



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



**Technical description**

**PA 030**

**Watchdog board**

Edition: 07.01 - 10/2006

## Copyright

All rights reserved. This manual is intended for the manager and its personnel.  
No part of this publication may be reproduced or transmitted by any means.  
Offences can have penal consequences.

## Guarantee and responsibility

Basically are effective our "general terms of delivery and payment". The manager receives them at the latest with the invoice. Claims for guarantee and responsibility in case of injuries and material damages are excluded, if they are due to one or some of the following causes:

- if the board has not been used for the intended purpose
- improper installation, operation and maintenance of the board
- if the board has been operated with defective safety devices or with not appropriate or non-functioning safety equipment
- nonobservance of the instructions concerning: transport, storage, inserting the board, use, limit values, maintenance, device drivers
- altering the board at the user's own initiative
- altering the source files at the user's own initiative
- not checking properly the parts which are subject to wear
- disasters caused by the intrusion of foreign bodies and by influence beyond the user's control.

## Licence for ADDI-DATA software products

Read carefully this licence before using the standard software. The right for using this software is given to the customer, if he/she agrees to the conditions of this licence.

- this software can only be used for configuring ADDI-DATA boards.
- copying the software is forbidden (except for archiving/ saving data and for replacing defective data media).
- deassembling, decompiling, decoding and reverse engineering of the software are forbidden.
- this licence and the software can be transferred to a third party, so far as this party has purchased a board, declares to agree to all the clauses of this licence contract and the preceding owner has not kept copies of the software.

## Trademarks

Borland C and Turbo Pascal are registered trademarks of Borland International, INC.

Burr-Brown is a registered trademark of Burr-Brown Corporation

Intel is a registered trademark of Intel Corporation

Microsoft, MS-DOS, Visual Basic and Windows are registered trademarks of Microsoft Corporation

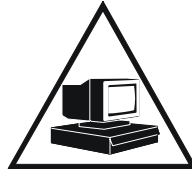
***The original version of this manual is in German. You can obtain it on request.***

# WARNING

**In case of improper use and if the board is not used for the intended purpose:**



**people may be injured**



**the board, PC and peripheral may be damaged**



**the environment may be polluted**

**★★★ Protect yourself, other people and the environment★★★**

- **Do read the safety leaflet!**

If this leaflet is not with the manual, please contact us.

- **Observe the instructions in the manual !**

Make sure that you do not forget or skip any step. We are not liable for damage resulting from an improper use of the board.

- **Symbols used**



## **WARNING!**

It designates a possibly dangerous situation.

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



## **IMPORTANT!**

designates hints and other useful information.

- **Do you have any questions?**

Our technical support is at your disposal.







<b>1</b>	<b>INTENDED PURPOSE OF THE BOARD</b> .....	<b>1</b>
1.1	Limits of use.....	1
<b>2</b>	<b>USER</b> .....	<b>2</b>
2.1	Qualification .....	2
2.2	Personal protection.....	2
<b>3</b>	<b>HANDLING THE BOARD</b> .....	<b>3</b>
<b>4</b>	<b>TECHNICAL DATA</b> .....	<b>4</b>
4.1	Electromagnetic compatibility .....	4
4.2	Physical set-up of the board.....	4
4.3	Limit values.....	5
<b>5</b>	<b>SETTINGS</b> .....	<b>6</b>
5.1	Component scheme.....	6
5.2	Jumper location and settings at delivery .....	7
5.2.1	Jumper location on the board .....	7
5.2.2	Jumper settings .....	8
5.3	I/O mapping .....	10
<b>6</b>	<b>INSTALLATION</b> .....	<b>11</b>
6.1	Setting the base address through DIP switches.....	12
6.2	Inserting the board.....	13
6.2.1	Opening the PC.....	13
6.2.2	Plugging the board into the slot.....	14
6.2.3	Closing the PC .....	14
6.3	Installing the software.....	15
6.3.1	Software installation under MS-DOS and Windows 3.11 .....	15
6.3.2	Software installation under Windows NT/95/98 .....	15
6.4	Board configuration with ADDIREG.....	16
6.4.1	Program description .....	16
6.4.2	Registering a new board .....	19
6.4.3	Changing the registration of a board .....	20
6.4.4	Removing the ADDIREG program.....	21
6.5	Software downloads from the Internet.....	21
<b>7</b>	<b>CONNECTION TO THE PERIPHERAL</b> .....	<b>22</b>
7.1	Connector pin assignment.....	22

<b>8</b>	<b>FUNCTIONS .....</b>	<b>23</b>
<b>8.1</b>	<b>Introduction .....</b>	<b>23</b>
<b>8.2</b>	<b>Programming and operation .....</b>	<b>24</b>
8.2.1	Temperature.....	24
	Initiating the temperature measuring .....	25
	Read temperature (ADCDATA) .....	25
	Read status (ADCSTATUS) .....	25
	Automatic temperature measuring (ADCMODE).....	25
8.2.2	Triggering the timer DOG .....	26
	Timer programming.....	26
<b>8.3</b>	<b>Timer and watchdog .....</b>	<b>26</b>
8.3.1	Timer.....	26
8.3.2	Watchdog timer .....	27
8.3.3	Possibility of interrupts.....	28
8.3.4	Reset .....	28
8.3.5	Timer programming.....	29
	Programming examples.....	30
8.3.6	Watchdog programming .....	31
	Programming examples.....	31
<b>8.4</b>	<b>Interrupt.....</b>	<b>32</b>
<b>9</b>	<b>STANDARD SOFTWARE.....</b>	<b>33</b>
<b>9.1</b>	<b>Introduction .....</b>	<b>33</b>
<b>9.2</b>	<b>Software functions (API).....</b>	<b>34</b>
9.2.1	Initialisation .....	34
	1) i_PA030_SetBoardInformation .....	34
	2) i_PA030_CloseBoardHandle (...).....	34
9.2.2	Timer/watchdog functions .....	35
	1) i_PA030_ReadTemp .....	35
	2) i_PA030_ReadAdcState .....	35
	3) i_PA030_AutoMode.....	35
	4) i_PA030_ManuMode .....	36
	5) i_PA030_SetDogPeriod .....	36
	6) i_PA030_RetriggerDog.....	36
	7) i_PA030_ReadDogState.....	36
	8) i_PA030_SetTimer0Periode.....	37
	9) i_PA030_SetTimer1Periode.....	37
	10) i_PA030_EnaEnable.....	37
	11) i_PA030_EnaDisable .....	37
<b>9.3</b>	<b>Responding INIT030 functions.....</b>	<b>38</b>
<b>9.4</b>	<b>ERROR.TXT (only for 16-bit drivers).....</b>	<b>38</b>
<b>9.5</b>	<b>Program TEST030.EXE (only for 16-bit drivers) .....</b>	<b>38</b>
9.5.1	Input parameters.....	38
9.5.2	Display DIP switches.....	39
9.5.3	Board test .....	39
	Watchdog retrigger .....	39



Temperature measuring .....	39
-----------------------------	----

<b>SYSTEM ADDRESS RANGES .....</b>	<b>A</b>
------------------------------------	----------

## INDEX C

## Figures

Fig. 3-1: Wrong handling .....	3
Fig. 3-2: Correct handling .....	3
Fig. 5-1: Component scheme .....	6
Fig. 5-2: Jumper location and settings at delivery .....	7
Fig. 6-1: DIP switches S1 .....	12
Fig. 6-2: types of slots .....	13
Fig. 6-4: Inserting the board .....	14
Fig. 6-5: Securing the board at the back cover .....	14
Fig. 6-6: The ADDIREG registration program .....	16
Fig. 6-7: Configuring a new board .....	18
Fig. 6-8: The ADDI_UNINSTALL program .....	21
Fig. 7-1: 9-pin SUB-D female connector .....	22
Fig. 8-1: Block diagram .....	23
Fig. 8-2: Wiring principle of the timers .....	27
Fig. 8-3: Output wiring for watchdog signals .....	27

## Tables

Table 5-1: Jumper settings - Watchdog .....	8
Table 5-2: Jumper settings - Output status after running down of the watchdog .....	8
Table 5-3: Jumper settings - Interrupt wiring .....	9
Table 5-4: I/O mapping .....	10
Table 6-1: Decoding table .....	12
Table 8-1: Timer modes .....	29
Table 9-1: Type Declaration .....	33



# 1 INTENDED PURPOSE OF THE BOARD

The **PA 030** board is intended for the supervision of the software running in a personal computer (PC).

It is to be used in a free PC ISA slot. The PC is to comply with the EU directive 89/336/EEC and the specifications for EMC protection.

Products complying with these specifications bear the  mark.

The following signals are available on the front connector:

- the watchdog signal,
- the change-over contacts of a relay,
- the output channel of an open collector,
- the TTL output channel.

The watchdog signal is connected through a shielded cable to the reset input of the unit to be supervised.

The use of the board in a PC could change the PC features regarding noise emission and immunity. Increased noise emission or decreased noise immunity could result in the system not being conform anymore. Check the shielding capacity of the PC prior to putting the device into operation.

The use of the board according to its intended purpose includes observing all advice given in this manual and the safety leaflet. Uses beyond these specifications are not allowed.

The manufacturer is not liable for any damages which would result from the non-observance of this clause.

## 1.1 Limits of use

**Our boards are not to be used for securing emergency stop functions.**

The emergency stop functions are to be secured separately.  
This securing must not be influenced by the board or the PC.

Make sure that the board remains in the protective blister pack **until it is used**.

Do not remove or alter the identification numbers of the board.  
If you do, the guarantee expires.

## **2 USER**

### **2.1 Qualification**

Only persons trained in electronics are entitled to perform the following work:

- installation,
- use,
- maintenance.

### **2.2 Personal protection**

Consider the country-specific regulations about

- the prevention of accidents,
- electrical and mechanical installations,
- radio interference suppression.

### 3 HANDLING THE BOARD

Fig. 3-1: Wrong handling

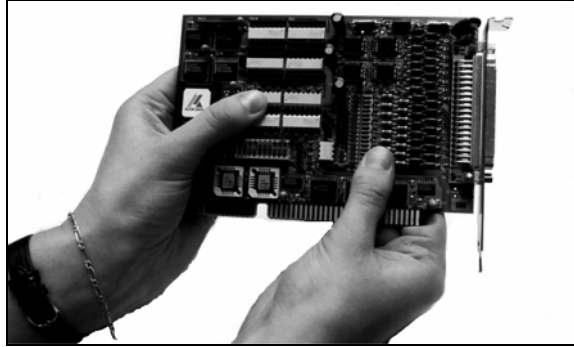
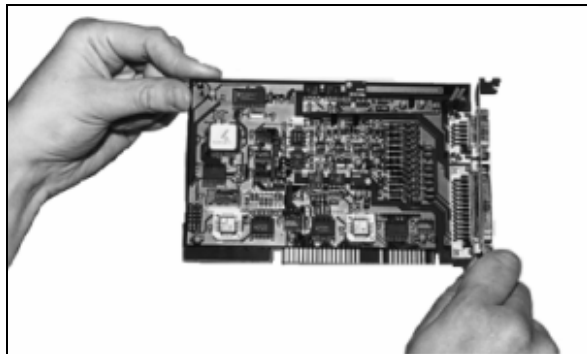


Fig. 3-2: Correct handling



## 4 TECHNICAL DATA

### 4.1 Electromagnetic compatibility

The board has been subjected to EMC tests in an accredited laboratory in accordance with the norms EN50082-2, EN55011, EN55022.

The board complies as follows with the limit values set by the norm EN50082-2:

	<u>True value</u>	<u>Set value</u>
ESD.....	4 kV	4 kV
Fields .....	10 V/m	10 V/m
Burst.....	2 kV	2 kV
Conducted radio interferences .....	10 V	10 V
Noise emissions .....	B-class	



#### **WARNING!**

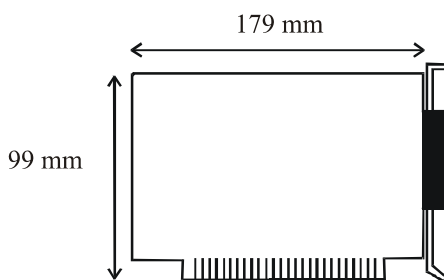
The EMC tests have been carried out in a specific appliance configuration. We guarantee these limit values **only** in this configuration<sup>1)</sup>.

#### **Consider the following aspects:**

- your test program must be able to detect operation errors.
- your system must be set up so that you can find out what caused errors.

### 4.2 Physical set-up of the board

The board is assembled on a 4-layer printed circuit card.



Weight: 110 g  
 Installation in: XT / AT slot  
 Connection to the peripheral through 9-pin SUB-D female connector

<sup>1)</sup> We transmit our appliance configuration on request.

### 4.3 Limit values

Operating temperature: ..... 0 to 70°C  
Storage temperature: ..... -20 to 80°C  
Relative humidity: ..... 80% non condensing

**Minimum PC requirements:**

ISA bus interface

Bus speed: ..... 8 MHz

**Energy requirements:**

Operating voltage from the PC: ..... 5 V  $\pm$  5%

12 V  $\pm$  12%

Current consumption in mA: ..... 5 V: 65 mA

12 V: 10 mA

Timer: ..... 2 x 16-bit cascaded,  
programmable

Watchdog timer: ..... 16-bit, programmable  
through software

Temperature measurement range: ..... 0-125°C

Resolution: ..... 0.5°C per digit

Conversion time: ..... Typ. 100  $\mu$ s

**Signals on the front connector**

Timer output channel: ..... TTL  $\leq$  5 mA

Watchdog output channel: ..... Typ.  $\cong$  80 ms

TTL  $\leq$  5 mA

O.C: ..... max. 20 V /30 mA

Relay: switching current ..... max. 0.25 A,

switching current with closed contacts: ..... max. 1 A

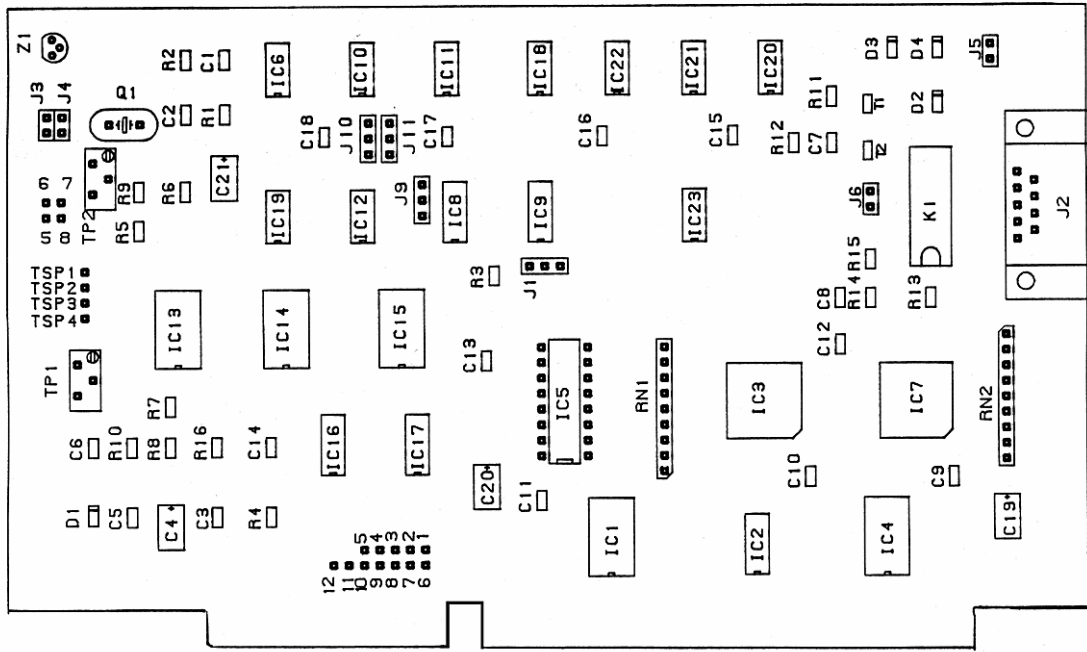
Reset input channel: ..... TTL input channel

5 V output channel: ..... 5 V through 560 R pull up

# 5 SETTINGS

## 5.1 Component scheme

Fig. 5-1: Component scheme

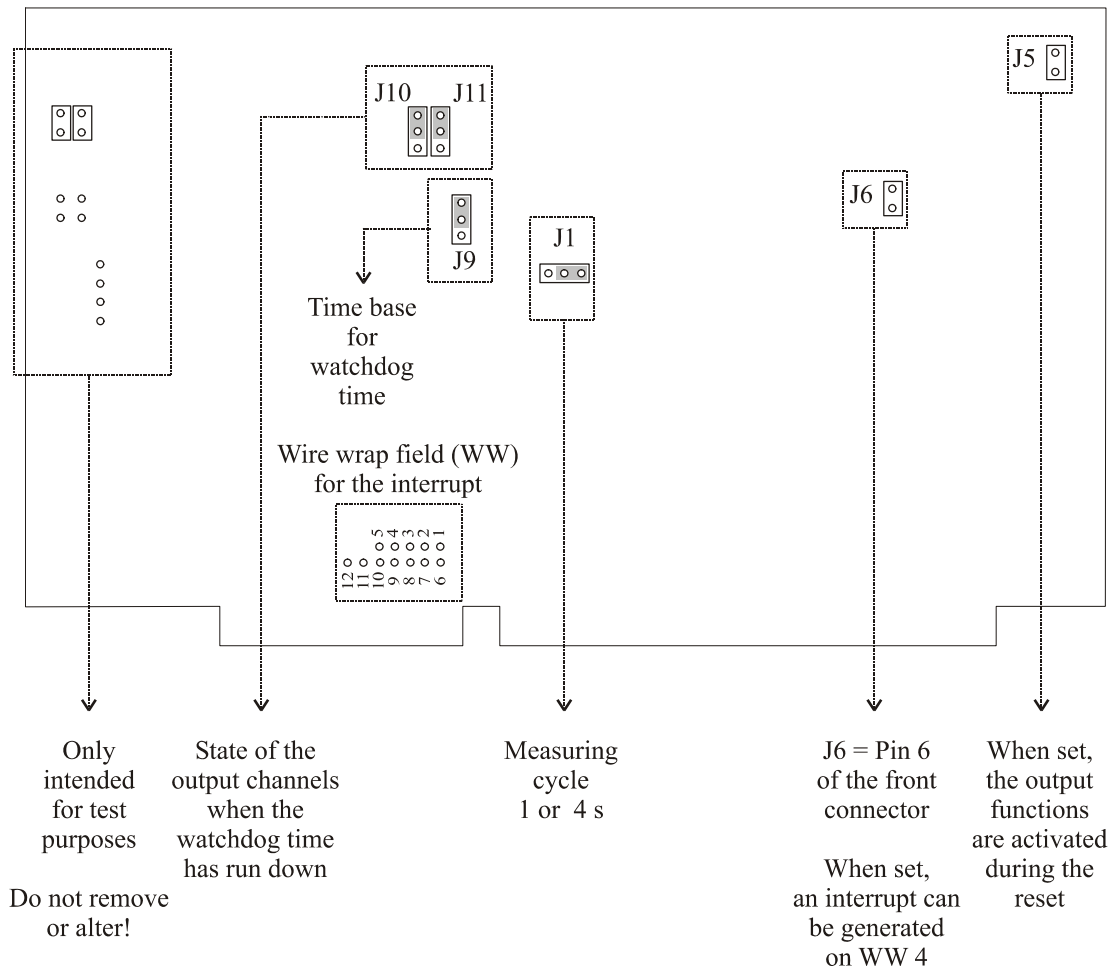




## 5.2 Jumper location and settings at delivery

### 5.2.1 Jumper location on the board

Fig. 5-2: Jumper location and settings at delivery



## 5.2.2 Jumper settings

Table 5-1: Jumper settings - Watchdog





Jumper	Settings	Functions	Settings at delivery
J1		Measuring cycle: 1 Hz → 1 s	✓
		Measuring cycle: 0,25 Hz → 4 s	
J9		Time base for the watchdog time: 1.953 kHz	✓
		Time base for the watchdog time: 7.812 kHz	

Table 5-2: Jumper settings - Output status after running down of the watchdog




Jumper	Settings	Functions	Settings at delivery
J10, J11	J10  J11	A pulse is generated after time has run. - relay changes to pin 1-3 (80 ms ± 10 ms) - transistor closed for 80 ms – VCE 0.4 V - TTL output is set to high for 80 ms	✓
	J10  J11	If timer function enabled through ENA bit: - relay switches back to position 1-2 - transistor is open - TTL output is set to low (0 V)	
	J10  J11	If watchdog function enabled through ENA bit: - relay switches to position 1-3 - transistor is closed- VCE 0.4 V - TTL output high (5 V)	

Table 5-3: Jumper settings - Interrupt wiring

	<b>Designation</b>	<b>WW pin assignment</b>
<b>Source of the interrupt on the board</b>	Timer	WW1
	EOC*	WW2
	Watchdog**	WW3
	Reset***	WW4
	UTEMP****	WW5
<b>End of the interrupt on the PC bus</b>	IRQ3	WW6 (PC,AT)
	IRQ4	WW7 (PC,AT)
	IRQ10	WW8 (AT)
	IRQ11	WW9 (AT)
	IRQ12	WW10 (AT)
	IRQ14	WW11 (AT)
	IRQ15	WW12 (AT)

\* This interrupt signal is cleared as soon as a new temperature measuring is started by software.

\*\* This signal is cleared after a new retrigger of the watchdog circuit.

\*\*\* This signal is a copy of the internal reset signal (J6) and only active as long as the reset signal remains.

\*\*\*\* This interrupt signal is set when the programmed limit temperature has been exceeded in the automatic mode. The signal is cleared with an output to the ADCMODE register

## 5.3 I/O mapping

The board requires an address range of 8 I/O address within the I/O space of the PC.

**Table 5-4: I/O mapping**

Base address	I/O functions	BYTE FORMAT								Description
		7	6	5	4	3	2	1	0	
Base+0	IORD	D7	D6	D5	D4	D3	D2	D1	LSB	ADCDATA
Base+0	IOWR	X	X	X	X	X	X	X	X	CONVSTART
Base+1	IORD	EOC	DOG	UTEMP	X	X	X	X	X	ADCSTATUS
Base+1	IOWR	X	X	X	X	X	X	X	MOD	ADCMODE
Base+2	IOWR	X	X	X	X	X	X	X	ENA	DOG
Base+3	IOWR	D7	D6	D5	D4	D3	D2	D1	D0	TEMPLIMIT
Base+4	IOWR IORD	D7	D6	D5	D4	D3	D2	D1	D0	TIMER0
Base+5	IOWR IORD	D7	D6	D5	D4	D3	D2	D1	D0	TIMER1
Base+6	IOWR IORD	D7	D6	D5	D4	D3	D2	D1	D0	TIMER2
Base+7	IOWR IORD	D7	D6	D5	D4	D3	D2	D1	D0	TIMERSTATUS

MOD = "0" ; Initiates the temperature measurement through software

MOD = "1" ; Initiates the temperature measurement automatically  
after the first software triggering

ENA = "0" ; Disables the external watchdog functions

ENA = "1" ; Enables the external watchdog functions

**Note:** When MOD = "1" ENA must be set to "1"

Base: base address

## 6 INSTALLATION

### **i**

#### **IMPORTANT!**

If you want to install simultaneously **several** ADDI-DATA boards, consider the following procedure.

- **Install and configure** the boards one after the other.  
You will thus avoid configuration errors.
1. Switch off the PC
  2. Install the **first** board
  3. Start the PC
  4. Install the software (once is enough)
  5. Configure the board
  6. Switch off the PC
  7. Install the **second** board
  8. Start the PC
  9. Configure the board
- etc

You will find additional information in the sections 6.1 to 6.5.

### **i**

#### **IMPORTANT!**

You have installed already **one or more** ADDI-DATA boards in your PC, and you wish to install **an additional** board?

Proceed as if you wished to install one single board.

## 6.1 Setting the base address through DIP switches

The PA 030 requires 8 I/O addresses.

You can choose the position of this address block within the available I/O address space in intervals of 8 bytes.



### WARNING!

If the base address set is wrong, the board and/or the PC may be damaged

#### *Before installing the board*

At delivery, the base address is set on the address 0390H.

#### ### Check, that

- the base address is free
- the address range required by the board is not already used by the PC or by boards already installed in the PC.

If the base address or the address range **are wrong**,

- **select** another base address with the 8-pin block of DIP switches S1.

#### *Decoding the base address*

The base address is decoded in steps of 8 I/O addresses.

It can be selected between 0 and 07FFH within the PC I/O address space.

In table 6-1 the address 0390H is decoded (settings at delivery).

**Table 6-1: Decoding table**

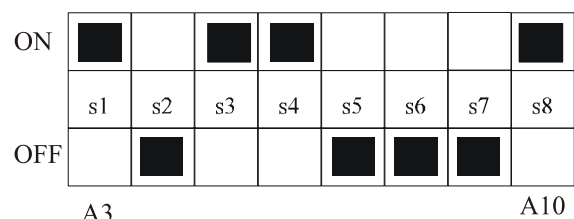
	MSB												LSB			
Decoded address bus	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
Wished base address Hex	0				3				9				0			
Wished base address binary	0	0	0	0	0	0	1	1	1	0	0	1	0	0	0	0
DIP switch S1 Logic "0"= ON Logic "1" = OFF	*	*	*	*	*	s8 ON	s7 OFF	s6 OFF	s5 OFF	s4 ON	s3 ON	s2 OFF	s1 ON	X	X	X

X: decoded address range of the board  
\* : permanently decoded on logic "0"

**Fig. 6-1: DIP switches S1**

### IMPORTANT!

You will find the switch **s1 on the left** of the block of DIP switches!



## 6.2 Inserting the board

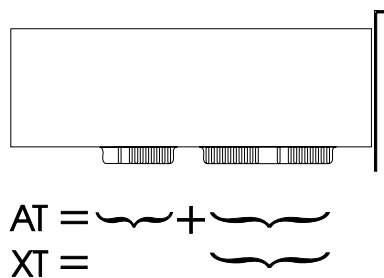
**i****IMPORTANT!**Do observe the *safety instructions*.

### 6.2.1 Opening the PC

- Switch off your PC and all the units connected to the PC.
- Pull the PC mains plug from the socket.
- Open your PC as described in the manual of the PC manufacturer.

#### 1. Select a free ISA slot

Fig. 6-2: types of slots



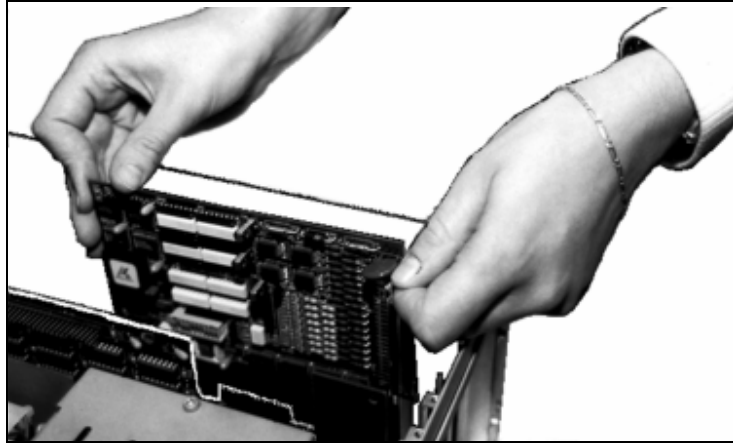
The board can be inserted either in a slot XT or AT.  
It can also be inserted in EISA slots.

2. **Remove the back cover of the selected slot** according to the instructions of the PC manufacturer.  
Keep the back cover. You will need it if you remove the board.
3. **Discharge yourself from electrostatic charges**
4. **Take the board from its protective pack**

## 6.2.2 Plugging the board into the slot

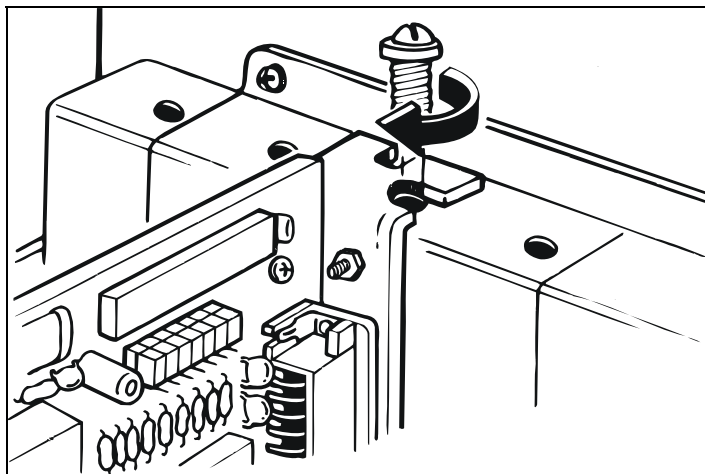
- Discharge yourself from electrostatic charges
- Insert the board vertically into the chosen slot.

Fig. 6-4: Inserting the board



- Fasten the board to the rear of the PC housing with the screw which was fixed on the back cover.

Fig. 6-5: Securing the board at the back cover



- Tighten all loosen screws.

## 6.2.3 Closing the PC

- Close your PC as described in the manual of the PC manufacturer.



## 6.3 Installing the software

The board is delivered with a CD-ROM containing ADDIREG for Windows NT 4.0 and Windows 95/98.

You can download the latest version of the ADDIREG program from the Internet:

<http://www.addi-data.com>

**i**

### **IMPORTANT!**

Before using the board or in case of malfunction during operation, check if there is an update of the product (technical description, driver). The current version can be found on the internet or contact us directly.

The CD also contains standard software for the ADDI-DATA boards:

- 16-bit for MS-DOS and Windows 3.11
- 32-bit for Windows NT/95/98.

### 6.3.1 Software installation under MS-DOS and Windows 3.11

- Copy the contents of PA030\16bit on a disk.  
If several disks are to be used, the directory contents is stored in several sub-directories (Disk1, Disk2, Disk3...).
- Insert the (first) disk into a drive and change to this drive.
- Enter <INSTALL>.

The installation program gives you further instructions.

### 6.3.2 Software installation under Windows NT/95/98

- Select the directory PA030\32bit\Disk1.
- Start the set-up program "setup.exe" (double click)
- Select one of the 3 parameters
  - 1- typical
  - 2- compact
  - 3- custom

Proceed as indicated on the screen and read the "Software License" and "Readme". Under "custom", you can select your operating system.

The installation program gives you further instructions.

## 6.4 Board configuration with ADDIREG

The ADDIREG registration program is a 32-bit program for Windows NT/ 95/98. The user can register all hardware information necessary to operate the ADDI-DATA PC boards.

**i**

**IMPORTANT!**

If you use one or several resources of the board, you cannot start the ADDIREG program.

### 6.4.1 Program description

**i**

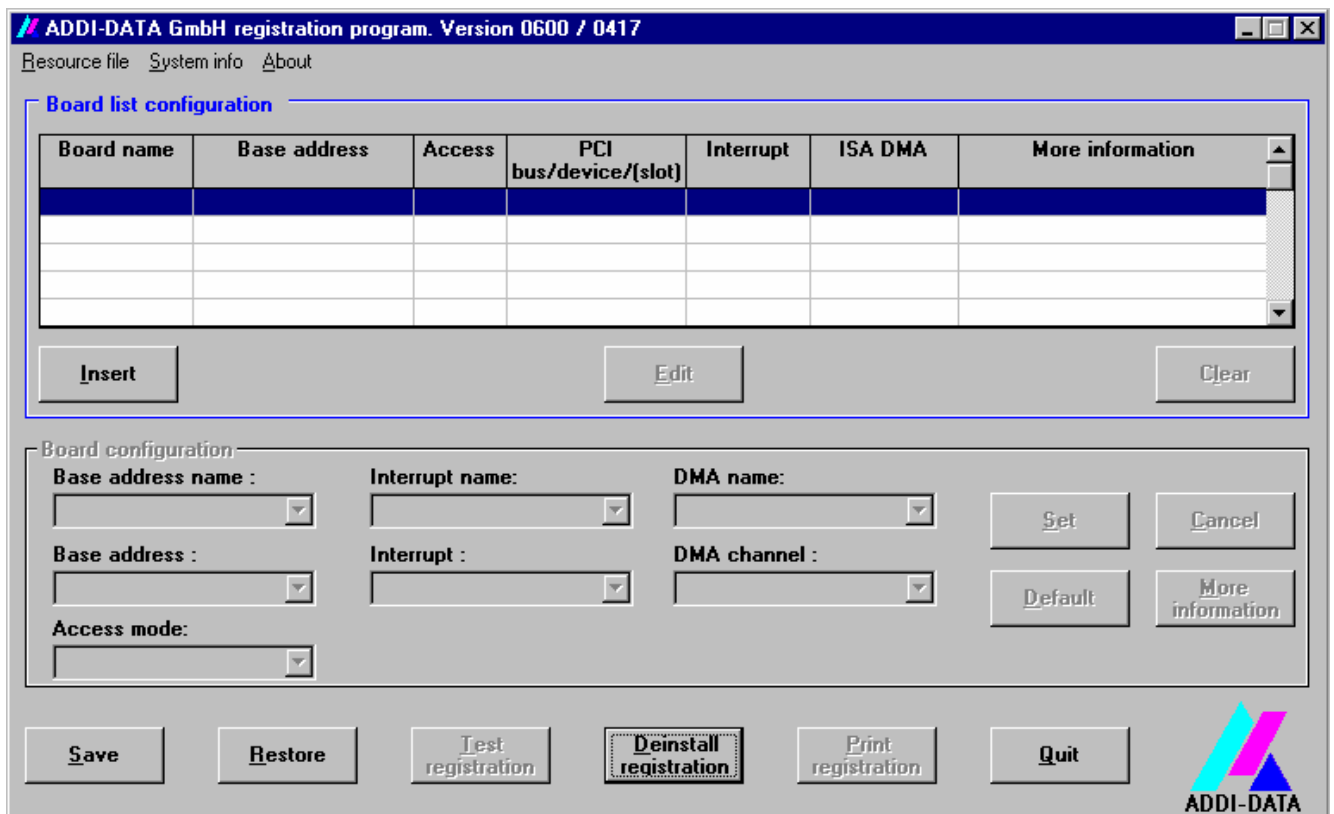
**IMPORTANT!**

Insert the ADDI-DATA boards to be registered before starting the ADDIREG program.

If the board is not inserted, the user cannot test the registration.

Once the program is called up, the following dialog box appears.

Fig. 6-6: The ADDIREG registration program



### Board list configuration

The table in the middle lists the registered boards and their respective parameters.

**Board name:**

Names of the different registered boards

When you start the program for the first time, no board is registered in this table.

**Base address:**

Selected base address of the board.

**i****IMPORTANT!**

The base address selected with the ADDIREG program must correspond to the one set through DIP-switches.

**Access:**

Selection of the access mode for the ADDI-DATA digital boards.  
Access in 8-bit or 16-bit.

**PCI bus/device/(slot):**

Used PCI slot. If the board is no PCI board, the message "NO" is displayed.

**Interrupt:**

Used interrupt of the board. If the board uses no interrupt, the message "Not available" is displayed.

**i****IMPORTANT!**

The interrupt selected with the ADDIREG program must correspond to the one set through DIP-switches.

**ISA DMA:**

Indicates the selected DMA channel or "Not available" if the board uses no DMA.

**More information:**

Additional information like the identifier string (e.g.: PCI1500-50) or the installed COM interfaces.

**Text boxes:**

Under the table you will find 6 text boxes in which you can change the parameters of the board.

**Base address name:**

When the board operates with several base addresses (One for port 1, one for port 2, etc.) you can select which base address is to be changed.

**Base address:**

In this box you can select the base addresses of your PC board. The free base addresses are listed. The used base addresses do not appear in this box.

**Interrupt name:**

When the board must support different interrupt lines (common or single interrupts), you can select them in this box.

**Interrupt:**

Selection of the interrupt number which the board uses.

**DMA name:**

When the board supports 2 DMA channels, you can select which DMA channel is to be changed.

**DMA channel:**

Selection of the used DMA channel.

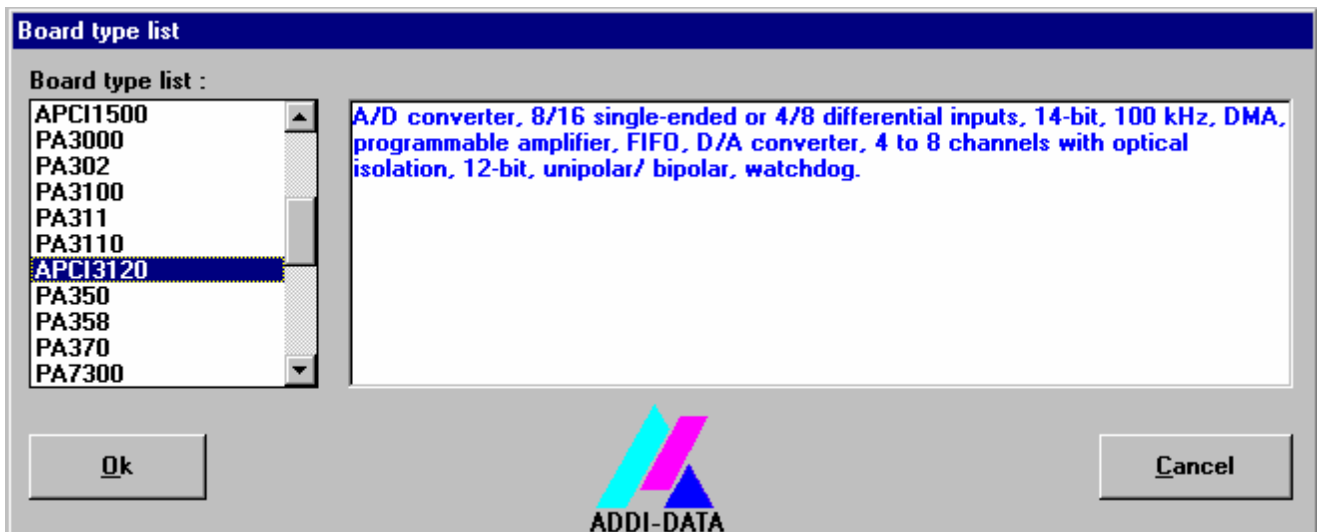
**Buttons:****Edit 1:**

Selection of the highlighted board with the different parameters set in the text boxes. Click on "Edit" to activate the data or click twice on the selected board.

**Insert:**

When you want to insert a new board, click on "Insert". The following dialog window appears:

Fig. 6-7: Configuring a new board



All boards you can register are listed on the left. Select the wished board. (The corresponding line is highlighted).

On the right you can read technical information about the board(s).

Activate with "OK"; You come back to the former screen.

**Clear:**

You can delete the registration of a board. Select the board to be deleted and click on "Clear".

<sup>1</sup> "x": Keyboard shortcuts; e.g. "Alt + e" for Edit

**Set:**

Sets the parameterised board configuration. The configuration should be set before you save it.

**Cancel:**

Reactivates the former parameters of the saved configuration.

**Default:**

Sets the standard parameters of the board.

**More information:**

You can change the board specific parameters like the identifier string, the COM number, the operating mode of a communication board, etc...

If your board does not support this information, you cannot activate this button.

**Save:**

Saves the parameters and registers the board.

**Restore:**

Reactivates the last saved parameters and registration.

**Test registration:**

Controls if there is a conflict between the board and other devices.

A message indicates the parameter which has generated the conflict. If there is no conflict, "OK" is displayed.

**Deinstall registration:**

Deinstalls the registration of all boards listed in the table.

**Print registration:**

Prints the registration parameter on your standard printer.

**Quit:**

Quits the ADDIREG program.

## 6.4.2 Registering a new board

### **i**

**IMPORTANT!**

To register a new board, you must have administrator rights. Only an administrator is allowed to register a new board or change a registration.

- Call up the ADDIREG program. The figure 6-6 is displayed on the screen. Click on "Insert". Select the wished board.
- Click on "OK". The default address, interrupt, and the other parameters are automatically set in the lower fields. The parameters are listed in the lower fields. If the parameters are not automatically set by the BIOS, you can change them. Click on the wished scroll function(s) and choose a new value. Activate your selection with a click.

- Once the wished configuration is set, click on "Set".
- Save the configuration with "Save".
- You can test if the registration is "OK".  
This test controls if the registration is right and if the board is present.  
If the test has been successfully completed you can quit the ADDIREG program.  
The board is initialised with the set parameters and can now be operated.

In case the registration data is to be modified, it is necessary to boot your PC again. A message asks you to do so. When it is not necessary you can quit the ADDIREG program and directly begin with your application.

### 6.4.3 Changing the registration of a board

#### **i**

#### **IMPORTANT!**

To change the registration of a board, you must have administrator rights. Only an administrator is allowed to register a new board or change a registration.

- Call up the ADDIREG program. Select the board to be changed.  
The board parameters (Base address, DMA channel, ..) are listed in the lower fields.
- Click on the parameter(s) you want to set and open the scroll function(s).
- Select a new value. Activate it with a click.  
Repeat the operation for each parameter to be modified.
- Once the wished configuration is set, click on "Set".
- Save the configuration with "Save".
- You can test if the registration is "OK".  
This test controls if the registration is right and if the board is present.  
If the test has been successfully completed you can quit the ADDIREG program.  
The board is initialised with the set parameters and can now be operated.

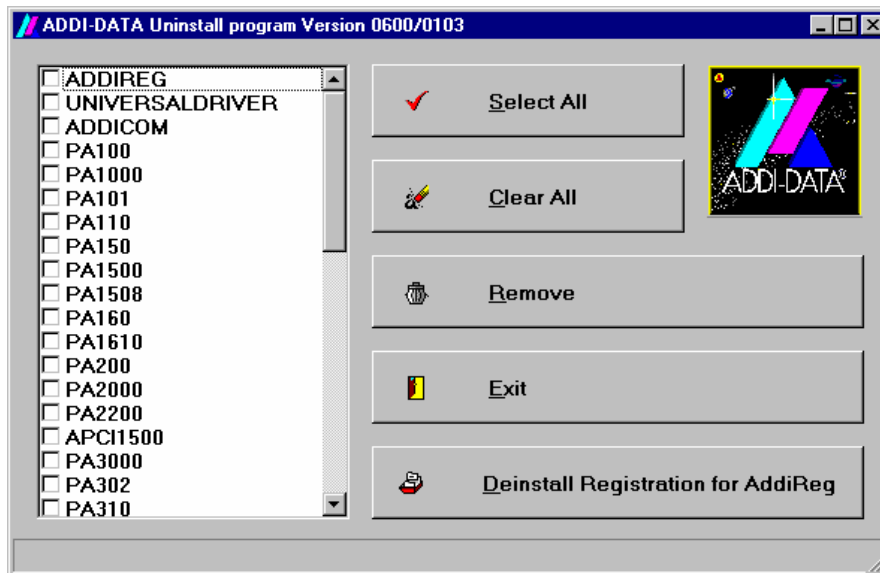
In case the registration data is to be modified, it is necessary to boot your PC again. A message asks you to do so. When it is not necessary you can quit the ADDIREG program and directly begin with your application.

### 6.4.4 Removing the ADDIREG program

The ADDI\_UNINSTALL program is delivered on the CD-ROM.

- Start the ADDI\_UNINSTALL program

Fig. 6-8: The ADDI\_UNINSTALL program



- Start the ADDIREG program and click on "Deinstall registration for AddiReg"
- Proceed as indicated until the complete removing of ADDIREG.

You can also download the program from Internet.

## 6.5 Software downloads from the Internet

You can download the latest version of the device driver for the **PA 030** board.

<http://www.addi-data.com>

**i**

### IMPORTANT!

Before using the board or in case of malfunction during operation, check if there is an update of the product (technical description, driver). The current version can be found on the internet or contact us directly.

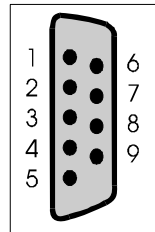
If you have any questions, do not hesitate to send us an e-mail to

info@addi-data.de            or  
hotline@addi-data.com

## 7 CONNECTION TO THE PERIPHERAL

### 7.1 Connector pin assignment

Fig. 7-1: 9-pin SUB-D female connector



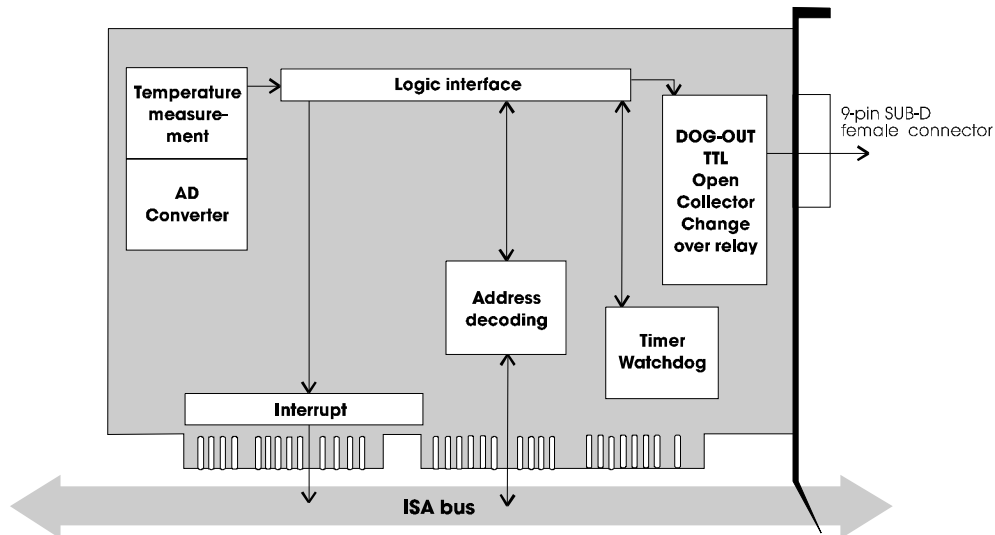
- |   |   |
|---|---|
| 1 | Relay change-over contact                       |
| 2 | Relay opening contact                           |
| 3 | Relay closing contact                           |
| 4 | Buffered TTL signal                             |
| 5 | 5 V through intern. pull up resistor of 560R    |
| 6 | Extern. reset input (no system reset)           |
| 7 | Open Collector with protection diode (time out) |
| 8 | Ground  |
| 9 | Buffered TTL, timer output (timer 1)            |



## 8 FUNCTIONS

### 8.1 Introduction

Fig. 8-1: Block diagram



The board is to be initialised by software for the timer and watchdog function.

All external watchdog signals are blocked as long as the watchdog function is not released by software.

Temperature is measured over an on-board temperature sensor. An 8-bit A/D converter is switched behind the sensor. Temperature measuring can be started directly through software and requested at the end of conversion via a flag. Automatic conversion can also be cyclically through timer. The trigger time is set to 1 or 4 s. (See jumper configuration).

The temperature is compared on the board with the programmed limit temperature. When the limit temperature is reached, a "UTEMP" signal is released as an interrupt on WW5. The signal can be requested through the status bit "UTEMP".

The software can supervise the A/D conversion, a temperature exceeding the programmed limit temperature, the watchdog time.

After timeout of the watchdog, 3 signals are available on the front connector:

- Buffered TTL signal
- Open-collector with protection diode
- potentially free relay contact (opening and closing contact).

The board reacts to I/O Read and I/O Write commands. The board I/O address decoding allows to command the board within the 64KB I/O address space. The board itself occupies 8 bytes within the I/O address space.

The I/O address range is fully decoded over the 64KB I/O address space. The board can be adjusted in the address range 0 - 07FFH. The base address is set via a 8-pin block of DIP switches in steps of 8 bytes.

When the board is responding, no wait state is produced.

## 8.2 Programming and operation

### 8.2.1 Temperature

Temperature measuring occurs through sensor LM335 and the 8-bit A/D converter ADC0804.

The temperature measuring range is calibrated on 0-127°C with a resolution of 0.5°C per digit.

Temperature measuring is started by a simple I/O output command.

EOC can be read over a status byte.

An interrupt can be triggered in the same way with the EOC signal (WW2).

By using the automatic conversion mode, the board automatically generates a signal when the programmed limit temperature has been exceeded. The identifying signal can be evaluated with the status bit ("UTEMP") or as an interrupt (WW5).

The limit value is set via software. The trigger time is adjusted through jumper J1.

The software programmable 32-bit timer is fitted with a 82C54 timer module. Timer 0 und timer 1 are used.

Input frequency = 2MHz.

The output of timer 0 is connected with the input of timer 1.

To set a frequency you have to program always timer 0 and 1.

The watchdog timer 2 can be retriggered by hardware over the external Gate input. But timer 2 has to be programmed in the corresponding mode.

The input frequency of timer 2 is adjusted over jumper J9. With a frequency of 7.812 kHz, the watchdog time is at least 0.12 ms and at most 8.39 s. With an input frequency of 1.953 kHz the watchdog time is at least 0.5 ms and at most 33.55 s.

After the programmed time, timer 2 generates a pulse, if no retrigger pulse has been produced within the programmed time on a defined I/O address. The timer has to be retriggered cyclically over this address.

If the time runs down without retrigger, the output pulse of timer 2 is prolonged for about 80 ms or it effects a level change at the output, according to the jumper settings of J10 and J11.

This signal is available in different types for further processing internally and externally.

The board **PA 030** is built to operate with a supply voltage of +5V and +12V.

## Initiating the temperature measuring

A conversion of the A/D converter is started by a dummy writing on the address CONVSTART. The identification flag EOC is simultaneously reset. When the conversion is completed, the EOC flag is set. It can be read on bit 7 of address ADCSTATUS. "0" corresponds to end of conversion. It can also be used as an interrupt over WW2 (WW2=EOC).

## Read temperature (ADCDATA)

LSB - D7: After the end of conversion the data bits D0-D7 of the converted temperature value are retained in LSB-D7. The scale interval is 0.5°C per digit

## Read status (ADCSTATUS)

EOC bit: "1": conversion is under way.  
"0": conversion is completed.

DOG bit: "1": watchdog has not run  
"0": watchdog has run (timeout), no other triggering

UTEMP Bit: "1": temperature not exceeding  
"0": temperature exceeding

## Automatic temperature measuring (ADCMODE)

The maximum temperature is to be determined by software. The limit value is written in the TEMPLIMIT register (base address +3) as an 8-bit value. Each bit has a value of 0.5°C.

The value "0" corresponds to a temperature of 0°C. (128 = 64°C, 255 = 127°C).

By writing "1" on MOD of the ADCMODE register, the automatic conversion mode is selected. The automatic conversion is started for the first time by a dummy writing in the CONVSTART register. The measurements are then automatically carried out every 1sec (or 4s. according to the configuration of jumper J1).

If "0" is written on MOD of the ADCMODE register, temperature measuring is set to a conversion controlled by software.

If the programmed limit temperature is exceeded an interrupt signal is generated. The interrupt signal is reset when the ADCMODE register is written or when the ADCSTATUS register is read.

## 8.2.2 Triggering the timer DOG

DOG: When ENA=1 or ENA=0 (according to the initialisation) is written on the address DOG, the timer is reset. But timer 2 has to be previously initialised.

### Timer programming

TIMER 0 - 2,      The individual divider factors are programmed over the  
TIMERSTATUS      addresses "timer 0 - 2".  
:                    The address TimerStatus is used for the mode selection.  
                      The timer signal is available as an interrupt request signal  
                      on pin WW1 or on pin 9 of the 9-pin female connector.

## 8.3 Timer and watchdog

The board **PA 030** is provided with a quartz oscillator. The frequency is 2 MHz. This frequency is supplied to the first timer as an input frequency. The frequency is reduced to 7.812 kHz and 1.953 kHz via other dividers. These two frequencies are optionally available over jumper J9 as input frequencies of the watchdog timer.

### 8.3.1 Timer

The 82C54 timer is equipped with 3 freely programmable 16-bit timers. 2 timers can be programmed to produce a time signal.

Timer 0 and timer 1 are therefore cascaded. The input frequency of timer 0 is 2 MHz. This clock signal is produced by a separated quartz-stabilised oscillator. The output signal of timer 0 is simultaneously the input signal of timer 1. The output of timer 1 is connected to a driver.

The output signal of timer 1 is available:

- as a TTL interrupt request signal on pin WW1
- as a buffered TTL signal on pin 9 of the external female connector.

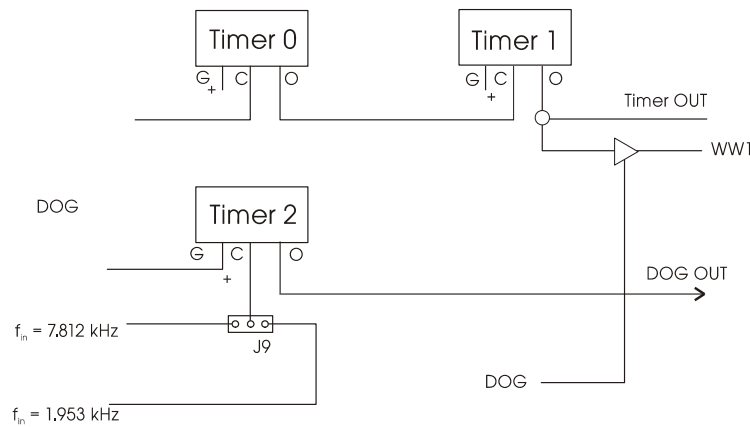
Timer 2 functions as a watchdog timer.

The 3 timers can be independently programmed from the others.

The gate inputs gate 0 - 1 are supplied with +5V.

The hardware retrigger of the watchdog timer is carried out through gate 2.

Fig. 8-2: Wiring principle of the timers



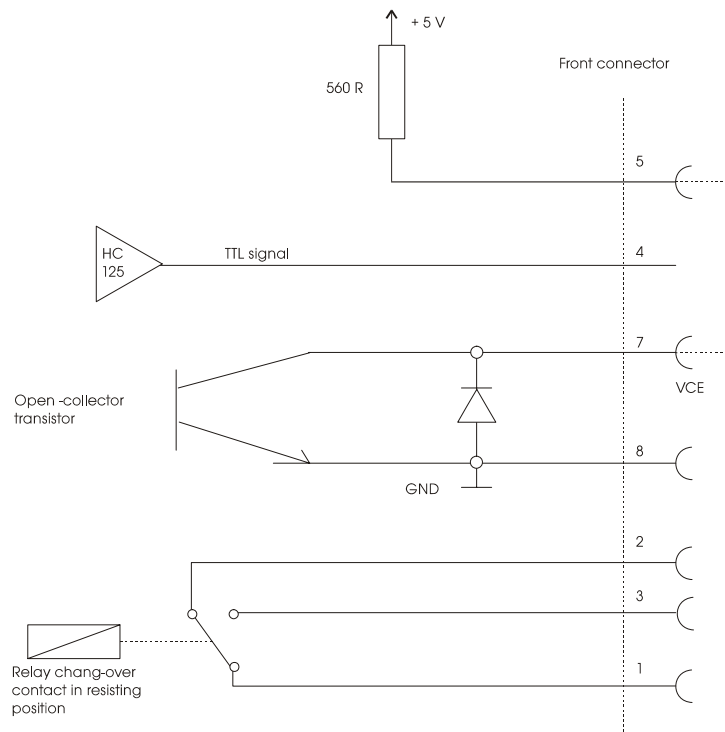
### 8.3.2 Watchdog timer

The watchdog timer can either be used internally or externally from the PC.

**Internally:** the watchdog status can be requested through status bit. An interrupt can be generated. But it must be previously enabled over the ENA bit.

**Externally:** the watchdog is used as a buffered TTL signal, as an open collector transistor, as a potentially free relay change-over contact.

Fig. 8-3: Output wiring for watchdog signals



For the status of the outputs after running down of the watchdog, refer to the jumper settings of J10 and J11 in chapter 5.

Module 82C54 requires an initialisation by software to select the corresponding function. It can be programmed immediately after applying the operating voltage.

### 8.3.3 Possibility of interrupts

The timer function allows to generate interrupts. Therefore is the buffered output signal of timer 1 on pin WW1 available.

To produce an interrupt signal, timer 0 and timer 1 are to be programmed.

The release of interrupt, TTL and watchdog signals at the front connector occurs after DOG trigger.

DOG status:        ENA = 0 ; disable  
                  ENA = 1 ; enable.

This bit has to be updated at each trigger of DOG.

### 8.3.4 Reset

Jumper J6 or equivalent pin 6 of the external SUB-D connector can also be used as an external reset input. If this position is set, an interrupt can be generated on WW4.

When jumper J5 is adjusted, the external watchdog function O.C.<sup>2</sup> and the relay function are activated during reset.

---

<sup>2</sup> O.C.: Opening contact

### 8.3.5 Timer programming

The individual timer functions are responded over the following I/O addresses:

Base+04H ;timer 0, data address

Base+05H ;timer 1, data address

Base+07H ;timer status address

Base: selected base address

The requested timer modes are programmed over status address Base+07H.

**Table 8-1: Timer modes**

TIMER STATUS

SC1	SC0	RL1	RL0	M2	M1	M0	BCD																																																										
<div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>0: Binary counter (16-bit) 1: BCD counter (4 decades)</p> <p>Counter mode</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>M2</th> <th>M1</th> <th>M0</th> <th>Mode</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>Mode 0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>Mode 1</td></tr> <tr><td>x</td><td>1</td><td>0</td><td>Mode 2</td></tr> <tr><td>x</td><td>1</td><td>1</td><td>Mode 3</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>Mode 4</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>Mode 5</td></tr> </tbody> </table> <p>RW - Read/Write</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>RW1</th> <th>RW0</th> <th>Mode</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>Counter latch command</td></tr> <tr><td>0</td><td>1</td><td>R/W low byte</td></tr> <tr><td>1</td><td>0</td><td>R/W high byte</td></tr> <tr><td>1</td><td>1</td><td>R/W both bytes first; LSB, then MSB</td></tr> </tbody> </table> <p>SC - Select counter</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th>SC1</th> <th>SC0</th> <th>Mode</th> </tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>Select counter 0</td></tr> <tr><td>0</td><td>1</td><td>Select counter 1</td></tr> <tr><td>1</td><td>0</td><td>Select counter 2</td></tr> <tr><td>1</td><td>1</td><td>Read back Command</td></tr> </tbody> </table> </div> <div style="width: 50%; font-size: small;"> <p>Diagram description: The table above shows bit fields SC1, SC0, RL1, RL0, M2, M1, M0, and BCD. Lines connect these fields to their respective descriptions: BCD to counter type, M2-M0 to counter mode, RL1-RL0 to read/write commands, and SC1-SC0 to counter selection.</p> </div> </div>								M2	M1	M0	Mode	0	0	0	Mode 0	0	0	1	Mode 1	x	1	0	Mode 2	x	1	1	Mode 3	1	0	0	Mode 4	1	0	1	Mode 5	RW1	RW0	Mode	0	0	Counter latch command	0	1	R/W low byte	1	0	R/W high byte	1	1	R/W both bytes first; LSB, then MSB	SC1	SC0	Mode	0	0	Select counter 0	0	1	Select counter 1	1	0	Select counter 2	1	1	Read back Command
M2	M1	M0	Mode																																																														
0	0	0	Mode 0																																																														
0	0	1	Mode 1																																																														
x	1	0	Mode 2																																																														
x	1	1	Mode 3																																																														
1	0	0	Mode 4																																																														
1	0	1	Mode 5																																																														
RW1	RW0	Mode																																																															
0	0	Counter latch command																																																															
0	1	R/W low byte																																																															
1	0	R/W high byte																																																															
1	1	R/W both bytes first; LSB, then MSB																																																															
SC1	SC0	Mode																																																															
0	0	Select counter 0																																																															
0	1	Select counter 1																																																															
1	0	Select counter 2																																																															
1	1	Read back Command																																																															

The divider factors of the individual timers are programmed via data addresses. They are divided in 2 bytes (low byte and high byte).

In a data address is first written the LOW and then the HIGH byte.

## Programming examples

**Exercise:** An interrupt signal of 100Hz is required on pin WW1.

Calculation of the divider factor:

$$T = f_{in} / f_{out} \quad \rightarrow \quad T = 2\text{MHz} / 100\text{Hz} = 20000$$

This divider factor is the product of the divider factors of timer 0 and timer 1.

This factor is divided in 2 individual divider factors.

ex. Factor 0 = 500

Factor 1 = 40

The product of both factors must be 20000 in this example. The individual divider factors can also have other values (4000/5, 1000/20, 400/50, etc.), where the product is always 20000, in this example.

In the following programs you will find solution propositions in ASM86 and BASIC

### Solution 1: ASM86

```

MOV DX,0397H          ; Timer status address (Base = 0390H)
                      ;
MOV AL,76H           ; Mode Word timer 1
OUT DX,AL            ; Program mode
*                   ;
MOV AL,36H           ; Mode Word timer 0
OUT DX,AL            ; Program mode
*                   ;
MOV AX,500           ;
MOV DX,0395H         ; Timer 1 data address
OUT DX,AL            ; Program divider factor (Low byte)
*                   ;
MOV AL,AH            ; Program divider factor (High byte)
OUT DX,AL            ;
*                   ;
MOV DX,0394H         ; Timer 0 data address
MOV AX,40             ; Divider factor (High + Low byte) in AX
OUT DX,AL            ; Program divider factor (Low byte)
*                   ;
MOV AL,AH            ; Program High byte after register A
OUT DX,AL            ;
*                   ;
MOV DX,0392H         ; Enable buffer through the address DOG
MOV AL,01             ;
OUT DX,AL            ;
                      ;
                      ; Timer 1 and timer 0 are programmed.
                      ; The 100Hz signal is available on pin WW1

```

\* See page 32



**Solution 2: BASIC**

```

OUT &H397,&H76      ; Program mode Timer 1
*                   ;
OUT &H397,&H36      ; Program mode timer 0
*                   ;
OUT &H395,L_byte1   ; Divider factor timer 1 (Low byte)
*                   ;
OUT &H395,H_byte1   ; Divider factor timer 1 (High byte)
*                   ;
OUT &H394,L_byte0   ; Divider factor timer 0 (Low byte)
*                   ;
OUT &H394,H_byte0   ; Divider factor timer 0 (High byte)
*                   ;
OUT &H392,&H01      ; Enable buffer through the address DOG
*                   ;
                   ; Timer 1 and timer 0 are programmed.
                   ; The 100Hz signal is available on
                   ; pin WW1.

```

**8.3.6 Watchdog programming**

The watchdog timer functions are responded over the following I/O addresses

Base+06H ;Timer data address

Base+07H ;Timer status address

Mode 1 is programmed over status address Base+07H with the status value, ex. 0B2H (one-shot).

**Programming examples**

**Exercise:** The watchdog timer has to be set on the time of 1sec.

Calculation of the divider factor:

$$T = f_{in}/f_{out} \quad \text{--->} \quad T = 7.812\text{KHz} / 1\text{Hz} = 7812 = 1\text{E84H}$$

This divider factor has been programmed with two outputs on the timer data address. The following programs show solution propositions in ASM86 and BASIC.

**Solution 1: ASM86**

```

MOV DX, 0397H      ; Timer status address at Base = 0390H
                   ;
MOV AL, 0B2H       ; Mode 1 for watchdog timer (one-shot)
OUT DX, AL         ; Program mode
*                 ;
MOV AX, 7812       ; Divider factor in AX (1E84H)
MOV DX, 0396H     ; Timer 2, data address
OUT DX, AL        ; Program LOW byte of
*                 ; divider factor
MOV AL,AH         ; Program HIGH byte of
OUT DX, AL        ; divider factor
                   ; The watchdog timer is now programmed
                   ; on the time of 1sec. The output signal is
                   ; produced after 1 sec, unless having been
                   ; triggered before over the DOG address
                   ;
MOV DX, 0392H     ; DOG retrigger address
MOV AL, 1         ; ENA DOG
OUT DX, AL        ; Retrigger watchdog
                   ; DOG time is again 1 sec

```

**Solution 2: BASIC**

```

OUT &H397, &H0B2   ; Program mode of timer 2
*                 ;
OUT &H396, &H084   ; Divider factor of timer 2, L-byte
*                 ;
OUT &H396, &H01E   ; Divider factor of timer 2, H-byte
*                 ;
                   ; timer 2 is now programmed as
                   ; watchdog timer on 1 sec
OUT &H392, 1       ; Enable Dog and retrigger

```

(\*) If your computer is used with a microprocessor frequency  $\geq 12\text{MHz}$ , make sure that there is a delay between the different outputs. Otherwise a correct programming of the timers can not be guaranteed (ex. 2 - 4 x NOP).

## 8.4 Interrupt

The board can generate interrupts through timer frequency, EOC (end of conversion) and watchdog. These signals occupy WW pins 1 - 3. They can be switched to one of the PC's 2 interrupt bus lines (resp. to 7 interrupt bus lines of the AT).

The lines IRQ3, IRQ4 are for PC interrupts and the lines IRQ10 - IRQ15 are for AT interrupts. The interrupt functions timer and EOC are only enabled after the first retrigger of the watchdog timer.

## 9 STANDARD SOFTWARE

### 9.1 Introduction

#### **i**

#### **IMPORTANT!**

Note the following conventions in the text:

Function: "i\_PA030\_SetBoardInformation"

Variable *ui\_Address*

**Table 9-1: Type Declaration**

	<b>Borland C</b>	<b>Microsoft C</b>	<b>Borland Pascal</b>	<b>Microsoft Visual Basic Dos</b>	<b>Microsoft Visual Basic Windows</b>
<b>VOID</b>	void	void	pointer		any
<b>BYTE</b>	unsigned char	unsigned char	byte	integer	integer
<b>INT</b>	int	int	integer	integer	integer
<b>UINT</b>	unsigned int	unsigned int	word	long	long
<b>LONG</b>	long	long	longint	long	long
<b>PBYTE</b>	unsigned char *	unsigned char *	var byte	integer	integer
<b>PINT</b>	int *	int *	var integer	integer	integer
<b>PUINT</b>	unsigned int *	unsigned int *	var word	long	long
<b>PCHAR</b>	char *	char *	var string	string	string

## 9.2 Software functions (API)

### 9.2.1 Initialisation

#### 1) i\_PA030\_SetBoardInformation

**Syntax:**

```
<Return value> = i_PA030_SetBoardInformation
                                     (UINT   ui_Address,
                                     PBYTE  pb_BoardHandle)
```

**Parameter:**

UINT	ui_Address	Base address of the <b>PA 030</b>
PBYTE	pb_BoardHandle	Handle <sup>1)</sup> of the board to use the functions

**Task:**

Verifies if the board **PA 030** is present and stores the base address. A handle is returned to use the next functions. Handles allow to operate several boards.

**Return value:**

0 : No error  
 -1 : Board not present or address already occupied.  
 -2 : No handle available for the boards (up to 10 handle can be used)  
 -3: Error when opening the driver under Windows NT/95/98

#### 2) i\_PA030\_CloseBoardHandle (...)



**IMPORTANT!**

Call up this function each time you want to quit the user program!

**Syntax:**

```
<Return value> = i_PA030_CloseBoardHandle (BYTE  b_BoardHandle)
```

**Parameter:**

BYTE	b_BoardHandle	Handle of the PA 030
------	---------------	----------------------

**Task:**

Releases the handle of the board and blocks the access to the board.

**Return value:**

0: No error  
 -1: The handle parameter of the board is wrong.

---

<sup>1</sup> Identification number of the board

## 9.2.2 Timer/watchdog functions

### 1) i\_PA030\_ReadTemp

**Syntax:**

<Return value> = i\_PA030\_ReadTemp (BYTE b\_BoardHandle,  
PDOUBLE pd\_Temperature)

**Parameter:**

BYTE	b_BoardHandle	Handle of the <b>PA 030</b>
PDOUBLE	pd_Temperature	Measured temperature

**Task:**

Reads the temperature on the board and returns it as a real value.  
When "-1" is returned, the automatic mode is active.

### 2) i\_PA030\_ReadAdcState

**Syntax:**

<Return value> = i\_PA030\_ReadAdcState (BYTE b\_BoardHandle)

**Parameter:**

BYTE	b_BoardHandle	Handle of the <b>PA 030</b>
------	---------------	-----------------------------

**Task:**

Displays the ADC current state.  
The return value is a 16-bit integer.

### 3) i\_PA030\_AutoMode

**Syntax:**

<Return value> = i\_PA030\_AutoMode (BYTE b\_BoardHandle,  
INT i\_Temperature)

**Parameter:**

BYTE	b_BoardHandle	Handle of the PA 030
INT	i_Temperature	Temperature range

**Task:**

Sets the board to automatic temperature measuring.

**Return value:**

0: No error.

-1: If the delivered temperature is not comprised between 0 and 127 °C.

-2: If the ENA bit is not enabled

#### 4) i\_PA030\_ManuMode

**Syntax:**

<Return value> = i\_PA030\_ManuMode (BYTE b\_BoardHandle)

**Parameter:**

BYTE                b\_BoardHandle                Handle of the **PA 030**

**Task:**

Switches the automatic temperature measuring off.  
No input and output parameters.

#### 5) i\_PA030\_SetDogPeriod

**Syntax:**

<Return value> = i\_PA030\_SetDogPeriod (BYTE     b\_BoardHandle,  
  UINT     ui\_TeilerFaktor)

**Parameter:**

BYTE                b\_BoardHandle                Handle of the **PA 030**  
UINT                ui\_TeilerFaktor                Divider factor

**Task:**

Sets the watchdog divider factor.  
No return value.

#### 6) i\_PA030\_RetriggerDog

**Syntax:**

<Return value> = i\_PA030\_RetriggerDog        (BYTE b\_BoardHandle)

**Parameter:**

BYTE                b\_BoardHandle                Handle of the **PA 030**

**Task:**

Allows to retrigger the watchdog.

#### 7) i\_PA030\_ReadDogState

**Syntax:**

<Return value> = i\_PA030\_ReadDogState        (BYTE b\_BoardHandle)

**Parameter:**

BYTE                b\_BoardHandle                Handle of the **PA 030**

**Task:**

Reads the current watchdog state.  
Return value is "0", when the watchdog time has run.  
Return value is "1", when the watchdog time has not already run.

**8) i\_PA030\_SetTimer0Periode****Syntax:**

<Return value> = i\_PA030\_SetTimer0Periode (BYTE b\_BoardHandle,  
 UINT ui\_TeilerFaktor,  
 BYTE b\_Mode)

**Parameter:**

BYTE	b_BoardHandle	Handle of the <b>PA 030</b>
UINT	ui_TeilerFaktor	Divider factor to set the period
BYTE	b_Mode	Counter mode

**Task:**

Sets the period of Timer 0.  
 No return value.

**9) i\_PA030\_SetTimer1Periode****Syntax:**

<Return value> = i\_PA030\_SetTimer1Periode (BYTE b\_BoardHandle,  
 UINT ui\_TeilerFaktor,  
 BYTE b\_Mode)

**Parameter:**

BYTE	b_BoardHandle	Handle of the <b>PA 030</b>
UINT	ui_TeilerFaktor	Divider factor to set the period
BYTE	b_Mode	Counter mode

**Task:**

Sets the period of Timer 1  
 No return value.

**10) i\_PA030\_EnaEnable****Syntax:**

<Return value> = i\_PA030\_EnaEnable (BYTE b\_BoardHandle)

**Parameter:**

BYTE	b_BoardHandle	Handle of the <b>PA 030</b>
------	---------------	-----------------------------

**Task:**

Enables interrupts.

**11) i\_PA030\_EnaDisable****Syntax:**

<Return value> = i\_PA030\_EnaDisable (BYTE b\_BoardHandle)

**Parameter:**

BYTE	b_BoardHandle	Handle of the <b>PA 030</b>
------	---------------	-----------------------------

**Task:**

Disables interrupts. Interrupts are blocked.

## 9.3 Responding INIT030 functions

The different functions can be called up through several programming languages. Thus the interface programs below:

- C : PA030.C  
Description program PA030.H
- Turbo Pascal : PA030.PAS (from version 4.)
- Turbo Basic : PA030.INC (from version 1.0)
- Quick Basic: PA030.BAS (version 4.5) PA.030 BI

## 9.4 ERROR.TXT (only for 16-bit drivers)

When program INIT030.EXE is loaded, file ERROR.TXT is generated. It contains a word which can have the following meanings:

- 1 = Number of parameter is wrong
- 2 = Base address is wrong
- 3 = INIT030 has already been installed
- 0 = No error

ERROR.TXT is used to protect the application program from errors which could occur during the INIT030 program. The error messages are thereby not displayed on the screen.

## 9.5 Program TEST030.EXE (only for 16-bit drivers)

This program carries out a short test of the board **PA 030**.

It uses the functions available in INIT030.

Immediately after starting the program, the "Input parameters" submenu appears. You have to parameter the base address and the software interrupt.

If you return to the main menu TEST030.EXE tries to install the runtime.

It checks if:

- the base address has been properly adjusted
- and the runtime has been already installed.

The main menu leads to the following submenus:

- Input parameters
- Display DIP switches
- Board test
- End of program

### 9.5.1 Input parameters

At this stage you can no longer change the base address and the software interrupt.

You can determine the following parameters: timer interrupt, watchdog divider factors, Timer 0 und Timer 1.



## 9.5.2 Display DIP switches

Shows how to adjust the DIP switches.

Make sure that the DIP switches on the board are adjusted in the same way.

## 9.5.3 Board test

The following options are available:

- Watchdog retrigger
- Temperature measuring
- Main menu

### Watchdog retrigger

A little point appears on the watchdog line after some time. It means that the watchdog time has run down. The interrupts are symbolised on the line below.

A point is displayed each time the timer generates an interrupt.

A point appears each time the option "Watchdog retrigger" has been selected.

### Temperature measuring

This option represents graphically and numerically the value of the temperature currently measured.



## SYSTEM ADDRESS RANGES

Address	Function
000..01F	1st DMA controller
020..03F	1st interrupt controller
040..05F	Timer 82C54
060..06F	Keyboard controller
070..07F	Real time clock & NMI mask
080..09F	DMA-page register
0A0..0BF	2nd interrupt controller
0C0..0DF	2nd DMA controller
0F0..0FF	Coprocessor
0100..01EF	Free
01F0..01F7	Hard disk controller
01F8..01FF	Free
0200..020F	Game port
0210..021F	Extension unit
0220..025F	Reserved
0260..0277	Free
0278..027F	LPT2
0280..02E7	Reserved
02E8..02EF	COM4

Address	Function
02F0..02F7	Reserved
02F8..02FF	COM2
0300...031F	Prototype board (generally free)
0320...032F	Hard disk controller (only PC)
0330...035F	Free
0360...036F	Network boards
0370...0377	2nd floppy disk drive controller
0378...037F	LPT1
0380...038F	SDLC / BSC
0390...039F	Free
03A0...03AF	SDLC / BSC
03B0...03BF	Monochrome graphic board
03C0...03CF	EGA graphic board
03D0...03DF	CGA graphic board
03E0...03E7	Free
03E8...03EF	COM3
03F0...03F7	Diskette drive controller
03F8...03FF	COM1
0400...0FFFF	Free or redundant addressing

These indications refer to AT/386/486 systems.

# INDEX

- ADDIREG 16–21
  - changing the configuration 20
  - removing 21
- base address 12
- board
  - handling 3
  - inserting 13
  - intended purpose 1
  - limits of use 1
  - physical set-up 4
  - programming 24
- channel
  - set-up 23
- component scheme 6
- connection to the peripheral 22
  - connector pin assignment 22
- DIP switches 12
- electromagnetic compatibility 4
- functions 23–32
- I/O mapping 10
- installation 11–14
- Internet
  - software downloads 21
- jumper
  - location 7
  - settings at delivery 7
- limit values 5
- pin assignment 22
- software 15
  - downloads from the Internet 21
- standard software 33–39
- user 2
- watchdog
  - jumper settings 9