to 25) 0: PD0 1: PD1 2 to 9: PA0 to PA7 10 to 17: PB0 to PB7 18 to 25: PC0 to PC7

- Output:

There is no output.

Task:

Resets the output that is entered with the parameter *b_Channel*. Resetting an output means setting an output to "Low".

Calling convention:

ANSI C:

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

```
i_ReturnValue = i_APCI1710_SetTTLIOChlOff
                (b_BoardHandle,
                 0,
                 0);
```

Return-value:

0: No error
-1: The handle parameter of the board is wrong.
-2: Wrong module selection
-3: The module is no "TTL I/O" module.
-4: The selection of the TTL output is wrong.
-5: TTL I/O not initialised. See function "i_APCI1710_InitTTLIO"

3.5 Using the functions in the kernel module

i

IMPORTANT!

This function is only available for the user interrupt routine in the synchronous mode under Windows NT and Windows 95/98.
See function "i_APCI1710_SetBoardIntRoutineWin32".

1) i_APCI1710_KRNL_ReadTTLIOChannelValue (...)

Syntax:

```
<Return value> = i_APCI1710_KRNL_ReadTTLIOChannelValue
                    (UINT      ui_BaseAddress,
                     BYTE      b_ModulNbr,
                     BYTE      b_SelectedPort,
                     BYTE      b_InputChannel,
                     PBYTE     pb_ChannelStatus)
```

Parameter:

-Input:

UINT	ui_BaseAddress	Base address of the board
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_SelectedPort	Selection of the TTL I/O port (0 to 3) 0: Port A 1: Port B 2: Port C 3: Port D
BYTE	b_InputChannel	Selection of the TTL input 0 to 7 for port A,B,C 0 or 1 for port D

-Output:

PBYTE	pb_ChannelStatus	Status of the digital input. 0: Channel is not active 1: Channel is activated
-------	------------------	---

Task:

Returns the status of the selected TTL input (*b_InputChannel*) for the module (*b_ModulNbr*).

Calling convention:

ANSI C:

```
int          i_ReturnValue;  
unsigned int ui_BaseAddress  
unsigned char b_ChannelStatus;
```

```
i_ReturnValue = i_APCI1710_KRNL_ReadTTLIOChannelValue  
                (ui_BaseAddress,  
                 0,  
                 0,  
                 0,  
                 &b_ChannelStatus);
```

Return value:

0: No error

-1: Wrong module selection

-2: The module is no "TTL I/O" module.

-3: The selected TTL input port is wrong.

-4: The selected TTL input is wrong.

2) i_APCI1710_KRNL_ReadTTLIOPortValue (...)

Syntax:

```
<Return value> = i_APCI1710_KRNL_ReadTTLIOPortValue
                    (UINT      ui_BaseAddress,
                     BYTE      b_ModulNbr,
                     BYTE      b_SelectedPort,
                     PBYTE     pb_PortValue)
```

Parameter:

-Input:

UINT	ui_BaseAddress	Base address of the board
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)
BYTE	b_SelectedPort	Selection of the TTL I/O port (0 to 3) 0: Port A 1: Port B 2: Port C 3: Port D

-Output:

PBYTE	pb_PortValue	Status of the digital TTL input port
-------	--------------	--------------------------------------

Task:

Returns the status of the digital input port (*b_SelectedPort*) for the selected module (*b_ModulNbr*).

Calling convention:

ANSI C:

```
int          i_ReturnValue;
unsigned int ui_BaseAddress;
unsigned char b_PortValue;
```

```
i_ReturnValue = i_APCI1710_KRNL_ReadTTLIOPortValue
                (ui_BaseAddress,
                 0,
                 0,
                 &b_PortValue);
```

Return value:

0: No error
-1: Wrong module selection
-2: The module is no "TTL I/O" module.
-3: The selected TTL input port is wrong.

3) `i_APCI1710_KRNL_ReadTTLIOAllPortValue (...)`

Syntax:

```
<Return value> = i_APCI1710_KRNL_ReadTTLIOAllPortValue
                    (UINT          ui_BaseAddress,
                     BYTE          b_ModulNbr,
                     PULONG       pul_AllPortValue)
```

Parameter:

-Input:

UINT	ui_BaseAddress	Base address of the board
BYTE	b_ModulNbr	Number of the module to be configured (0 to 3)

-Output:

PULONG	pul_AllPortValue	Status of the digital TTL input port A, B, C and D.
--------	------------------	--

Task:

Returns the status of all digital input ports (port A, B, C and D) for the selected TTL I/O module (*b_ModulNbr*).

Calling convention:

ANSI C:

```
int          i_ReturnValue;
unsigned int  ui_BaseAddress;
unsigned long ul_AllPortValue;
```

```
i_ReturnValue = i_APCI1710_KRNL_ReadTTLIOAllPortValue
                (ui_BaseAddress,
                 0,
                 &ul_AllPortValue);
```

Return value:

0: No error
-1: Wrong module selection
-2: The module is no "TTL I/O" module

4) `i_APCI1710_KRNL_SetTTLIOChlOn (...)`

Syntax:

```
<Return value> = i_APCI1710_KRNL_SetTTLIOChlOn
                    (UINT          ui_BaseAddress,
                     BYTE          b_ModulNbr,
                     BYTE          b_OutputChannel)
```

Parameter:

-Input:

UINT	ui_BaseAddress	Base address of the board
BYTE	b_ModulNbr	Number of the selected module (0 to 3)
BYTE	b_OutputChannel	Selection of the digital output channel (0 or 25) 0: PDO 1: PD1 2 to 9: PA0 to PA7 10 to 17: PB0 to PB7 18 to 25: PC0 to PC7

-Output:

There is no output.

Task:

Sets the output that is entered with the parameter *b_Channel*. Setting an output means setting an output to "High".

Calling convention:

ANSI C:

```
int          i_ReturnValue;
unsigned int ui_BaseAddress;
```

```
i_ReturnValue = i_APCI1710_KRNL_SetTTLIOChlOn
                (ui_BaseAddress
                 0
                 0);
```

Return value:

0: No error
-1: Wrong module selection
-2: The module is no "TTL I/O" module
-3: The selection of the TTL output is wrong.

5) **i_APCI1710_KRNL_SetTTLIOChIOff (...)**

Syntax:

```
<Return Wert> = i_APCI1710_KRNL_SetTTLIOChIOff
                    (UINT      ui_BaseAddress,
                     BYTE      b_ModulNbr,
                     BYTE      b_OutputChannel)
```

Parameter:

-Input:

UINT	ui_BaseAddress	Base address of the board
BYTE	b_ModulNbr	Number of the selected module (0 to 3)
BYTE	b_OutputChannel	Selection of the digital output channel (0 or 25) 0: PD0 1: PD1 2 to 9: PA0 to PA7 10 to 17: PB0 to PB7 18 to 25: PC0 to PC7

-Output:

There is no output.

Task:

Resets the output that is entered with the parameter *b_Channel*. Resetting an output means setting an output on "Low".

Calling convention:

ANSI C:

```
int          i_ReturnValue;
unsigned int ui_BaseAddress;
```

```
i_ReturnValue = i_APCI1710_KRNL_SetTTLIOChIOff
                (ui_BaseAddress,
                 0,
                 0);
```

Return value:

0: No error
-1: Wrong module selection
-2: The module is no "TTL I/O" module.
-3: The selection of the TTL output is wrong.