PC/104-MIO-2

and

PC/104-MIO-3

Technical Manual

Rev.: 1.7

JUMP*tec*®
Industrielle Computertechnik GmbH
Brunnwiesenstraße 16
94469 Deggendorf/ Germany

PN of Manual: 96035-0006-00-0
Manual Rev.: 1.7
File: Mio2m117.doc

Table of Contents

User Information

In this document JUMP Industrielle Computertechnik GmbH will also be referred to by the short form "JUMP". MIO2 and MIO3 will be used as short forms for the corresponding PC/104-MIO2 and PC/104-MIO3 device.

The information in this document has been carefully checked and is believed to be accurate and reliable. However, no responsibility is assumed for inaccuracies. Furthermore, JUMP reserves the right to make changes to any portion of this manual to improve reliability, function or design. JUMP does not assume any liability for any product or circuit described herein.

Trademarks

AT and IBM are trademarks of International Business Machines.

XT, AT, PS/2 and Personal System/2 are trademarks of International Business Machines Corporation.

Microsoft is a registered trademark of Microsoft Corporation.

Intel is a registered trademark of Intel Corporation.

ModulAT® is a trademark of JUMP Industrielle Computertechnik GmbH

All other products and trademarks mentioned in this manual are trademarks of their respective owners.

General

For the circuits, descriptions and tables indicated no responsibility is assumed as far as patents or other rights of third parties are concerned.

The information in the Technical Descriptions describes the type of the boards and shall not be considered as assured characteristics.

Warranty

Each board is carefully and thoroughly tested before being shipped. If, however, problems should occur during the operation, please check your user specific settings of all boards included in your system. This is often the source of the fault. If a board is defective, it can be sent to your supplier for repair. Please take care of the following steps:

1. The board returned should correspond to the factory default settings since a test is only possible under this settings.
2. If possible, the board will be upgraded to the latest version without additional cost.
3. Upon receipt of the board, please be aware that your user specific settings could have been changed during the test.

Within the guarantee, the repair is free as long as the guarantee conditions were kept. If no fault has been found, you will be charged with the test cost due to the high test expenditure. Repairs outside of the guarantee will be charged.

This JUMP product is warranted against defects in material and workmanship for our guaranteed warranty period from the date of shipment. During the warranty period, JUMP will, at its option, either repair or replace products which prove to be defective.

For warranty service or repair, the product must be returned to a service facility designated by JUMP.

The foregoing warranty shall not apply to defects resulting from improper or inadequate maintenance or handling by the customer, unauthorized modification or misuse, operation outside of the environmental specifications for the product, or improper installation or maintenance.

JUMP will not be responsible for any defects or damages due to a faulty JUMP product other than the products supplied by JUMP.

Features of the PC/104-MIO2/MIO3

The MIO2/MIO3 incorporates following features:

- Two serial interfaces (RS232), configurable as COM1 to COM4 or COM port with user specified address.
- Second serial COM B can be used as RS232/RS485 interface alternatively
- 8 digital open collector outputs
- 14 digital inputs
- 8 analog inputs multiplexed to an AD-converter (only optional)
- Low power CMOS technology
- 5V only power supply
- Non volatile FerroRAM for 255 Bytes of user data
- BIOS calls for efficient use of extensions
- JIDA extensions
- PC/104 Bus interface
- Board size : 90 x 95 mm

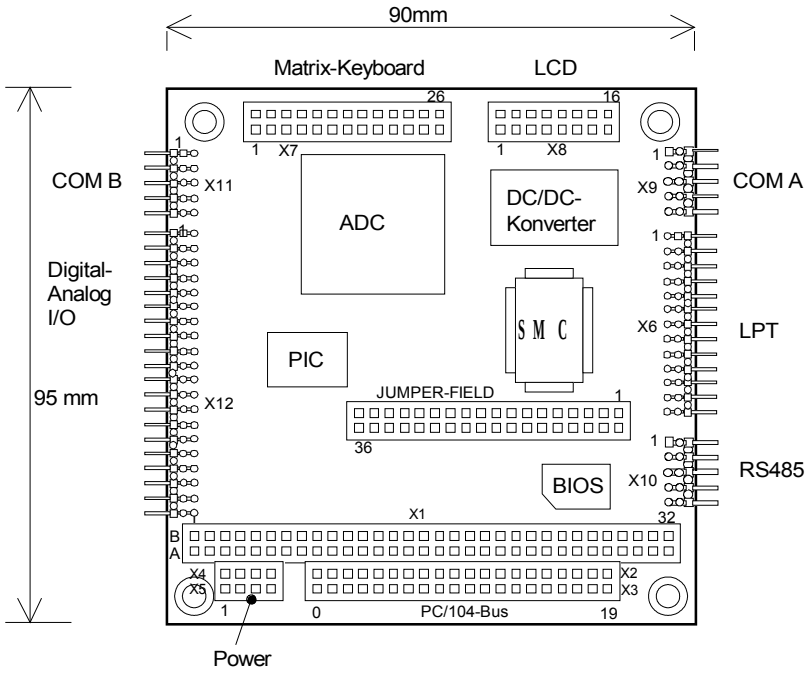
Additional features only on the MIO2:

- Touch Matrix Controller.
- Onboard DC/DC converter for positiv and negative LCD contrast voltages.
- LC display interface.

Additional features only on the MIO3:

- Printer port configurable as LPT1 to LPT3

Layout of the Connectors



Connector Tables for the PC/104-MIO2/MIO3

	PC/104 - Bus				Power	
Pin	**X1,A**	**X1,B**	**X2**	**X3**	**X4**	**X5**
0			GND	GND		
1	/IOCHCK	GND	/SBHE	/MEMCS16	Vcc	GND
2	SD7	RESETDRV	LA23	/IOCS16	n.c.	Key pin
3	SD6	VCC	LA22	IRQ10	n.c.	n.c.
4	SD5	IRQ9	LA21	IRQ11	Vcc	GND
5	SD4	-5V	LA20	IRQ12		
6	SD3	DRQ2	LA19	IRQ15		
7	SD2	-12V	LA18	IRQ14		
8	SD1	/0WS	LA17	/DACK0		
9	SD0	+12V	/MEMR	DRQ0		
10	IOCHRDY	GND	/MEMW	/DACK5		
11	AEN	/SMEMW	SD8	DRQ5		
12	SA19	/SMEMR	SD9	/DACK6		
13	SA18	/IOW	SD10	DRQ6		
14	SA17	/IOR	SD11	/DACK7		
15	SA16	/DACK3	SD12	DRQ7		
16	SA15	DRQ3	SD13	VCC		
17	SA14	/DACK1	SD14	/MASTER		
18	SA13	DRQ1	SD15	GND		
19	SA12	/REFRESH	GND	GND		
20	SA11	SYSCLK				
21	SA10	IRQ7				
22	SA9	IRQ6				
23	SA8	IRQ5				
24	SA7	IRQ4				
25	SA6	IRQ3				
26	SA5	/DACK2				
27	SA4	T/C				
28	SA3	BALE				
29	SA2	VCC				
30	SA1	OSC				
31	SA0	GND				
32	GND	GND				

	LPT[1]	Matrix[2]	LCD[2]	SER A	SER B	RS485	I/O
Pin	**X6**	**X7**	**X8**	**X9**	**X11**	**X10**	**X12**
1	/STB	GND	GND	RLSD A	RLSD B	A2 (send)	AGND
2	/AFD	Y0	VCC	DSR A	DSR B	A1 (receive)	ADC0
3	PD0	RA0	VEE	RXD A	RXD B	B2 (send)	AGND
4	/ERR	Y1	RS	RTS A	RTS B	B1 (receive)	ADC1
5	PD1	RA1	R/W	TXD A	TXD B	Term 2	AGND
6	/INIT	Y2	Enable	CTS A	CTS B	Term 1	ADC2
7	PD2	RA2	DB0	DTR A	DTR B	/E485	AGND
8	/SLIN	Y3	DB1	RI A	RI B	GND	ADC3
9	PD3	RA3	DB2	GND	GND	VCC	SYNC
10	GND	Y4	DB3	VCC	VCC	n.c.	ADC4
11	PD4	RA4	DB4				I²CLK
12	GND	Y5	DB5				ADC5
13	PD5	RB0	DB6				I²DAT
14	GND	Y6	DB7				ADC6
15	PD6	RB1	Backlight +				GND
16	GND	Y7	Backlight -				ADC7
17	PD7	RB2					OC0
18	GND	/MCLR					In0
19	/ACK	RB3					OC1
20	GND	OC0					In1
21	/BUSY	RB4					OC2
22	GND	OC1					In2
23	PE	RB5					OC3
24	GND	RB7					In3
25	/SLCT	RB6					OC4
26	VCC	VCC					In4
27							OC5
28							In5
29							OC6
30							In6
31							OC7
32							In7
33							In9
34							In8
35							In11
36							In10
37							In13
38							In12
39							VCC
40							VCC

[1] (PC/104-MIO3 only)

[2] (PC/104-MIO2 only)

Working with the PC/104-MIO2/MIO3

Overview

The MIO2/MIO3 offers some useful extensions compared with other peripheral modules, which will be described below:

Software Configurable I/O-Controller
The on board I/O controller chip offers two serial and one parallel port. Any of the two serial ports can be configured as COM1 - COM4 or as additional COM Port with a non-standard address. The parallel port can be configured as LPT1 - LPT3 and used as standard PP or EPP.

Serial Interface
The serial interface is realized within the FDC37C665 controller. The interface supports two serial ports COMA and COMB. Via V.24 drivers the COM-signals are driven with the V.24 level.

The V.24 levels are generated onboard with a DC/DC-converter, so only a +5 volt supply is necessary.

RS485
COMB is equipped with two RS485 transceivers (realised through MAX483). By pulling the /E485 signal low on connector X10, COMB can be configured to work with RS485 instead of RS232. The maximum data transfer rate for this interface is 115 Kbit/s.

Parallel Interface (PC/104-MIO3 only)
The printer interface is realized within the FDC37C665 controller. The interface may be configured as bi-directional Standard Parallel Port (SPP) or as Enhanced Parallel Port (EPP).

Support for LC-Character-Displays (PC/104-MIO2 only)
Character Displays may be connected to the MIO2. The complete display mode selection and configuration is supported by the video BIOS.

Almost any LC-character display equipped with an Hitachi HD44780 or compatible controller may be directly connected to the LC interface. The LC interface is used alternatively to LPT interface. Therefore only one of both is realized on either MIO2 or MIO3. Nevertheless it is necessary to leave configuration of parallel interface turned on in MIO2SET utility, when working with a LC display. By setting parallel to OFF the LC-character display is turned off too.

The MIO2 offers easy to use output function via BIOS INT10h. Backlight can be enabled and disabled by software.

An on board DC/DC converter is capable of generating positive (0V to 4.1V) or negative voltages (-5V to -12V) which are needed by many LCD panels. The voltage (the panel's contrast) can be controlled by software or by the user via keyboard. Note that the default setting (defined by onboard hardware configuration) for the contrast range reaches from 0.6V to +3.V !

Decoder for Matrix Keyboard (PC/104-MIO2 only)
Any matrix keyboard or matrix touch panel may be directly connected to the MIO2. The matrix is scanned by hardware. If a connection is detected, it will be processed just like a keystroke from the PC keyboard. The assignment between keys on the matrix keyboard and generated scan codes is completely user configurable. The configuration is stored in a non volatile FRAM.

Digital Outputs
Eight digital open collector outputs are available, each capable of sinking 250mA (a total of 500mA for all 8 outputs may not be exceeded). Maximum Voltage is +30V. The outputs are controlled via BIOS calls.

Digital Inputs
There are 14 digital inputs available to the user. Each input is protected by a 1k series resistor and a clamp diode pair DBAV99. This allows input voltages ranging from -30V to +30V. The switching level is TTL compatible. The inputs are controlled via BIOS calls.

Analog Inputs (the A/D-converter is only optional available))
The PC/104-MIO2/MIO3 may include an optional onboard LM12458 A/D-converter from NATIONAL Semiconductor. This offers a data acquisition system with a 12 bit + sign resolution and sample-and-hold. Up to 32 consecutive conversions can be stored in a FIFO data buffer. Eight analog input lines are available for the user, which can be configured as eight single ended or four differential channels. A programable watchdog mode can compare input values against two limits. The A/D-converter functions can be controlled via memory mapped access.

Non-volatile User RAM
256 bytes of non volatile RAM (residing in an FRAM) can be accessed by the user via simple BIOS calls.

Easy configuration
Configuration data for most of the features described above can easily be changed by using the program MIO2SET. This is a mouse compatible, menu driven software following the SAA standard.

All the settings may also be made by your own software via well documented BIOS calls. The software diskette will be delivered with this technical manual.

Software configurable I/O-Controller
The software configuration features of the on board I/O controller are fully supported by on board BIOS and the MIO2SET utility program.

Systembus (PC/104-bus)
The **PC/104** bus is defined by the AMPRO company. The PC/104 bus is electrically identical to the standard AT-Bus (ISA-Bus) and functionally full compatible to the AT. The driver power is 12mA instead of 24mA on the ISA bus.

Power Supply
MIO2/MIO3 boards only need a +5V power supply. The +5V can be supplied through a power connector on board.

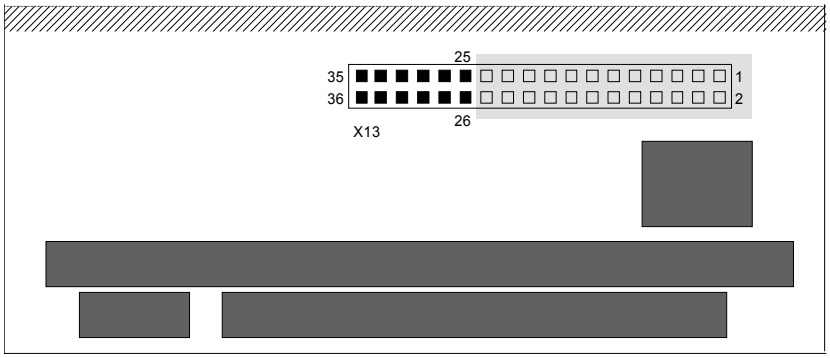
Jumper Settings

The adaption to the given hardware should be done with the jumper-field (X13). It is possible to configure the actual hardware-address and to choose between different hardware-interrupts of the serial ports.

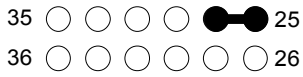
There is a relation between the actual hardware-address and the location of the BIOS extension. The figures on the next pages shows the detailed relation.

Selection of BIOS-Location and I/O-Address

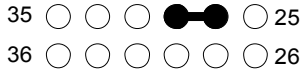
The location of the jumper field X13 is shown on page 6 of this manual.



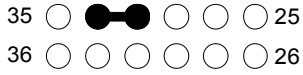
BIOS-Address C000:0000h
I/O-Address 018Fh, 118Fh



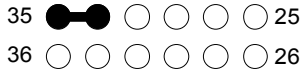
BIOS-Address C400:0000h
I/O-Address 0210h, 1210h



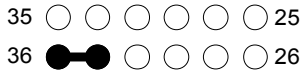
BIOS-Address C800:0000h
I/O-Address 0300h, 1300h



BIOS-Address CC00:0000h
I/O-Address 009Fh, 109Fh

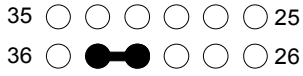


BIOS-Address D000:0000h
I/O-Address 018Fh, 118Fh

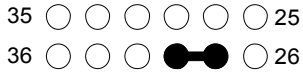


default

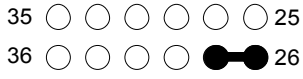
BIOS-Address D400:0000h
I/O-Address 0210h, 1210h



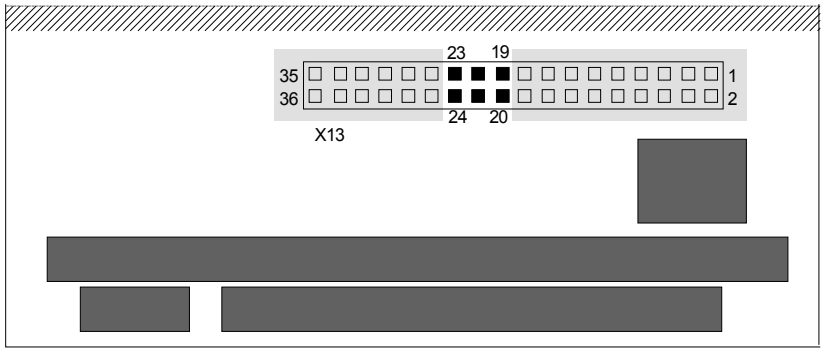
BIOS-Address D800:0000h
I/O-Address 0300h, 1300h



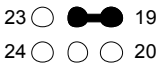
BIOS-Address DC00:0000h
I/O-Address 009F, 109Fh



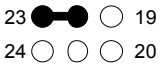
Interrupt selection for AD Converter



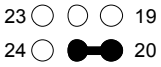
ADC, IRQ 15



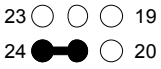
ADC, IRQ 10



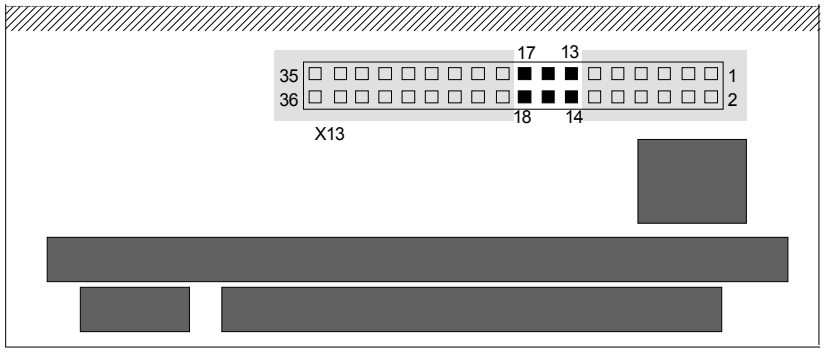
ADC, IRQ 5



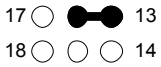
ADC, reserved, do not use



Interrupt selection for serial port COM A

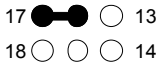


serial port A, IRQ 12

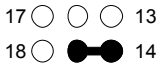


default

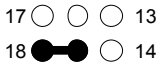
serial port A, IRQ 11



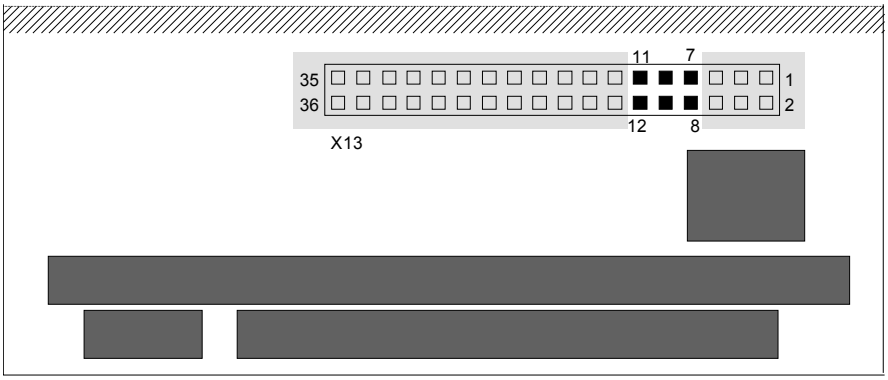
serial port A, IRQ 10



serial port A, IRQ 4

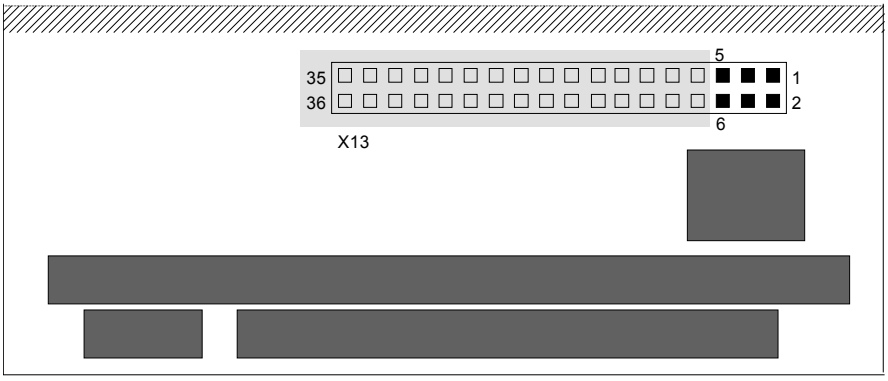


Interrupt selection for serial port COM B



serial port B, IRQ 15	11 ○ ●—● 7 12 ○ ○ ○ 8	default
serial port B, IRQ 12	11 ●—● ○ 7 12 ○ ○ ○ 8	
serial port B, IRQ 11	11 ○ ○ ○ 7 12 ○ ●—● 8	
serial port B, IRQ 3	11 ○ ○ ○ 7 12 ●—● ○ 8	

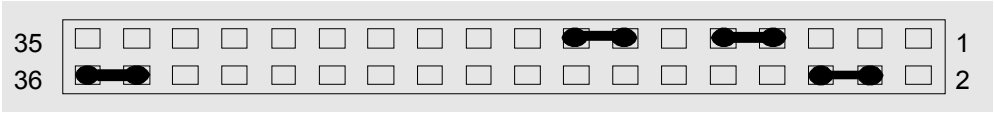
Interrupt selection for parallel port LPT (only for PC/104-MIO3)



LPT, IRQ 12	5 ○ ●—● 1 6 ○ ○ ○ 2	
LPT, IRQ 15	5 ●—● ○ 1 6 ○ ○ ○ 2	
LPT, IRQ 5	5 ○ ○ ○ 1 6 ○ ●—● 2	
LPT, IRQ 7	5 ○ ○ ○ 1 6 ●—● ○ 2	default

Default settings

The MIO2/MIO3 is provided with following default settings:



X13

BIOS Address	D000:0000h
I/O Address	018Fh, 118Fh
ADC	No IRQ
COMA	IRQ 12
COMB	IRQ 15
LPT	IRQ 7

The default setting for optional ADC is IRQ 10.

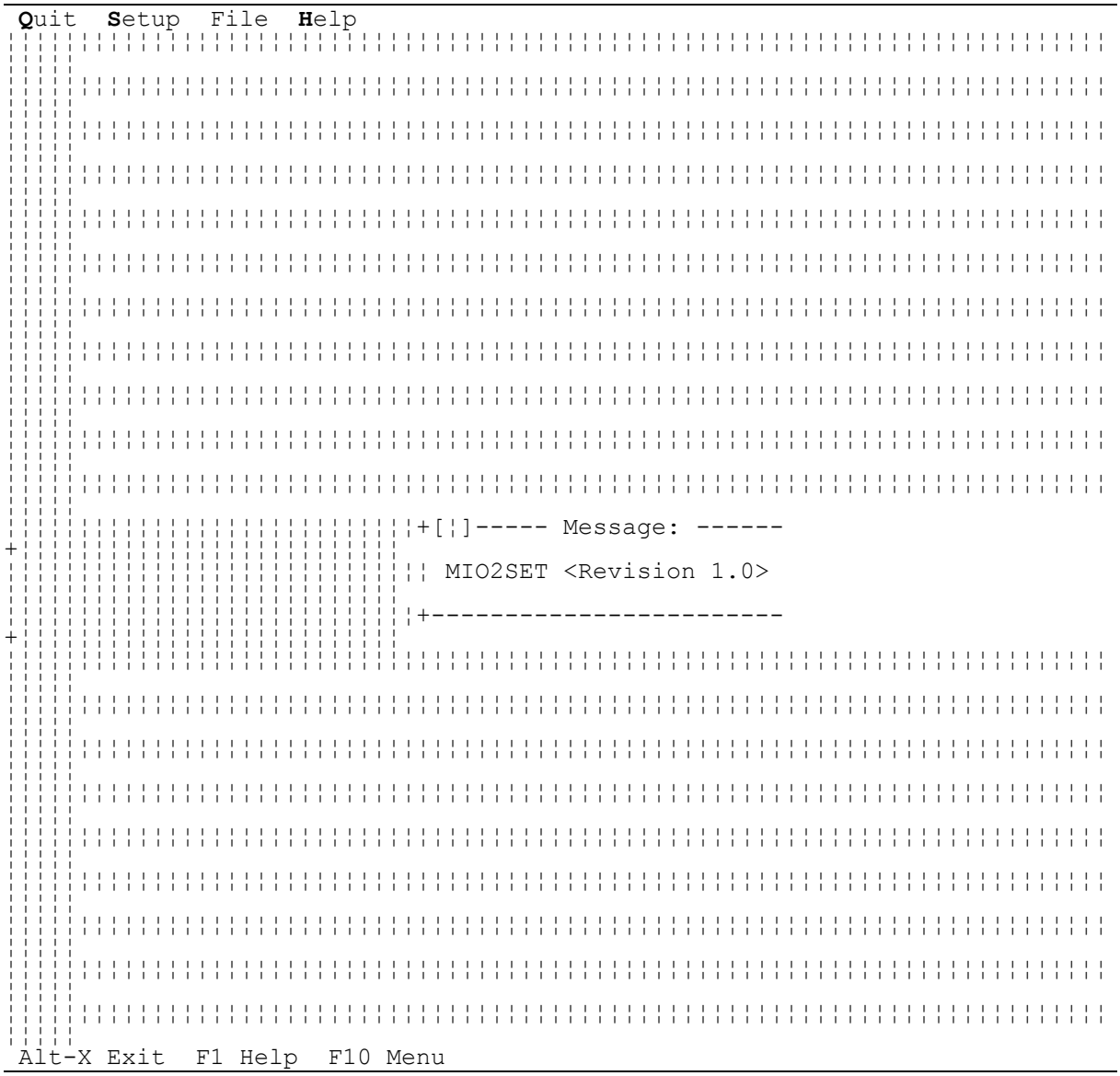
Using the MIO2SET Utility

General Information

A special setup utility, called MIO2SET, is available for the MIO2/MIO3 board. MIO2SET may be used to configure the on board I/O components, to setup the matrix keyboard decoder, the LCD sub system and to obtain information about the board. The program's name is MIO2STxx.EXE, where xx is the version number. This version number should always be the same as the version of the BIOS. If you use BIOS version 1.0 (labeled MIO2R110), you should also use MIO2ST10.EXE.

Starting MIO2SET

From the DOS-Prompt, simply Type □□□□□□□. The program starts and welcomes you as shown here:



Basic Concepts

MIO2SET follows the SAA standard for user interfaces, which allows easy navigation through the program by using a mouse or the keyboard.

Selecting Menus

- To select a menu, click on the menu's name with your mouse.
- Alternatively, you may select a menu by keyboard. The easiest way to do this is to find the highlighted character in the menu's name (on color systems, this character is red, on monochrome systems, the character may be underlined). Then hold down the key and press the key with the character you found. Example: Press + to activate the "**S**etup"-Menu.
- A second way to select a menu by keyboard is the following: hit the key to jump to the menu bar, select the menu you need using the ¥¤ keys, then hit ¢ to pull the menu down.
- If a menu was accidentally selected, you may deselect it by clicking on the empty desktop or by hitting the ○ key.

Selecting a Menu Entry
To select and activate a menu entry, do one of the following:

- Click on the entry with the mouse.
- Select the entry with the ¢ keys and hit ¬
- Hit the highlighted character (without holding down)

Working with Dialog Windows
- To activate any item in a dialog window, simply click to it with the mouse.
- If you do not have a mouse, select the dialog item by holding down the key and hitting the key that is highlighted on or beneath the desired item. Use the (Space) key to switch items on or off. Use the ¢ keys to select one of different options. Hit to jump to the next item or ¨ to jump to the previous item.
- ○ may be used within any dialog to cancel the operation and return to the main menu.

The MIO2SET Menus

The following graphic gives you an overview on all the menu entries of the MIO2SET software:

Quit - Exit MIO2SET

The function "Exit" located in the "Quit" menu enables you to leave the MIO2SET software. Click on "Quit", then click on "Exit..." to do this.

Alternatively, you may use □+□ to leave the program.

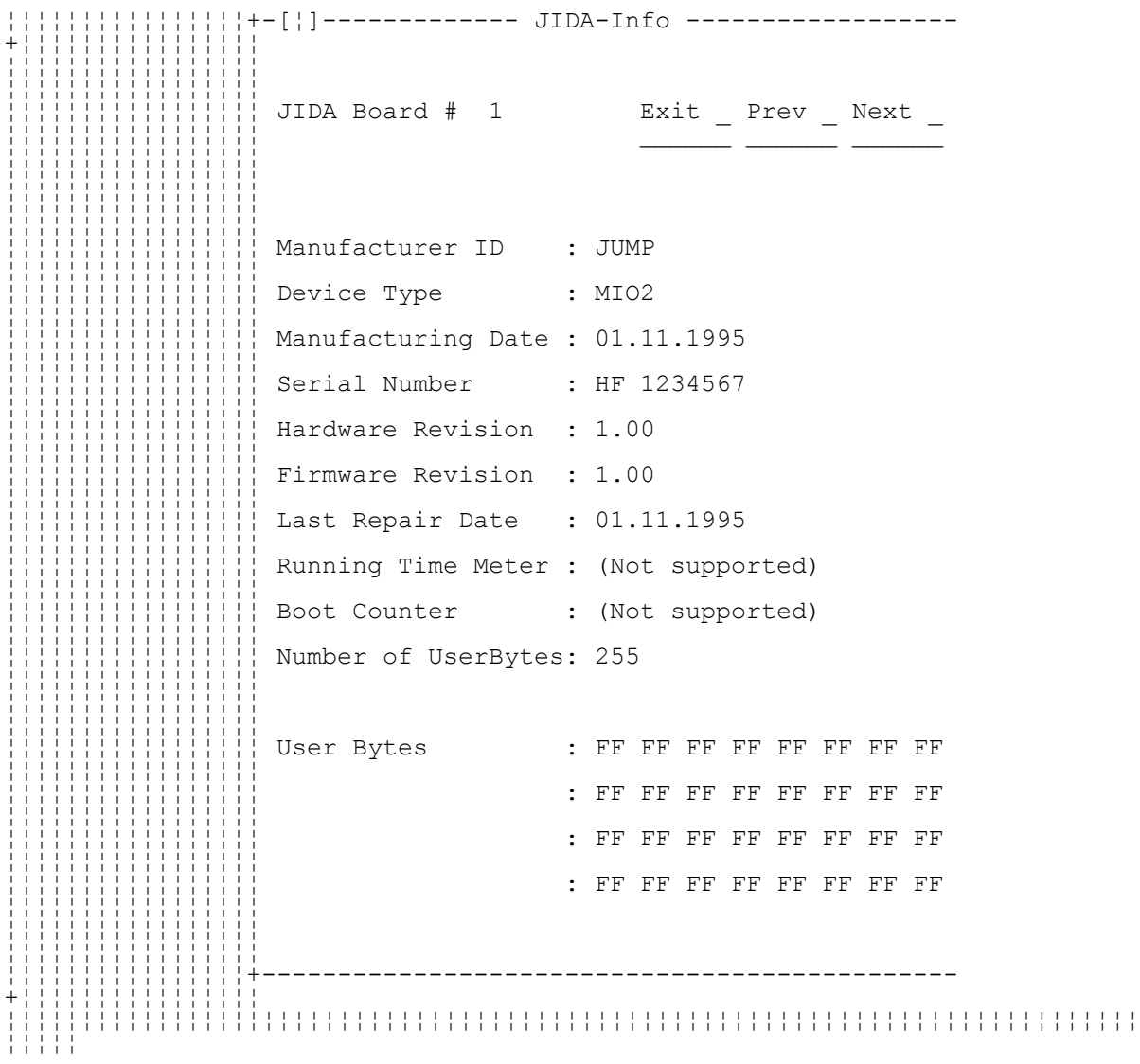
Close dialogs with ○ before trying to leave the program □+□ or clicking on "Quit" will not work when in dialog boxes.

JIDA Info Dialog

JUMP introduced JIDA as a new standard for all boards with onboard BIOS. JIDA enables the user to access additional features of the board. The JIDA Info Dialog shows all available information on the first JIDA board in your system. (Board ID = 1).

You may use the "Prev" or "Next" button, to switch to a JIDA board with a higher or lower ID (JIDA IDs range from 1 to 127). If no previous or next board is available, the corresponding button will be invisible.

The "Ok" button may be used to leave the dialog, which looks like follows:



A description of each line is given in the following. For further details about accessing the JIDA functions refer to the section "Programmer's Guide" in this documentation.

Manufacturer ID:	Identifies the manufacturer. Mostly "JUMP"
Device Type:	Identifies the device "MIO2" with PC/104-MIO2, "MIO3" with PC/104MIO3 boards
Manufacturing Date:	Tells you, when your board was manufactured.
Serial Number:	A board individual serial number. This allows to identify each single board.
Hardware Revision:	The hardware revision of your board.
Firmware Revision:	The firmware (=BIOS) revision of your board.

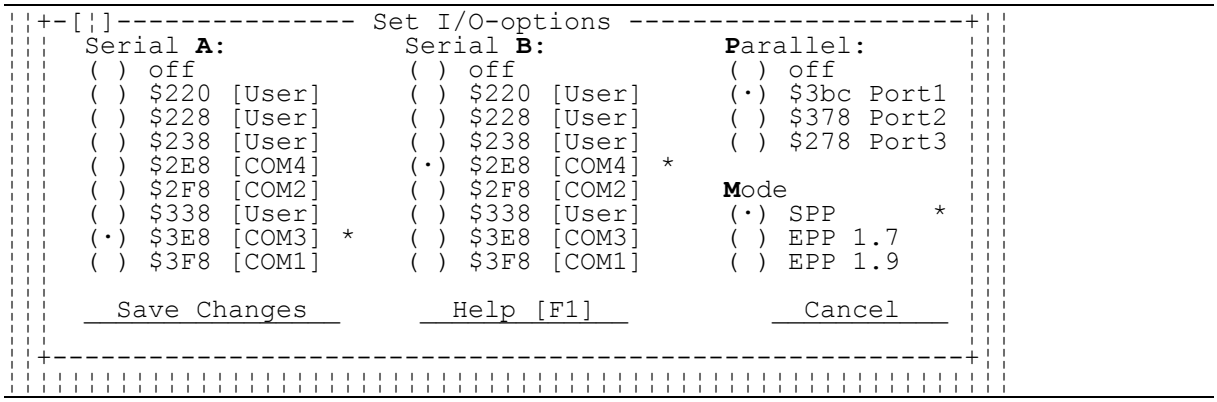
Last Repair Date:	Shows, when the board was repaired last. If this date is smaller or equal to the manufacturing date, the board was never repaired.
Running Time Meter:	Total Running Time (in hours) of the board. Not supported by the MIO2/MIO3.
Boot Counter:	Incremented with every boot of the system. Not supported by the MIO2/MIO3.
Number of Userbytes:	Informs you, how many bytes of user accessible non volatile ram are available.
User Bytes:	Shows the first 32 bytes of non volatile user data (or all bytes if less than 32 bytes are available)

I/O-Settings

This dialog enables you to configure the on-board I/O-controller on the MIO2/MIO3. If you select this option, MIO2SET will try to determine, which I/O-functions are present in the system. This test may last up to 30 seconds.

The MIO2/MIO3 conducts the same automatic I/O test during boot. If any of the I/O-controllers functions collides with a component on another board, the on board component will be automatically turned off.

After the test is finished, this screen will be displayed:



Serial A: Allows you to configure the first of two serial ports. You can switch the port off or use it as port COM1 - COM4 or with a user defined address. There are different Interrupts possible which can be selected with jumpers on the MIO2/MIO3. If a defined address is already in use on another board in the system, the COM port on the MIO2/3 will be turned off during boot up.

Serial B: Like "Serial A". Selecting the same COM for "Serial A" and ‚Serial B" will cause a turn off of "Serial B" during reboot of the system. There are some combinations which are not allowed between "Serial A" and "Serial B". After you have selected Serial A, all illegal addresses will be grayed out on Serial B.

Parallel: Use this field to select the address of the printer port or to turn off the printer port. (only for PC/104-MIO3). Remember not to turn off printer port on the MIO2, because this will also turn off your character LC display.

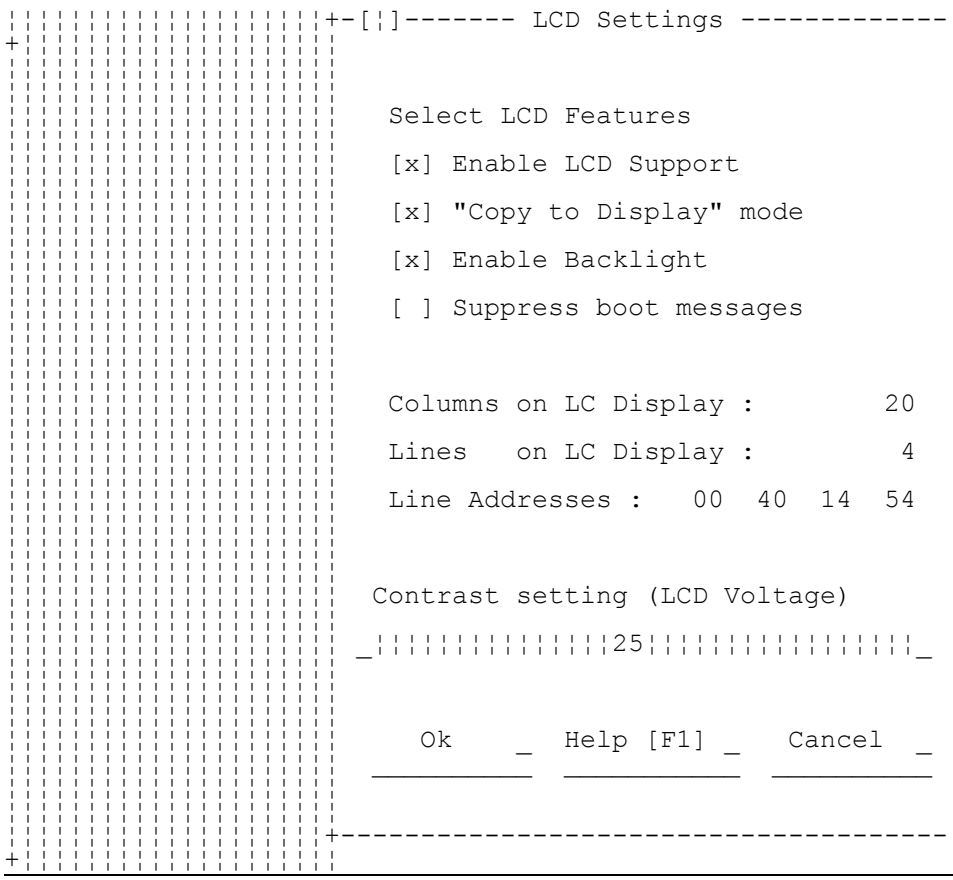
Mode: SPP Selects a bidirectional Standard Printer Port, EPP selects Extended Parallel Port. Note: You can either use the parallel port or the LCD interface, but not both at a time.

"Save Changes" will write your changes into the configuration FRAM. After a reboot of the system, the new settings will be active. "Help" gives you some usefull information about the current screen. "Cancel" leaves this menu item without saving any changes you´ve made before.

To provide a shorter way for field selection, every field contains one highlighted character (for example "B" in "Serial **B**"). Simply hold down ○ and press the highlighted key to select the corresponding field.

LCD Options

This dialog is used to configure the LCD options. All options except the ”Suppress boot messages” are only usefull on the MIO2. The MIO3 offers no LCD interface.



Enable LCD Support: By selecting this function, you tell the on board BIOS that you want to use the LC-Display connector instead of the parallel port connector. You can either use the parallel port on PC/104-MIO3 or the LCD interface on PC/104-MIO2. Nevertheless you have to leave "parallel” setting in the menu "IO settings" turned on. If these setting are enabled, the display will be initialized and powered during boot time. The parallel lines will then be controlled by the INT10h BIOS calls described later in this manual.

Copy to Display mode: This enables a special mode, where all outputs that are made via INT10h BIOS calls are also copied to the LC Display. Of course, this is only true as far as it is directed to that upper left part of the screen that corresponds with the size of the character display. Scrolling - as known of CRT-monitors - will not effect the character display output, because these types of display usually have no memory to store the missing lines. Therefore, if you want to see the actual DOS prompt line, you have to clear the screen after your output to the character display has reached the last line.

If you are using no character display and your output to a VGA monitor seems to be very slow while displaying the first lines, you possibly enabled this feature. Turn items ”Enable LCD Support” and ”Copy to Display mode” of when using no character display at the MIO2.

Enable Backlight: This setting allows you to enable or disable Backlight during boot.

Suppress Boot Messages: If you enable this setting, the MIO2/MIO3 extension BIOS will not make any outputs to the screen during boot.

Note: The rest of the settings in this dialog are meaningless, if the LC-Display is not enabled (refer to the first setting of this dialog). Nevertheless, if LC-Display is enabled, the following informations are very important for a proper operation of your display. Incorrect settings will cause unpredictable results to be displayed.

Columns on LC Display: Enter the correct number of columns that your LC-Display can handle. If you use a 20*4 display, enter 20.

Lines on LC Display: Enter the correct number of lines that your LC-Display can handle. If you use a 20*4 display, enter 4.

Line Addresses: Displays with the Hitachi HD44780 display controller have special RAM addresses assigned to each display position. To enable the BIOS to conduct functions like scrolling or line wrap, it must be told this adresses in hexadecimal forms. Most of the usual displays use addresses 00h for line 1, 40h for line 2, 14h for line 3 and 54h for line 4. Do not change this values unless you are sure that your display needs different values.

Contrast Setting: This scrollbar controls the onboard voltage generator. It is used to adjust the contrast of those LC-Displays that need an additional driving voltage as most panels in the market do. Press □□ to select the scrollbar, then use the ¥¤ keys to adjust. The contrast may be adjusted in 64 steps ranging form 0 (minimum voltage) to 63 (maximum voltage).

The default onboard configuration results in 0 = +3.0V and 63 = 0V.

Matrix Keyboard setup (only useful on MIO2)

Overview

A matrix keyboard may be connected to the MIO2 via the matrix keyboard connector. Pins Y0-Y7 are outputs that should be wired with matrix lines, the inputs RA0-RA4/RB0-RB4 should be connected with matrix columns. Each connection between a line and a column then generates an internal scan code which is transferred to the system. The BIOS converts this internal scan code to a keyboard scan code using the matrix decoder table. This matrix decoder table may completely be configured by using the MIO2SET utility. The matrix definition dialog looks like this (picture showing the enhanced matrix keyboard):

```
+-[ ]------ Set scan codes for matrix keyboard ----------
|  |||||||||||  RA0 RA1 RA2 RA3 RA4 RB0 RB1 RB2 RB3 RB4   Name
|
|     3   5   7   9  11  13  15  17  19  21 Pin
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF   2 Y0
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF   4 Y1
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF   6 Y2
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF   8 Y3
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  10 Y4
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  12 Y5
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  14 Y6
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  16 Y7
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  18 Y8
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  20 Y9
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  22 YA
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  24 YB
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  26 YC
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  28 YD
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  30 YE
|
|    FF  FF  FF  FF  FF  FF  FF  FF  FF  FF  32 YF
|
|
|   Ok _  Help _  Cancel _   Teach In  _   Save _  Load _
|   ____  ______  ________   ___________   ______  ______
|
|  +-------------------------------------------------------
+  |
Tab select line.  Space toggle on/off.  _ _ select option.  ESC exit
dialog.
```

Defining Scan Codes manually

The dialog box contains one entry for each crosspoint on the matrix keyboard. To define the scan code that should be generated when RA3 is connected with Y1, you must fill in the fourth column in line two.

The numbers, which are entered in hexadecimal form, are the scan codes, not the ASCII codes of the corresponding key. Scan codes are individual numbers for every key on your keyboard. For example, the scan code of the [Esc] key is 01, the [A] key has scan code 1Eh, while the left [Shift] key is associated with scan code 2Ah. A detailed listing of scan codes can be found in the Section ”Keyboard Scan Codes” in this Manual.

Some keys on MF-II keyboards generate so called "extended scan codes". This means, when the key is pressed, not only the scan code is transmitted to the keyboard, but the scan code is preceded by a "pre-code" with value 0Eh. If you want the matrix keyboard to generate one of those "extended scan codes", add 80h to the scan code. Example: Entry 1Ch will result in scan code 1Ch (associated to the

[Return] key). Entry 9Ch (added 80h) will generate the scan codes 0Eh 1Ch, designating the [Enter] key on the numeric keypad.

Using the Teach-In-Mode

As it may be difficult to determine the desired scan code or the row and column of a key on your matrix keyboard, MIO2SET offers an easy to use "Teach-In-Mode":

To determine the row and column associated with a key on the matrix keyboard, simply connect the matrix to the MIO2 and press the desired key. The cursor will automatically jump to the correct input field. We call this ”Auto-Jump”. In addition, there is a function to determine the scan code of any key on your standard AT keyboard. To do this, place the cursor in the input field that is to be filled (by using □, ¨ +□, □¢ keys or by using the "Auto-Jump" method described above). Then click on the ”Teach In” Button or press □□ Release all keys then. MIO2SET will now display a message, asking you to press the desired key. Press any key on your keyboard now. MIO2SET automatically determines the scan code and enters the value to the selected input field.

The scan codes may also be defined by your own software using the JIDA extension interrupt. See section ”Programmers Reference” for details.

For further details on how to connect a 8*10 or a 16*10 matrix to the MIO2, refer to section ”Matrix Touch Controller”

Load-Settings

This entry opens a standard ”file selection list box”, which offers the possibility to load a configuration file from floppy- or harddisk. The configuration file can be generated by making all necessary settings for MIO2 configuration and then saving them with the dialog ”Save Settings”.

Save-Settings

This entry is the counterpart to dialog ”Load Settings”. After you´ve made all necessary settings concerning I/O, LC display support and matrix translation table, this item allows to save your configuration to a file. Now you´just have to load these saved settings to achieve you´re standard configurations without entering every single menu item again and again.

Controlling the MIO2 by Keyboard (MIO2 only)

Some important and often used functions concerning the LCD options have been assigned to special key-combinations. To use one of these combinations, you have to hold down the § and the □ key and then strike one of the numbers ~ -□. Please note that you have to use the numbers on the alpha keyboard (located above the □□□□□□□□□□ keys). The keys on the numeric pad have no effect.

The following information about the key-functions are concerning the standard hardware configuration of the contrast voltage. If you are using a MIO2 with none standard configuration (e.g. negative contrast voltage) the output voltage will be decreased with §+□+I keys and increased with §+□+□ keys.

§+□+I	Increase output voltage (maximum +3.0V). To do this, internally the programmed DC-DC value is decreased by one. If the DC-DC value is already 0, no action is performed.
§+□+□	Decrease output voltage (minimum +0.6V). To do this, internally the programmed DC-DC value is increased by one. If the DC-DC value is already 63, no action is performed.
§+□+□	Set output voltage to mid range. To do this, internally the programmed DC-DC value is set to 32.

§+□~+ Toggle backlight on and off.

Information displayed during Boot

At Power up time, the MIO2/MIO3 extension BIOS will check for external I/O-components, read your configuration data out of FRAM and program the on-board I/O-controller corresponding to your selections. If any external component is found with the corresponding on-board component configured as ”enabled”, the on-board component will be disabled to avoid address conflicts.

Please note that none of the following messages will be displayed, if ”Suppress boot messages” is selected in MIO2SET’s ”Boot Options” dialog.

During boot, the following information might be displayed:

```
MIO2 Extension BIOS V 1.0 <MIO2R110.ROM>  <D000-018F-20-In-Cn-LC-Cp>
Configuration Checksum Invalid. Default Parameters will be used.
   Serial A   : at 3E8h [COM3]            Serial B   : at 2E8h [COM4]
   LCD-Driver : at 3bch  SPP
```

The Title Section

```
MIO2 Extension BIOS V 1.0 <MIO2R110.ROM>
```

The first part of line one describes the BIOS and tells you about the revision. The name enclosed in <> is the internal file reference used by JUMP. You should find a label with the same information on your BIOS-Prom.

The System Information

```
                                        <D000-018F-20-In-Cn-LC-Cp>
                                          (1) (2)   (3)(4)(5)(6)(7)
```

The right half of line one contains information about the configuration of your system:

(1) The BIOS-Address you selected by the on board jumpers is displayed here. Only the segment part of the address is displayed in a hexadecimal notation.

(2) The MIO2/MIO3 uses two I/O-addresses to access all the additional features of the MIO2/MIO3. Those extension registers are used to access the FRAM, to program the I/O-Controller, to communicate with the matrix decoder, to set the display contrast to control the open collector outputs and to read the inputs. During boot, the BIOS selects an address for the extension registers depending on the BIOS-Address you selected and displays the first address used at position (2). The information is displayed hexadecimal.

(3) The DC/DC converter can be adjusted by software. The output voltage may be changed by the MIO2SET utility or by special key-combinations to change the displays contrast. The actual contrast setting is stored in FRAM and displayed in position (3) during boot. As the display is hexadecimal, possible values range from 00 to 3F.

(4) The MIO2/MIO3 contains a lot of identification information, that can be obtained by using the adequate JIDA functions. The section ”Programmer’s Guide” in this documentation describes how to access this information from your programs. The MIO2SET utility may be used too, to display the information. As a part of this JIDA information is stored in FRAM, and this information may be very important for some applications, there is a checksum to protect it. The information is checked during boot. If everything is ok, In- will be displayed at position (4), otherwise, this position will remain blank.

(5) The configuration for I/O- and display are stored in FRAM, too. During boot, the BIOS checks, if the configuration is valid and displays Cn- at position (5), if everything is ok. Otherwise, this position remains blank and the BIOS uses default-values to program the I/O-controller and the display.

(6) If you have enabled the LC-Display in MIO2SET, LC will be displayed at position (6).

(7) If you have enabled the Copy to Display Mode in MIO2SET, Cp will be displayed at position (7).

Memory Allocation Error

```
MIO2 BIOS NOT INSTALLED! Data memory could not be allocated.
```

The MIO2/MIO3 BIOS needs some Bytes of memory for internal use. This memory is allocated from the top of the DOS memory pool. If there is any condition that makes it impossible to allocate the needed amount of RAM, the BIOS will display the above message. In this case, the extensions will not install and all on board features will remain unaccessible. As far as we know, this should only happen if the system is equipped with less than 64kB of memory (which is a very strange configuration, isn't it?)

FRAM Error

```
MIO2 BIOS NOT INSTALLED! F-RAM defective or missing.
```

The information stored in the FRAM is very vital to the MIO2/MIO3 extension BIOS. Due to this, the access to FRAM is checked every time before the extensions are installed. If there are any problems accessing the FRAM, the above message is displayed, the extensions will not install and all on board features will remain unaccessible.

The most likely reason for this message is an I/O-Address conflict. If the I/O locations used by the MIO2/MIO3 (see section "Jumper Settings" for details) for FRAM access are already occupied by other components, accesses to the FRAM will probably fail. In this case, try to use a different BIOS- and I/O-address.

Another reason could be, that the I²C bus is blocked. This bus is used for communication between BIOS and FRAM. As the I²C bus is available to external circuitry at connector X7, pins 23 and 25 and on connector X12, pins 11 and 13, it should be checked, if any external component or cable might block the bus.

If none of the above reasons matches, there probably is a hardware defect and the MIO2/MIO3 board should be returned for repair.

Configuration Checksum Error

```
Configuration Checksum Invalid. Default Parameters will be used.
```

The configuration information for I/O- and display, which are made by using the MIO2SET utility are stored in FRAM. During boot, the BIOS checks, if the information is valid. If a checksum error is detected, the BIOS will use default-values to program the I/O-controller and the display. If you get this message during boot, you should use the MIO2SET utility to check your configuration. If you save the configuration within any dialog of the MIO2SET utility, a new checksum will be calculated.

IO-Settings

```
   Serial A   : at 3E8h [COM3]            Serial B   : at 2E8h [COM4]
   LCD-Driver : at 3BCh  SPP
```

The actual status for every of the three on board components is shown in these lines. Each component name is followed by an I/O address or the words `disabled` or `external`.

`disabled` means that the on-board component has been disabled by the user.

`external` is used to tell you that an external component has been found at the corresponding address and the on-board component has been disabled to avoid address conflicts.

If a serial port is enabled, you will get additional information about the I/O-address used by this COM-Port. If one of the standard addresses was used (3F8h, 2F8h, 3E8h, 2E8h), the port name ("COM1") will be displayed in square brackets, otherwise, only the address will be displayed.

With the printerport, you will get information about the address that is being used for the port. In addition, the BIOS displays `SPP` or `EPP` to indicate if the port is programmed as Standard Printer Port or as Extended Parallel Port

The LC-Display Interface (MIO2 only)

Connecting a Display

For connecting character LC-Displays, the MIO2 offers an interface with 16 Signals. If you want to see the -prompt and the DOS messages on your display too, you have to enable the corresponding LCD options with the MIO2 setup utility as described in chapter ”Working with the PC/104-MIO2/MIO3”.

LCD Connector Pinout (X8) and Signal explanation

Various types of character displays may be connected to the 16pin plug X8. The pinout is shown in this tables:

Name	Pin
GND	1
VCC	2
VEE	3
RS	4
R/W	5
Enable	6
DB0	7
DB1	8
DB2	9
DB3	10
DB4	11
DB5	12
DB6	13
DB7	14
Backlight +	15
Backlight -	16

GND	Ground
VCC	+5V
VEE	output voltage from DC/DC converter for adjusting contrast.
RS	Selects different registers of the LCD
R/W	Read / Write
ENABLE	Signal for data assumption
DB0-DB7	Data signals
BACKLIGHT+	Positive supply voltage for backlight (+5V)
BACKLIGHT-	Negative supply voltage for backlight (GND)

Backlight

A series resistor (4Ω) connects Pin 15 with VCC (+5V), therefore a led backlight may be plugged in directly on pin 15 and pin 16 of X8.

Contrast voltage

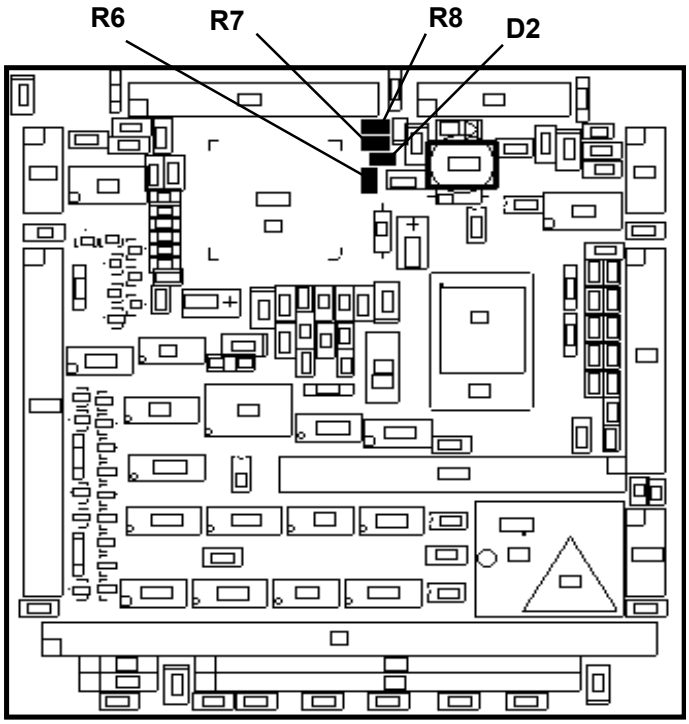
Today there are many different LCDs available which need various contrast voltages to achieve the optimal display output. However, using the wrong contrast voltage may damage the LCD or at least give an unsatisfactory display result.

The PC/104-MIO2 boards of JUMP GmbH have the ability to generate the contrast voltage by a MAX749 and some resistors in connection with a diode onboard.

To avoid destruction of a LCD connected to the PC/104-MIO2 by a customer, he has to check if the necessary resistors and/or diode are mounted to the printed circuit board. JUMP GmbH uses a standard configuration of these components, which sets the contrast voltage to a defined value for one LCD type - others connected to this configuration might be damaged or at least show no satisfactory display. The standard configuration offers a positiv contrast voltage from 0.6V to 3V as described in examble 3.

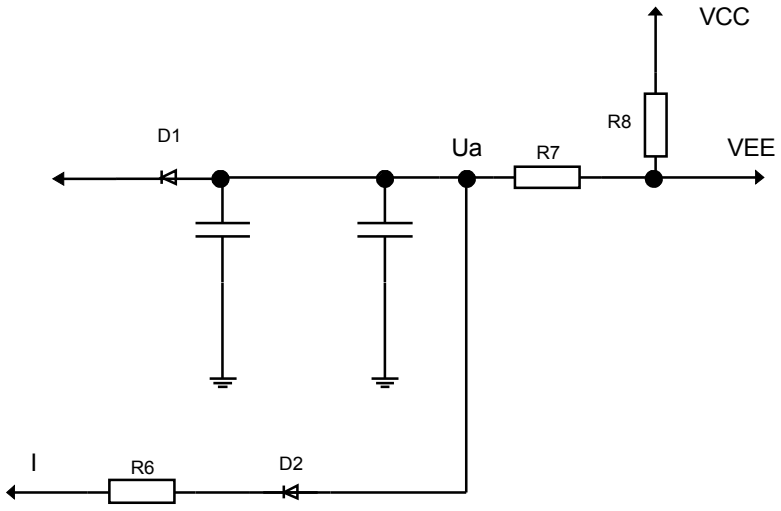
Therefore the customer who wants to connect his special display type to the PC/104-MIO2 has to determine the necessary resistors/diode following these informations.

The components R6, R7, R8 and D2 are responsible for the generation of the contrast voltage. The placement of these components on the PC/104-MIO2 printed circuit board can be found in drawing below. The component D2 can be placed - depending on the range of contrast voltage wanted - either as a zener diode or as a resistor.



position of R6,R7,R8 and D2 on MIO2L112

Here is a short extract of the connection diagram to explain the generation of the contrast voltage.



The current I is controlled by a MAX749. This device offers on the PC/104-MIO2 a current range from 6,6µA to 20µA. Due to this fact the voltage Ua is defined by Ohms Law : Ua = (R6 + D2)*I.

To generate a negative contrast voltage VEE, the resistor R8 is not to be placed and resistor R7 is fixed to 120R. In this case VEE results directly from I*(R7+R6+D2). In most cases R7 is very small compared with R6 and can be neglected.

Example 1: You have to generate a medium contrast voltage VEE = -8V.

The medium current -Imid from the MAX749 is : -Imid = (Imin+Imax)/2 = 13,3 µA

Therefore the whole resistor R consisting of R7 + R6 + D2 must be R = VEE/-Im = 601K.

So you have to place R7 = 120R, R6 = 1K and D2 = 600K. R8 not used.

In this case D2 is placed as resistor.

But note : With this configuration VEE ranges from Imin*R = -3,97V to Imax*R = -12V, this may damage a LCD with different contrast voltage range.

Example 2:

The necessary contrast voltage range of VEE is -7,8V to -8,2V.

The minimum contrast voltage is defined by VEEmin = Imin*R

The maximum contrast voltage has to be restricted with a zener diode.

Therefore D2 is placed as zener diode Z8,2 and R7 still 120R. R8 not used.

The resistor R6 = VEEmin/Imin = 1,18M will generate the minimum VEE at current Imin.

Example 3:

Generating a positive contrast voltage.

This examble shows PC/104-MIO2 and MIO3 state of delivery :

R6=120R, R7=4k7, R8=1k und D2=1M

Uamax=1000120R*(-6,66)µA=-6,66V and Uamin=1000120R*(-20)µA=-20V

i.e. : at the anode D1 will be max. -6.66V and min. -20V

The resistors R7 and R8 generate VEE :

VEEmin = VCC-(Ue*R8/(R7+R8))=5V-(25V*1k/5k7)=0,61V with Ue = VCC-(-20V)

VEEmax = VCC-(Ue*R8/(R7+R8))=5V-(11V*1k/5k7)=3.07V with Ue = VCC-(-6.6V)

The current Iemax=Uemax/(R7+R8)=25V/5k7=4mA can flow through R7 und R8

Note : With R7 not used (infinit) gets VEE maximum = 5V and can´t be controlled anymore, while reducing R7 increases the control range but forces VEEmax to decrease.

With R8 = 1K and R7 = 470R instaed of 4k7;

VEEmin=-0,32V and VEEmax=2,65V

Attention! To generate any contrast voltage that can be controlled by MAX749 the MIO2 Set-Utility item *Enable LCD Support* under menu *LCD Settings* has to be activated, otherwise 4.1V will always occure at VEE pin!

The actual results when measuring the contrast voltage will vary from the mathematic results calculated following these examples, because of device tolerances and dependace on the load connected to VEE. These calculations will help to achieve the first aproach to the definite values of the needed resistors.

Programming examples for character LCD

This chapter contains two little programming examples in TURBO PASCAL 6.0 format concerning outputs to the character LCD. The first example shows, how outputs at the DOS-prompt, that usually appear on the CRT monitor, will effect the character display. The second example is using the display specific BIOS calls, which are described in the programmer´s guide of this manual.

Both examples will be available as source code and as EXE-file on the utility disk.

example 1)

```
program LCDCHAR (input,output);


uses dos,crt;


var LINE,CHARLINES : byte;
    PRESS : char;


begin
  directvideo:=false;                    (* turn off direct video display
*)
  clrscr;
  write('How many lines has your character display (1-4) ? ');
  readln (CHARLINES);
  if (CHARLINES < 1) or (CHARLINES > 4) then CHARLINES := 1;
  LINE := 0;
  clrscr;
  repeat                                                  (* output loop
*)
    LINE := LINE + 1; (* the first lines are displayed on both CRT and LCD
*)
    if LINE <= CHARLINES                        (* the others only on CRT
*)
      then writeln('Line ',LINE,': visible on CRT and LCD')
      else
        if LINE <= 24
          then writeln('Line ',LINE,': visible on CRT only')
          else writeln('LINE ',LINE,': scrolling only visible on CRT');
    PRESS := readkey;
  until LINE = 30;      (* more then 24 lines causes scrolling only on CRT
*)
end.
```

example 2)

```
program LCDTEST (input,output);

uses dos,crt;

const CLEARSCR = $80; CURSOR   = $81; POSITION = $82; SWITCH   = $84;
      WRITECHAR= $8E; ON       = $01; OFF      = $00;

var AHIGH,ALOW,CHIGH,DHIGH,DLOW : byte;
    R : Registers;
    DUMMY : char;

begin
  directvideo:=false;                    (* turn off direct video display
*)
  clrscr;
  writeln ('Testing character display now: ');
  writeln ('');
  R.AH:=SWITCH; R.AL:=OFF; Intr($10,R);
  writeln ('Display should be off now');
  DUMMY := readkey;
  R.AH:=SWITCH; R.AL:=ON; Intr($10,R);
  writeln ('Display should be on now');
  DUMMY := readkey;
  R.AH:=CLEARSCR; R.AL:=$FF; Intr($10,R);
  writeln ('Cursor should be home now');
  DUMMY := readkey;
  R.AH:=CLEARSCR; R.AL:=$00; Intr($10,R);
  writeln ('Screen should be clear now');
  DUMMY := readkey;
  R.AH:=POSITION; R.DH:=$01; R.DL:=$05; Intr($10,R);
  writeln ('Cursor should be at line 2 / row 6 now');
  DUMMY := readkey;
  R.AH:=WRITECHAR; R.AL:=ORD('H'); Intr($10,R);
  R.AH:=WRITECHAR; R.AL:=ORD('A'); Intr($10,R);
  R.AH:=WRITECHAR; R.AL:=ORD('L'); Intr($10,R);
  R.AH:=WRITECHAR; R.AL:=ORD('L'); Intr($10,R);
  R.AH:=WRITECHAR; R.AL:=ORD('O'); Intr($10,R);
  writeln ('HALLO should appear on the display');
  DUMMY := readkey;
  writeln (' ');
  writeln ('End of test.');
end.
```

Matrix Touch Controller (MIO2 only)

The MIO2 is able to control a 8*10 (80 keys) Matrix Keyboard or Matrix Touch Screen in addition and fully parallel to the standard keyboard. The customer needs no additional driver software to support this feature, since the matrix support is fully integrated in the onboard BIOS. The user can configure the matrix interface with the MIO2SET utility program. This program allows to assign a scan code or extended scan codes to any crosspoint on the matrix. Refer to section ”Matrix keyboard setup” for details.

Any matrix keyboard may be connected to 26-pin connector X7.

Matrix Connector Pinout and Signal Explanation

Pin Name	Pin Description
GND	Ground
RA0-RA4	Scan inputs
/MCLR	Reserved for factory testing. Don’t connect anything.
Y0-Y7	Scan outputs
RB0-RB4	Scan inputs
RB5-RB7	Reserved. Don’t connect anything.
OC0, OC1	General purpose open collector outputs.
VCC	+5V

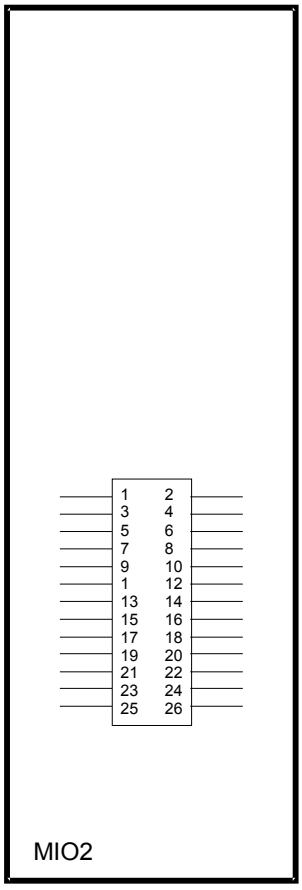
Name	Pin	Pin	Name
GND	1	2	Y0
RA0	3	4	Y1
RA1	5	6	Y2
RA2	7	8	Y3
RA3	9	10	Y4
RA4	11	12	Y5
RB0	13	14	Y6
RB1	15	16	Y7
RB2	17	18	/MCLR
RB3	19	20	OC0
RB4	21	22	OC1
RB5	23	24	RB7
RB6	25	26	VCC

To connect a matrix keyboard, wire outputs Y0-Y7 with matrix lines and inputs RA0-RA4, RB0-RB4 with matrix columns. Note that only one of the Y0-Y7 outputs is pulled low at a time when scanning. The other seven outputs remain high. Each connection between a line and a column generates an internal scan code which is transferred to the system. The BIOS converts this internal scan code to a keyboard scan code using the matrix decoder table. If you want to learn more about user defined scan codes, refer to section ”Matrix keyboard setup”.

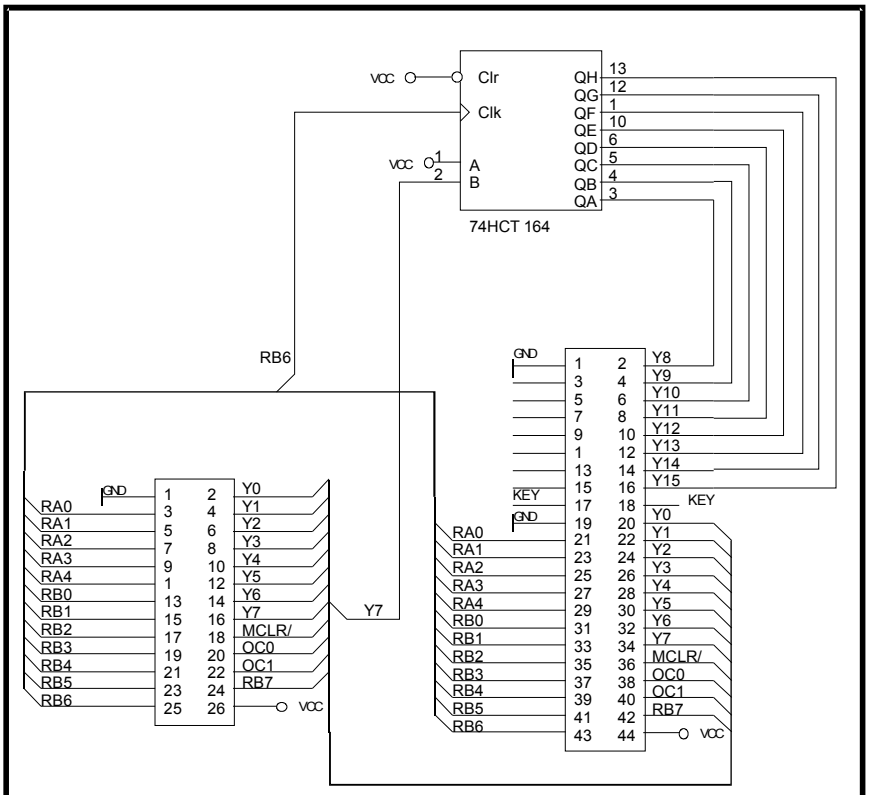
Standard and Enhanced Matrix Keyboards

Any Matrix with formats up to 8*10 crossings may be connected to a MIO2. In this case, the Pins Y0 to Y7 are used as outputs (active low), RA0 to RA4 and RB0 to RB4 are used as inputs.

If you need a larger matrix, there is a possibility to enhance the matrix to 16*10 crossings. To achieve this, you will need some external circuitry, which is shown in the following diagram:

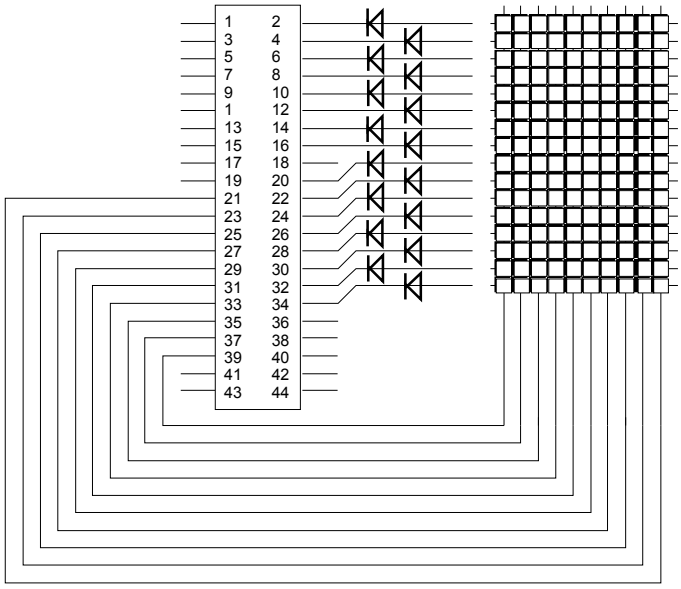


MIO2



Adapterboard

Matrix



Matrix-Keyboard

Digital and Analog I/O

Overview

All digital and analog inputs and outputs are available via the connector X12. The pinout for connector X12 is shown in the following table

Name	Pin	Pin	Name
AGND	1	2	ADC0
AGND	3	4	ADC1
AGND	5	6	ADC2
AGND	7	8	ADC3
Sync	9	10	ADC4
I²C Clock	11	12	ADC5
I²C Data	13	14	ADC6
GND	15	16	ADC7
OC0	17	18	In0
OC1	19	20	In1
OC2	21	22	In2
OC3	23	24	In3
OC4	25	26	In4
OC5	27	28	In5
OC6	29	30	In6
OC7	31	32	In7
In9	33	34	In8
In11	35	36	In10
In13	37	38	In12
VCC	39	40	VCC

Open Collector Outputs

There are eight general purpose open collector outputs available on the digital and analog I/O connector, named OC0 to OC7. The outputs are driven by an ULN2003D. This offers you a maximum collector-emitter voltage of 30V and the capability of sinking currents of 250mA (continuous, one output). Nevertheless, we recommend that the total emmitter current of all outputs should not exceed 500mA. After reset, all outputs are in high state. There is no onboard pull-up resistor, so an external resistor has to be used.

Your software may access the eight outputs by using a JIDA function call. Refer to the ”Programmer’s Guide” for details.

Digital Inputs

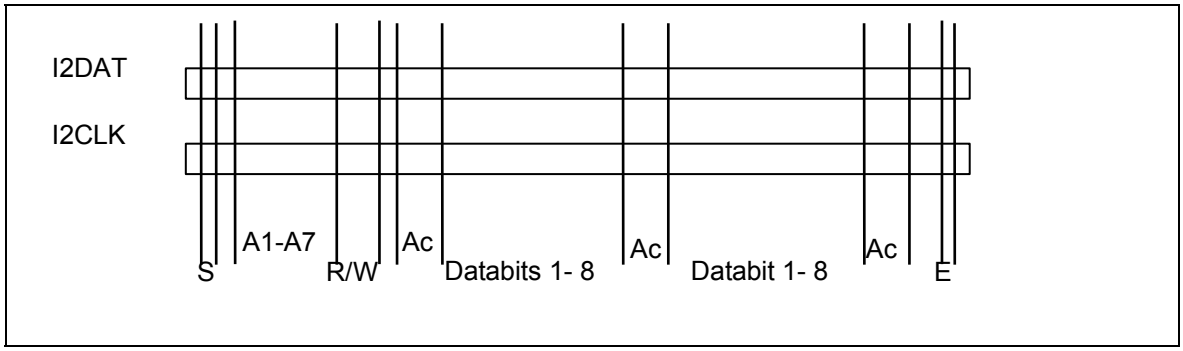
There are 14 digital inputs, named In0 to In13, available to the user. Each input is protected by a 1k series resistor and a clamp diode pair DBAV99. This allows input voltages ranging from -30V to +30V. The switching level is TTL compatible. The inputs are read via JIDA BIOS calls. Refer to the ”Programmer’s Guide” for details.

I²C Bus

Introduction to I²C-bus

The inter-IC bus (I²C) is a two-wired serial bus and provides a sort of small area network between the circuits of one system and between different systems. Any device with build-in I²C bus interface can be connected to the system by simply clipping it to the I²C bus. It consists of two bidirectional lines for serial data (I2DAT) and serial clock (I2CLK). Usually every device connected can be master or slave, so there is no central master. A device addressed as a slave during one data transfer could possibly be the master for the next data transfer. Devices are also free to transmit or receive data during a transfer. The inherent synchronization process in connection with the wired AND technique allows fast devices to communicate with slower ones.

For each data bit transferred, one clock pulse has to be generated. The data on the I2DAT line must be stable during the high period of the clock. The data lines state can only change during I2CLK line is low. Data transfer is entered by a start- and ended by a stop-condition. A high to low transition of the I2DAT line while the I2CLK is high signals the start condition and a low to high transition while I2CLK is high indicates the stop condition. Data transfer follows the format below:



After the start condition (S) the slave address byte is sent. This byte consists of seven address bits (A1-A7) and one direction bit (R/W) with low level indicating a transmission (WRITE) and high level indicating a request for data (READ).

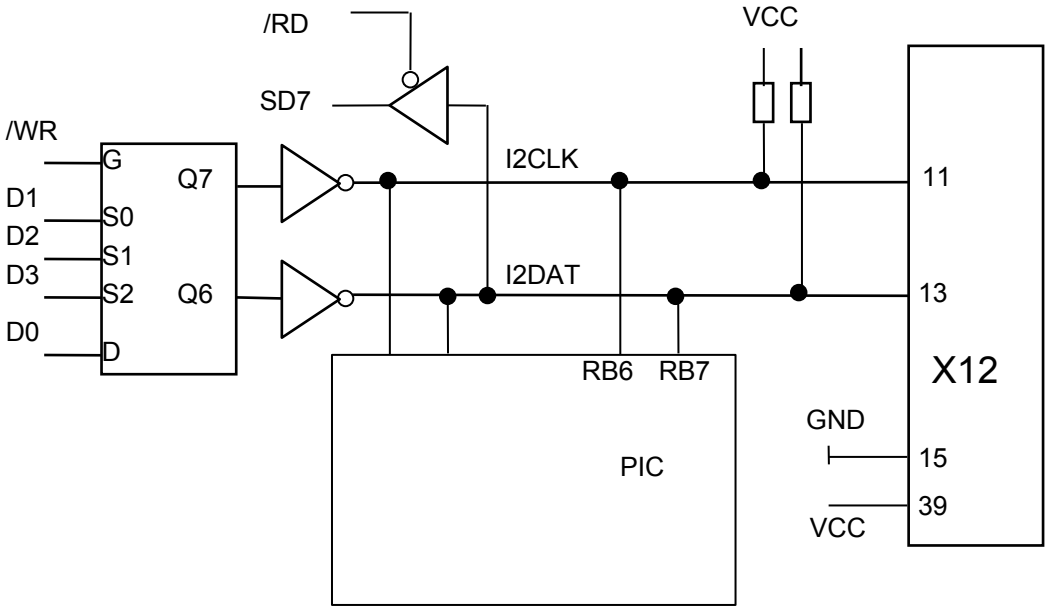
After the addressing of a slave device the masters next clock pulse is used for acknowledgement (Ac). During this acknowledge pulse the I2DAT line has to be pulled down to low by the receiving device. A data transfer is always terminated by a stop condition (E) generated by the master. However, if the master wants to communicate with another device on the bus, it generates another start condition to address another slave without the necessity of first generating a stop condition.

This was only a short summary concerning the I²C bus. For detailed information (e.g. timing problems, characteristics of devices) join I²C bus specifications, data books and specialized textbooks.

I²C bus on MIO2/MIO3

The I²C bus interface on MIO2/MIO3 has to be realized by the customer via software which drives the two lines I2DAT and I2CLK following the I²C bus specifications. But note, that the onboard drivers of these two lines are inverting. Once designed, it is possible to connect, read and write additional I²C devices to the MIO2/MIO3 by using connector X12 Pins 11,13,15 and 39. The two already connected onboard devices PIC and EEPROM are vital to the boards and shouldn´t be read or written to, because otherwise data corruption may cause system failure. Use first I/O address to write to the I2DAT- and I2CLK line and second I/O address to read the I2DAT line. This kind of interface does not support external masters.

The follwing drawing shows the bus interface and the onboard devices connected to the I²C bus on the MIO2/MIO3.



I/O address to generate /WR	:	depending on BIOS location	
		C000,D000	: 018Fh
		C400,D400	: 0210h
		C800,D800	: 0300h
		CC00,DC00	: 009Fh
I/O address to generate /RD	:	depending on BIOS location	
		C000,D000	: 118Fh
		C400,D400	: 1210h
		C800,D800	: 1300h
		CC00,DC00	: 109Fh

Device address of EEPROM	:	1010000
Device address of PIC16C84	:	0101101

Example: Guess you´re writing a software utility and want the I2DAT line to be pulled HIGH. Your BIOS location is jumpered to D800. You have to use I/O-port 0300h and set the data lines D0 to D3, with D0=0 (gets HIGH because of inversion) holding the output level for I2DAT and D3 to D1=110 choosing the right output line Q6.

The corresponding OUT instruction would be : OUT 0300h,0Ch

Several OUTs and INs will be necessary to address one I^2C bus device and write or read a byte, because the start and stop conditions and every single bit have to be set separately as well as the I2CLK line.

Analog Inputs (optional)

Analog I/O is an optional feature of the MIO2/MIO3, depending on wether the AD converter is plugged in or not.

Features of the AD converter

The AD converter used with the PC104/MIO2 and MIO3 is a LM12458 by NATIONAL Semiconductors and offers a highly integrated data acquisition system operating at just +5 V. Up to 32 consecutive conversions, using two´s complement, can store 13 bit (12 bit + sign) conversion results in a 32-word FIFO buffer. The internal 8 -word RAM can store up to eight instructions defining the conversion sequence for the eight-input multiplexer. The LM12458 can also operate with 9 bit (8 bit + sign) and in a supervising watchdog mode which compares the input value against two user-programmable values. Programmable conversion rates and data acquisition times are possible through the use of internal clock-driven timers. The negative and positive reference voltage pins define an input range from 0 V to +2,5 V, because V_{REF-} is connected to ground and the source of V_{REF+} is the internal +2,5 V bandgap reference. All registers, the RAM and the FIFO buffer are directly addressable via an eight-bit databus. A summary of the main features concerning analog inputs is shown in the table below:

resolution	12 bit + sign or 8 bit + sign (programmable)
input range	0 V to +2,5 V
conversion times	8,8 µs (max) for 13 bit 4,2 µs (max) for 9 bit
sampling rate	1/(8,8 µs +t_{TC}) for 13 bit (worst case[1]) 1/(4,2 µs + t_{TC}) for 9 bit (worst case[1]) with t_{TC} = time of transmission and calculation
through-put rate	88K-samples/s for 13 bit (min)
integral linearity error	+/- 1 LSB (max)
features	8 single ended or 4 differential inputs internal sample and hold +2,5 V build-in bandgap reference instruction RAM and event sequencer 8-channel multiplexer 32-word conversion FIFO programmable acquisition times programmable conversion rates self-calibration diagnostic mode

[1)] worst case : only one instruction is executed per loop and
the FIFO is read after one conversion

Two different calibration methodes are available for the LM12458. The first one compensates only for offset-voltage, while the second one additionaly compensates the linearity error. Once calibrated the correction coefficients are used to reduce their corresponding errors in the background during every conversion.

The analog input multiplexer may be configured for any combination of single ended or fully differential operation. In the single ended mode the multiplexer channel is referred to ground, while fully differential channels are formed by pairing any two input channels together.

32 consecutive conversions can be completed and stored in the FIFO buffer without any microprocessor intervention. The microprocessor can access to the FIFO at any time or may wait for an interrupt from the FIFO. This interrupt will be generated, when the FIFO is full or after any user-programmed number of conversion results have been stored.

Programming the AD converter

The AD converter LM12458 on the MIO2/MIO3 is programmed via memory mapped I/O,with the memory address of the ADC depending on the jumper setting that determines BIOS-location and I/O-address (see section ”jumper settings”). The ADC´s register access range location starts at an address 200h above of the jumpered BIOS-location. Because of the overlapping between BIOS-area and ADC access area it is necessary to tell both envolved devices which one is ment, otherwise a planed access to the ADC would end in an access to the MIO2/MIO3 BIOS. Therefore JUMP designed a way to enable the ADC and disable the BIOS (or the other way) by writing to one of the two I/O-ports of the MIO2/MIO3. The tables below shows how to enable/disable the ADC and access the registers.

BIOS location	ADC location	I/O address
C000:0000h	C200:0000h	018Fh
C400:0000h	C600:0000h	0210h
C800:0000h	CA00:0000h	0300h
CC00:0000h	CE00:0000h	009Fh
D000:0000h	D200:0000h	018Fh
D400:0000h	D600:0000h	0210h
D800:0000h	DA00:0000h	0300h
DC00:0000h	DE00:0000h	009Fh

Enable ADC / Disable BIOS	writing 05h to I/O address
Disable ADC / Enable BIOS	writing 04h to I/O address

The ADC offers 14 registers of 16 bit, which can be accessed to through an eight-bit data bus. This means, every register access consists of two reads/writes, with one addressing the lower and the other the higher byte of the 16-bit register in random order. The first eight registers are holding the sequencial instructions for the conversions. The ninth register is the configuration register followed by interrupt enable, interrupt status and timer register. Register number 13 is used to access the conversion FIFO buffer, while the last one gives information about the watchdog limits. The addresses of these user programmable registers are located at an offset added to the ADC-location (see table above). The bits of the configuration register are very important for the correct working of the ADC, that is why they have to be set explicitly and with care. The table below shows the available registers and their offsets.

register	type	offset	
		lower byte	higher byte
instruction RAM (??? = nr. of instruction)	R/W	0???0b	0???1b
configuration register	R/W	10000b	10001b
interrupt enable register	R/W	10010b	10011b
interrupt status register	R	10100b	10101b
timer register	R/W	10110b	10111b
conversion FIFO	R	11000b	11001b
limit status register	R	11010b	11011b

Example: Guess you want to read the configuration register of the ADC.

Your MIO2/MIO3 BIOS location is jumpered to D000h, which results in an ADC register area

starting at D200h. The offset of the configuration register is 10000b (10h) for the lower byte

and 10001b (11h) for the higher byte. Now you have to generate one memory mapped I/O

access to D200:0010h (for the lower byte) and one to D200:0011h (for the higher byte) to get

the whole configuration settings of the ADC.

As JUMP is a producer of industrial PC hardware our information about the ADC programming of the LM12458 on the PC/104-MIO2/MIO3 will end here. For detailed information about the register programming and the purpose of every single bit, we refer to the NATIONAL Semiconductor ”DATA ACQUISITION DATABOOK” edition 1995 or any other data sheet released for the LM12458 by NATIONAL. Nevertheless JUMP offers a small test program for the MIO2/MIO3 called MIO2ADC, which will be part of the utility disk holding the firmware revision 1.3 or higher. This test software is written in BORLAND´s ”Turbo PASCAL 6.0” and will be available as MIO2ADC.EXE file and as source code MIO2ADC.PAS .The source code includes several commentary information to help a programmer getting started with the ADC LM12458 but doesn´t claim to be a programmers tool.

RS485

A RS485 interface is available on connector X10, which can be used instead of the second serial interface COM B. The configuration for RS485 is made by pulling the /E485 signal to lOW (connect to GND), which causes the RS232 interface of connector X11 to be disabled. The pin assignment of the connector X10 is shown in the table below:

Name	Description	Pin	Pin	Description	Name
A2	channel A transmitter	1	2	channel A receiver	A1
B2	channel B transmitter	3	4	channel B receiver	B1
Term2	termination transmitter	5	6	termination receiver	Term1
/E485	enable RS485	7	8	ground	GND
VCC	+ 5V	9	10	not connected	nc.

The RS485 interface can be used in full-duplex mode for a point to point connection. If a RS485 bus system is used - i.e. several transmitters are connected with another - this interface can only support half-duplex mode. For half-duplex operation only the lines A1 and B1 can be connected to the bus system. To enable the half-duplex transmitter pull the /RTS signal of the COM B to HIGH (by software). To switch to half duplex receiver pull the /RTS signal of the COM B LOW (by software).

In some RS485 systems there has to be a termination with a 120R resistor at the end of the RS485 bus. Therefore the interface offers the possibility to connect these termination resistors, which are allready mounted on the MIO2/MIO3, by shortening Pin 5 with Pin 3 (transmitter) and Pin 6 with Pin 4 (receiver).

Programmer's Guide

The JIDA Interface

JIDA (Jump Intelligent Device Architecture) Overview

Sometimes it´s desirable to get some information about the boards integrated in the system. JUMP decided to equip JUMP boards with extension BIOSes to offer the user a possibility to get board information via JIDA function accesses. JIDA functions are called via Interrupt 15h with AH=EAh, AL=function number, DX=4648h (security word), CL=board number (starting with 1).

The interrupt will return with CL≠0, if a board with the number specified in CL does not exist. CL will be equal to 0 if the board number exists. In this case, the content of DX is used to determine, if the operation was successful. DX=6B6Fh indicates successful operation, any other value indicates an error.

To get information about the installed boards following the JIDA standard, the following procedure is recommended:

Call ”Get Device ID” with CL=1. The name of the first device installed will be returned. If result was ”Board exists” (CL=0), increment CL and call ”Get Device ID” again. Repeat until result is ”Board not present” (CL#0). You now know the names of all boards within your system that follow the JIDA standard. More information about a specific board may then be obtained by calling the appropriate inquiry function with the board's number in CL.

Warning! Association between board and board number may change due to configuration changes. Do **not** rely on any association between board and board number. Instead, always use the procedure described in the preceding paragraph first, to determine the association between board and board number.

The source of a Turbo-Pascal unit showing JIDA access is contained on the disk that is shipped with this manual.

Get Manufacturer ID	Int 15h	
Input:	AX = EA00h	DX = 4648h
	CL = Board number (1=first board a.s.o.)	
	ES:BX = Pointer to destination data area	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Error
Description:	If CL=0 and DX=6B6Fh, then 4 Byte manufacturer ID were copied to the area pointed to by ES:BX By default, the result will be "□□□□". Note: There is no ending zero byte. Function must be implemented on every device supporting the JIDA.	

Get Device ID	Int 15h	
Input:	AX = EA01h	DX = 4648h
	CL = Board number	
	ES:BX = Pointer to destination data area	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Error
Description:	If CL=0 and DX=6B6Fh, then 7 Byte device ID were copied to area pointed to by ES:BX By default, the result will be "□□□□□□□". Note: There is an ending blank but no ending zero byte. Function must be implemented on every device supporting the JIDA.	

Get Manufacturing Date	Int 15h	
Input:	AX = EA02h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BX = Manufacturing date	
Description	If CL=0 and DX=6B6Fh, then BX = Manufacturing date. Date format is the same as used for DOS files: Bit0..4: Day Bit5..8: Month Bit9..15: Years since 1980	

Get Serial Number	Int 15h	
Input:	AX = EA03h	DX = 4648h
	CL = Board number	
	ES:BX = Pointer to destination data area	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
Description:	If CL=0 and DX=6B6Fh, then 10 Byte serial number were copied to area pointed to by ES:BX The result is different for each single MIO2/MIO3. Note: There is no ending zero byte.	

Get Hardware Revision	Int 15h	
Input:	AX = EA04h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BH = Major revision number	
	BL = Minor revision number	

Get Firmware Revision	Int 15h	
Input:	AX = EA05h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BH = Major revision number	
	BL = Minor revision number	

Get Last Repair Date	Int 15h	
Input:	AX = EA06h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BX = Last repair date.	
Description:	If CL=0 and DX=6B6Fh, then BX = Last repair date. For date format see function ”Get Manufacturing Date”.	
	If board was never repaired, result will be equal or less to manufacturing date.	

Read Running Time Meter	Int 15h	(not implemented with MIO2/3)
Input:	AX = EA07h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BX = Running time (hours)	
	CH = Overflow counter	

Read Boot Counter	Int 15h	(not implemented with MIO2/3)
Input:	AX = EA08h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BX = Boot counter	

Get JIDA revision level	Int 15h		(implemented with BIOS R111 and later)
Input:	AX	= EA09h	DX = 4648h
	CL	= Board number	
Output:	CL=0: Board present		DX=6B6Fh: Function successful
	CL≠0: Board not present		DX≠6B6Fh: Fn. not implemented
	BH = Major revision number		
	BL = Minor revision number		

Get Device Subversion	Int 15h		(implemented with BIOS R111 and later)
Input:	AX	= EA0Ah	DX = 4648h
	CL	= Board number	
	ES:BX	= Pointer to destination data area	
Output:	CL=0: Board present		DX=6B6Fh: Function successful
	CL≠0: Board not present		DX≠6B6Fh: Error
Description:	If CL=0 and DX=6B6Fh, then 8 Byte device Subversion ID were copied to area pointed to by ES:BX Possible results are □□□□I=PC/104-MIO3 □□□□=customer version □□□□=PC/104-MIO2 Note: There is no ending zero byte.		

Get Numeric Device ID	Int 15h		(implemented with BIOS R111 and later)
Input:	AX	= EA0Bh	DX = 4648h
	CL	= Board number	
Output:	CL=0: Board present		DX=6B6Fh: Function successful
	CL≠0: Board not present		DX≠6B6Fh: Fn. not implemented
	BX=0202h identifies the MIO2/MIO3		

Get Contrast setting	Int 15h	
Input:	AX = EA20h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	CH = Actual contrast (value range 0..63)	

Set Contrast	Int 15h	
Input:	AX = EA21h	DX = 4648h
	CL = Board number	
	CH = New contrast value	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
Description:	Valid value range for contrast is 0..63. Other values will be ignored.	

Disable DC/DC Converter	Int 15h	
Input:	AX = EA22h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented

Enable DC/DC Converter	Int 15h	
Input:	AX = EA23h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented

Disable Backlight *	Int 15h	(implemented with BIOS R114 and later)
Input:	AX = EA2Eh	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented

Enable Backlight *	Int 15h	(implemented with BIOS R114 and later)
Input:	AX = EA2Fh	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented

* will be available with BIOS R114 and later versions

Get Matrix Translation Table *	Int 15h		
Input:	AX	= EA30h	DX = 4648h
	CL	= Board number	
	ES:BX	= Pointer to destination data area	
Output:	CL=0: Board present		DX=6B6Fh: Function successful
	CL≠0: Board not present		DX≠6B6Fh: Fn. not implemented
Description:	160 Byte matrix keyboard translation table will be copied to area pointed to by ES:BX		

Set Matrix Translation Table	Int 15h		
Input:	AX	= EA31h	DX = 4648h
	CL	= Board number	
	ES:BX	= Pointer to new translation table (holds 160 bytes)	
Output:	CL=0: Board present		DX=6B6Fh: Function successful
	CL≠0: Board not present		DX≠6B6Fh: Fn. not implemented
Description:	New matrix keyboard translation table will be copied from area pointed to by ES:BX		

Get Matrix Translation Entry	Int 15h		
Input:	AX	= EA32h	DX = 4648h
	CL	= Board number	
	BH	= Matrix line (0..15 allowed)	
	BL	= Matrix row (0..9 allowed)	
Output:	CL=0: Board present		DX=6B6Fh: Function successful
	CL≠0: Board not present		DX≠6B6Fh: Fn. not implemented
	CH = Table entry (undefined if BH or BL are invalid)		

Set Matrix Translation Entry	Int 15h		
Input:	AX	= EA33h	DX = 4648h
	CL	= Board number	
	BH	= Matrix line (0..15 allowed)	
	BL	= Matrix row (0..9 allowed)	
	CH	= New Entry	
Output:	CL=0: Board present		DX=6B6Fh: Function successful
	CL≠0: Board not present		DX≠6B6Fh: Fn. not implemented

* due to a bug in BIOS revision 1.0 to 1.3 only the first 80 bytes will be copied, will be fixed in BIOS revision 1.4

Read User Byte from FRAM	Int 15h	
Input:	AX = EA40h	DX = 4648h
	CL = Board number	
	BH = Number of byte to read (0..FFh allowed)	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BL = Value read	

Write User Byte to FRAM	Int 15h	
Input:	AX = EA41h	DX = 4648h
	CL = Board number	
	BH = Number of byte to write (0..FFh allowed)	
	BL = Value to write	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented

Get Number of User Bytes available in FRAM	Int 15h	
Input:	AX = EA42h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BL = Number of user bytes available (255 = FFh)	

Read OC Output State	Int 15h			
Input:	AX = EA50h			DX = 4648h
	CL = Board number			
Output:	CL=0: Board present			DX=6B6Fh: Function successful
	CL≠0: Board not present			DX≠6B6Fh: Fn. not implemented
	CH = Actual output state			
Description:	Data format:	CH.0=OC0	CH.1=OC4	CH.2=OC2
		CH.3=OC6	CH.4=OC1	CH.5=OC5
		CH.6=OC3	CH.7=OC7	

Switch OC Outputs	Int 15h			
Input:	AX = EA51h			DX = 4648h
	CL = Board number			
	CH = New output state			
Output:	CL=0: Board present			DX=6B6Fh: Function successful
	CL≠0: Board not present			DX≠6B6Fh: Fn. not implemented
Description:	Data format:	CH.0=OC0	CH.1=OC4	CH.2=OC2
		CH.3=OC6	CH.4=OC1	CH.5=OC5
		CH.6=OC3	CH.7=OC7	

Read Inputs	Int 15h	(implemented with BIOS R111 and later)
Input:	AX = EA52h	DX = 4648h
	CL = Board number	
Output:	CL=0: Board present	DX=6B6Fh: Function successful
	CL≠0: Board not present	DX≠6B6Fh: Fn. not implemented
	BX= State of Input Lines	
Description:	Data format: BX.0=In0, BX.1=In1 ...	

MIO2/MIO3 Display Specific BIOS Calls

The BIOS of the MIO2/MIO3 supports two groups of display specific BIOS calls:

- The first group is called ”direct to display” functions and is always available. Every command given by a ”direct to display” fuction is immediately transferred to the LC character display and does not affect the screen of a graphic card in the system. ”Direct to display” functions must have bit 7 set in register AH when called.
- The second group are the ”copy to display” functions. They must be enabled within the MIO2/MIO3 setup program. ”Copy to display” functions use standard VGA BIOS calls, but they do not only direct the output to the screen but also to the display.

”Direct to Display” Functions

Clear Screen	Int 10h		
Input:	AH=80h	AL=0xxxxxxxb or 1xxxxxxxb	AL.7=0 ClearScreen AL.7=1 CursorHome
Output:	-		
Result on Display:	If AL.7=0 the display is beeing cleared and cursor is set to home position. If AL.7=1 the cursor is set to home position.		

Set Cursor Type	Int 10h	
Input:	AH=81h	CH=Cursor Mode
Output:	-	
Result on Display:	CH=00h enables underline Cursor CH=20h disables the cursor CH=40h enables blinking character cursor	

Set Cursor Position	Int 10h	
Input:	AH=82h	DH=Cursor Line DL=Cursor Column
Output:	-	
Result on Display:	Positions the cursor on the display. Valid Lines are 0-3, valid columns are 0-19. If line or column are invalid, the cursor position will not be changed.	

Get Cursor Position	Int 10h	
Input:	AH=83h	
Output:	DH = Cursor Line	DL = Cursor Column
Result on Display:	-	

Enable/Disable Display	Int 10h	
Input:	AH=84h	AL=0 Disable Display AL=1 Enable Display
Output:	-	
Result on Display:	Display is enabled or disabled. The display content is not affected by disabling the display. Display content may be altered while display is off. If ”copy to display” mode is selected in setup, display is enabled by default. Else, display is disabled by default.	

Define User Character	Int 10h	
Input:	AH=85h AL=Character #	ES:DI=Pointer to 8 Bytes of definition data
Output:	-	
Result:	Defines one of the 8 user programmable characters. AL is the number of the character to define (valid values are 0-7). ES:DI points to an array of 8 bytes that define the pixels of the new character. To display one of the characters, use function ”Write Character” with ASCI-values 0-7.	

Write Character	Int 10h	
Input:	AH=8Eh AL=ASCII Code	
Output:	-	
Result on Display:	Writes character from AL to the screen. Cursor position is incremented. Beep, Backspace, LineFeed and CR are interpreted correctly.	

Write String	Int 10h	
Input:	AH=8Fh	ES:DI = Pointer to null-terminated string
Output:	-	
Result on Display:	Writes characters from the null-terminated string pointed to by ES:DI to the display. Cursor position is incremented. Beep, Backspace, LineFeed and CR are interpreted correctly.	

Send Raw Character	Int 10h	(will be implemented with BIOS R114)
Input:	AH=90h	AL=Character to send
Output:	-	
Result on Display:	Sends the raw data from AL directly to the display. Cursor position is not changed. No control codes are interpreted.	

Send Raw Control	Int 10h	(will be implemented with BIOS R114)
Input:	AH=91h	AL=Control Code to send
Output:	-	
Result on Display:	Sends the raw data from AL directly to the display’s control register. Cursor position is not changed. No codes are interpreted.	

"Copy to Display" Functions

Select Video Mode	Int 10h	
Input:	AH=00h	AL=Desired Mode
Output:	-	
Result on Screen:	The desired video mode is selected. If AL.7=0 the screen is beeing cleared.	
Result on Display:	If AL.7=0 the display is beeing cleared and cursor is set to home position. If AL.7=1 the cursor is set to home position.	

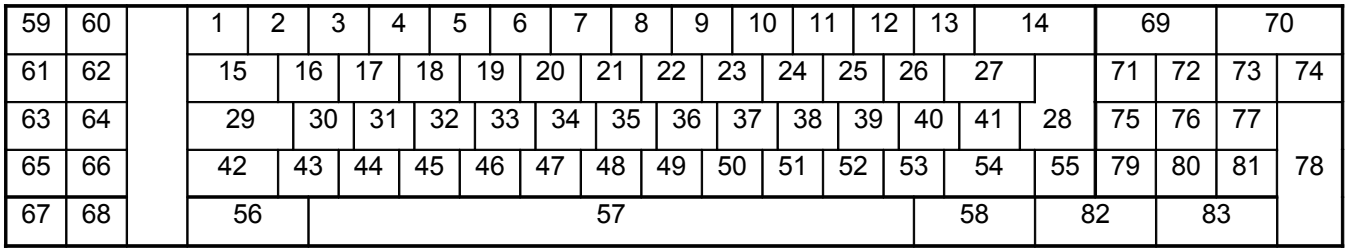
Set Cursor Type	Int 10h	
Input:	AH=01h	CH=Cursor Start Line CL=Cursor End Line
Output:	-	
Result on Screen:	Defines the start line and end line of the textmode cursor. CX=0607h selects standard cursor. CX=20xxh disables cursor.	
Result on Display:	CX=20xxh disables the cursor CX=40xxh enables blinking character cursor Any other value in CX enables underlined cursor.	

Set Cursor Position	Int 10h	
Input:	AH=02h BH=Video Page #	DH=Cursor Line DL=Cursor Column
Output:	-	
Result on Screen:	Positions the cursor on one of the available video pages. Valid lines are 0-24, valid columns are 0-79	
Result on Display:	Positions the cursor on the display if BH=0. Valid Lines are 0-3, valid columns are 0-19. If BH≠0 or line or column are invalid, the cursor position will not be changed.	

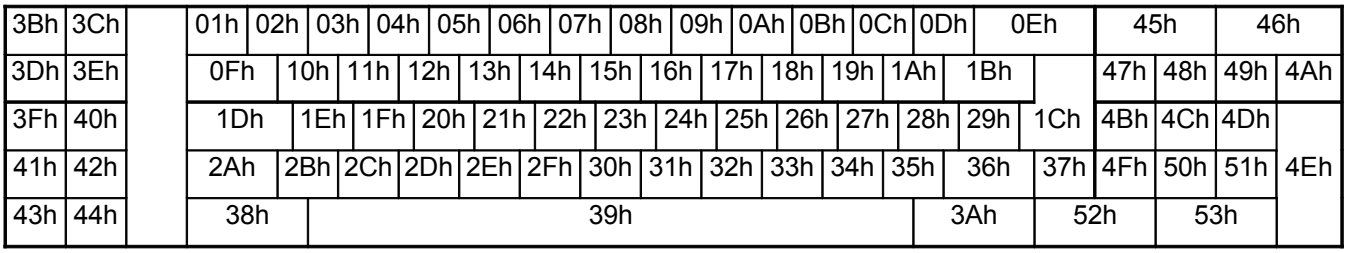
Write Character (& Attribute)	Int 10h		
Input:	AH=09h or AH=0Ah AL=ASCII Code	BH=Video Page # BL=Attribute or Color	CX=# of Repetitions
Output:	-		
Result on Screen:	Writes character(s) from AL to the screen. Cursor position is not changed.		
Result on Display:	Send character(s) if BH=0. CX is ignored. Cursor position is not changed.		

Write Character	Int 10h	
Input:	AH=0Eh AL=ASCII Code	BH=Video Page # BL=Color
Output:	-	
Result on Screen:	Writes character from AL to the screen. Cursor position is incremented. Beep, Backspace, LineFeed and CR are interpreted correctly.	
Result on Display:	Writes character from AL to the display. Cursor position is incremented. Beep, Backspace, LineFeed and CR are interpreted correctly.	

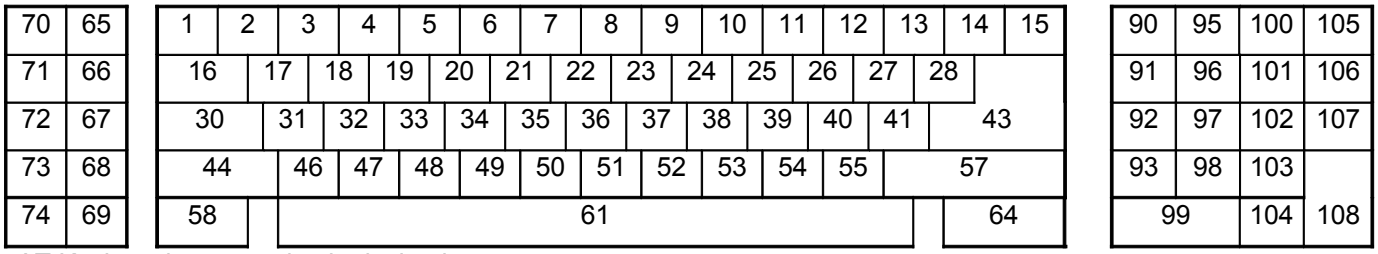
Scan Codes



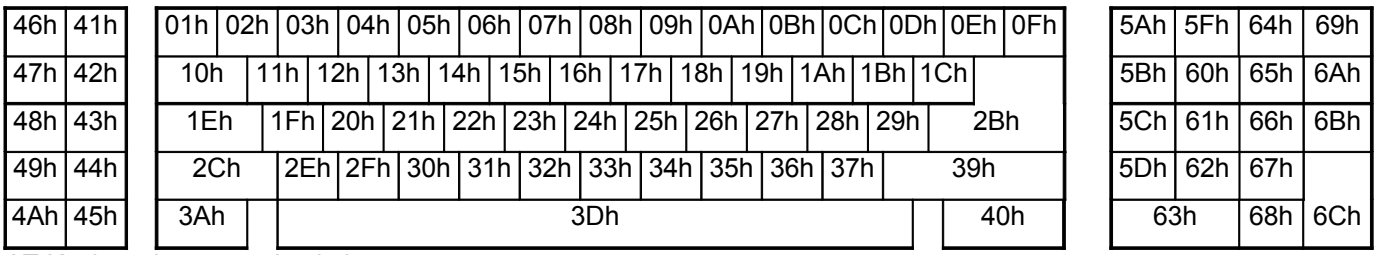
PC/XT Keyboard, scan codes in decimal



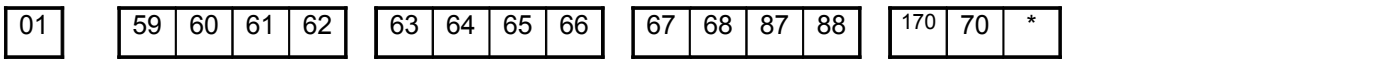
PC/XT Keyboard, scan codes in hex

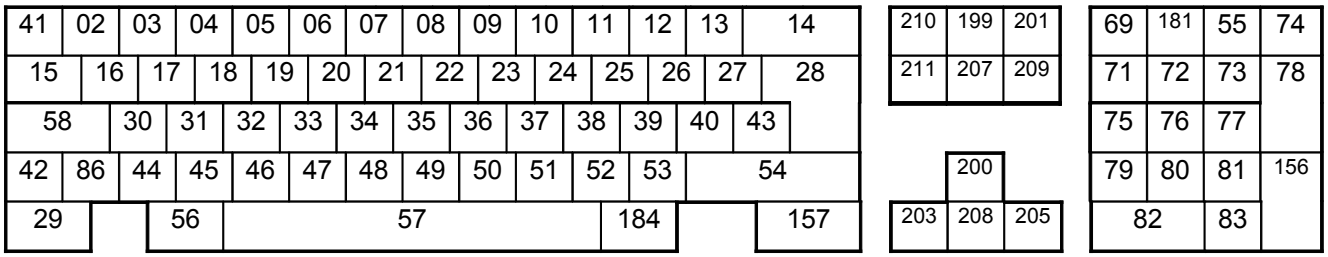


AT Keyboard, scan codes in decimal



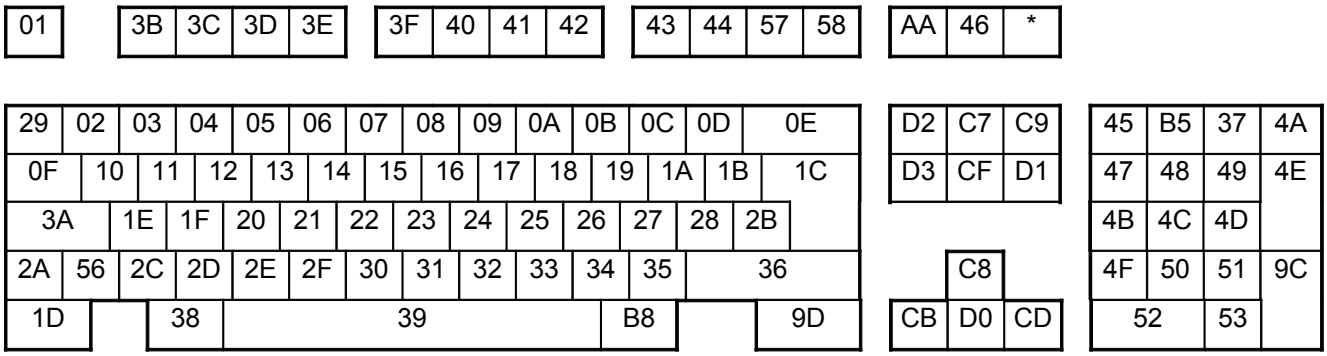
AT Keyboard, scan codes in hex





MF2 keyboard, scan codes in decimal

*) This key can not be used on a matrix keyboard



MF2 keyboard, scan codes in hex

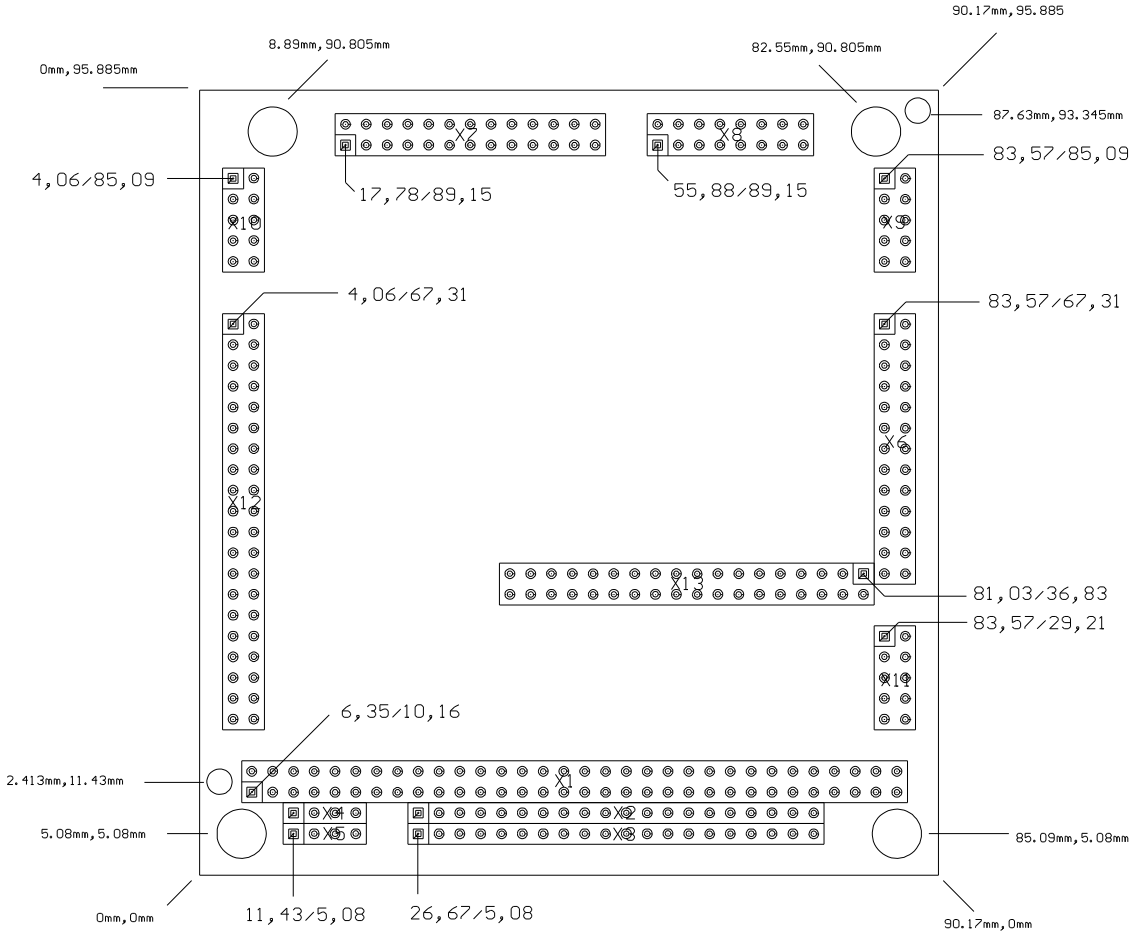
*) This key can not be used on a matrix keyboard

Specifications

Mechanical Specifications

Weight approximately 130 g without front bezel

Mechanical dimensions of the PC/104-MIO2/MIO3 board



Electrical Specifications

Supply voltage:	5V DC +/- 5%
Supply voltage ripple:	100 mV peak to peak 0 - 20 MHz
Supply current, typical:	100 mA at 25 °C (PC/104-MIO2 without ADC)
	85 mA at 25 °C (PC/104-MIO3 without ADC)

Environmental Specifications

Temperature

operating	0 °C to +60 °C see note (*1)
non-operating	-40 °C to +85 °C

Thermal gradient

operating	25 °C per hour
non-operating	40 °C per hour

Relative Humidity

operating	10 % - 90 % RH non-condensing
non-operating	5 % - 95 % RH non-condensing

Mechanical

Shock	50G/20ms square wave maximum
Vibration	1G/0-600Hz, dwell not to exceed

Altitude

operating	0 - 3000 m
non-operating	0 - 5000 m

(*1) The maximum operating temperature is the maximum measurable temperature on any spot of the modules surface. It is the users responsibility to keep this temperature within the above specification.

Available Utilities

- Configuration-diskette with program "MIO2SET.EXE". (Diskette is included in this Technical Manual)
- cables for COMA, COMB ; DSUB9 male connector to 10 pin header cable (orderno.: KAB-DSUB9-2)
- cable for printer ; DSUB25 female connector to 26 pin header cable (orderno.: KAB-DSUB25-1)

Revision History

File-name	last alteration	authors	alteration to previous version
MIO2D110.DOC	10.09.96	H. Fink H. Bruhn	no
MIO2D111.DOC	16.10.96	H. Bruhn	added scan codes for MF2 keyboard added further information about character displays
MIO2D112.DOC	21.11.96	H. Bruhn	changed default values of interrupt COM A to IRQ12 and COM B to IRQ15
MIO2D113.DOC	25.02.97	H. Bruhn	changed wrong information about JIDA calls READ OC OUTPUT STATE and SWITCH OC OUTPUT
MIO2D114.DOC	11.06.97	H. Bruhn	changed wrong information about RS485 connector (X11 to X10)
MIO2D115.DOC	16.06.97	H. Bruhn	added RS485 information
MIO2D116.DOC	09.02.98	J. Hagn	Layout revised, Filename corrected
MIO2D117.DOC	11.03.98	O. Thaler	Part No. entered