MicroPC

Technical Manual

taskit Rechnertechnik GmbH

Seelenbinderstr. 33
D-12555 Berlin
Germany

Tel +49 (30) 611295-0
Fax +49 (30) 611295-10
http://www.taskit.de

MicroPC

1. Introduction

The MicroPC can be used anywhere restricted energy and space requirements play a role, and where a PC-compatible solution is desired (e.g. for ease of programming).

Examples of potential applications:

- mobile or fixed data-recording equipment
- LCD terminals
- measurement and testing equipment
- alarm installations
- any simple (or more complex) automation task

The MicroPC can be programmed like a DOS PC with all of the usual DOS compilers, such as Borland or Microsoft C, Pascal and Basic, among others. Creating a program is thus very easy. Since the terminal program displays the PC's drives as drives of the MicroPC, the developer can directly start .EXE files on the host PC. Or the EXE file can be copied to the Flash disk and started from there. A starter kit is included for ease of implementation; it contains a developer board with a power supply and the necessary software.

The BIOS setup contains extensive options for memory configuration, serial interfaces and I/O ports. Additional serial interfaces are thus also supported by the BIOS. The I/O pins can be adjusted individually or in groups to input or output, in part also with interrupt function. Hardware initializations for peripherals can be defined in the BIOS Setup. In addition, the BIOS setup menu contains the functions for loading ROM-DOS or BIOS updates, as well as for saving to or deleting the Flash disk.

2. Overview of Technical Properties

2.1. CPU

- Intel 386EX embedded processor (extension of the 386SX)
- 25 MHz
- 3.3V operating voltage

2.2. Memory

- 2 MB flash memory with flash file system
- 512kB or 1MB static RAM
  maximum of 896kB DOS main memory (with 1MB RAM)
  RAM may be supported by external battery during power off
- 512 bytes serial EEPROM

2.3. Firmware

- PC-compatible BIOS, includes configuration menu in BIOS setup
- Datalight ROM-DOS (optional)

2.4. Interfaces

- two PC-compatible serial interfaces (8250-compatible), maximum of 781.25 kBaud (= 25 MHZ/32), or 115.2 kBaud as maximum PC-compatible Baud rate, LV-TTL level, configurable through IF module connection (IF232-9DIR, IF485) as RS-232 or RS-485
- PIF bus (5V-compatible)
- synchronous serial interface, up to 6.25 MBaud (= 25 MHz / 4)
- I²C bus
- Up to 32 digital I/O ports, programmable individually or in groups of four as input or output
- 6 IRQ lines externally available
- Real-time clock (RTC) with alarm output
- Timer clock, timer gate and timer output signals

Some of the various functions are realized by multiplexing connector pins; therefore not all functions may be used at the same time (see table in chapter 10).

2.5. Power Management

Energy consumption can be drastically reduced by the power management functions of the BIOS in many instances.

- Downclocking,
- Idle mode
- Power-down mode
- Deep power-down mode (stop mode)

Energy consumption:

- Active: 135 mA (at 25 MHz)
- Idle mode: 9 mA
- Power-down: 4 mA
- Deep power-down: 0,3 mA

2.6. Miscellaneous

- three 8254-compatible timers, of which 2 are available for external use
- two 8259-compatible interrupt controllers
- Real Time Clock (RTC), optionally with battery support
- programmable watchdog timer
- unique hardware serial number (can be used by programs for copy protection).
- available with extended temperature range, -20°C ... +85°C
- Housing dimensions: 43 x 36.4 x 5 mm (WxDxH)

3. MicroPC Starter Kit

3.1. Starter Kit Contents

The MicroPC starter kit contains the following components:

- MicroPC (CPU module)
- MicroPC base (base and prototyping board)
- Power supply pack, input AC 230V, output DC 9 to 16V, min. 400 mA
- Connection cable (serial null modem cable with two 9-pin DSUB connector)
- Adapter cable DSUB-9 to 10-pin header
- Diskette with BIOS and ROM-DOS files, plus utilities
- Manual

3.2. MicroPC Base

The starter kit board ("MicroPC base") makes it easy to put the MicroPC to use.

The MicroPC base is designed to be both simple and universal. Some elements of the circuit board will not always be needed, but facilitate implementation for certain purposes. The connection diagram of the MicroPC board can be found in chapter 11 of this manual.

Technical Properties

Power supply:

From an unregulated input voltage between 8 and 35V, three voltages are produced:

- 3.3V for the CPU module,
- 5V for optional connection of peripherals,
- 22V as contrast voltage for certain LCD modules (those that do not produce their own contrast voltage).

RS232 driver/receiver:

for the RxD, TxD, RTS and CTS signals of both MicroPC serial ports.

Connections:

- MicroPC slot (CompactFlash slot),
- 50-pin header connector (2.54mm spacing) for the MicroPC signals,
- Two 10-pin headers with 2.54mm spacing for the RS232 interface,
- Two single-row headers (2.54mm spacing) for LCD (X6) and matrix keyboard (X7) cables,
- 2-pin terminal block for the power supply. A DC connector can also be mounted here.

3.3. Start-up

Start the installation program INSTALL.BAT on the provided disk. The command

```
a:\install  c:\ micropc
```

will unpack the MicroPC files into the indicated directory (here, for example: c:\micropc).

In the subdirectory UTIL you will find the terminal program VTERM.EXE, which can be started without additional parameters. If using a COM port other than COM1, you can select a new COM port by typing Alt-C. With Alt-H you get an overview of the terminal program's commands (see also chapter 6.1.2). The Baud rate of the MicroPC normally is equal to 115200 Baud.

Note: If the ACPI function of the power management of the host PC are enabled, communication from the PC to the MicroPC will not work unter certain conditions. In this case, ACPI must be disabled beforehand.

Connect the MicroPC's 10-pin COM1 (X3) connector is connected to a free serial interface (COM port) of the PC using the serial cable and the adaptor cable. If a serial cable other than the one provided is used, it must be a null modem cable ("crossover cable"). The cable that comes with the starter kit supports only TxD and RxD. The signals CTS and DCD are each connected with RTS on the same end, inside the connector; in other words, they are not conducted to the other end of the cable**.**

Attach the power supply to the DC connection next to the serial ports. With a correct installation you can now observe the BIOS booting. After the memory test, DOS will boot and the MicroPC will display a DOS prompt. From the DOS prompt, you can work with the MicroPC as you are accustomed to working with a desktop PC.

3.4. MicroPC Drives

The Micro PC offers several "drives". Drive A: is a ROM disk, drive B: a RAM disk, drive C: a flashdisk (from which the MicroPC boots according to the standard settings) and drive D: a CompactFlash module (optional). These drives can be accessed like floppy disk drives or hard disk drives. Drives A:, B: and D: are deactivated by default, but can be activated in the BIOS setup. The RAM disk (drive B:) can then be used immediately. In order to be able to use drive A:, a ROM disk image must first be loaded via the BIOS setup.

3.5. Acceleration of the Boot Procedure

When the ROM-DOS is booting, the F5 key circumvents the execution of CONFIG.SYS and AUTOEXEC.BAT, and the F8 key permits individual execution of each command. The ROM-DOS waits a few seconds for a keyboard entry. You can be avoid this delay through the command

```
switches=/f
```

at the beginning of the CONFIG.SYS file. However, interrupting the the boot process by F5 or F8 is then more difficult (although still possible).

If the option "Boot Messages" in the BIOS Setup is turned off and "Fast Boot" is turned on, the complete boot procedure of the MicroPC will take less than two seconds.

3.6. Mapping PC Drives with Remote-Drive

Exchanging program or data files between the PC and the MicroPC might take place with any terminal program supporting the X-Modem or Z-Modem data transmission protocol. In addition, the more specialised terminal program VTERM, which comes with the MicroPC, provides the possibility of integrating PC drives (remote drives) into the DOS of the MicroPC. Standard DOS commands and DOS functions can therefore be used to access files on the host PC. The MAP.BAT file on the flash disk contains examples of how to initialize the appropriate drivers. Details can be found in the chapter about RDRIVE.

3.7. Configuring the BIOS

Communication takes place via the MicroPC's first serial interface. The baud rate is normally set to 115200 Baud. Configuration of the card is done in the BIOS setup. In order to start the BIOS setup, the "S" key must be pressed during the boot process.

Note: Some settings, e.g. switching off the standard I/O port, can disable any access to the MicroPC. In such a case, standard settings can temporarily be restored by shorting Pin 19 ("PIF Ready") of X2 to ground while booting. The BIOS setup is then activated automatically, allowing the correction of the settings.

This possibility can be turned off with the setting "Emergency Jumper: disabled" in order to prevent unauthorized access to the software. However, this should only be done if the final status of an application is reached.

The supply voltage should never be turned off during storage of the setup values, since otherwise the BIOS will be erased. Storing the setup values takes a few seconds, and afterwards the card reboots.

4. Programming the MicroPC

4.1. Use of PC Compilers

Because it is DOS compatible and for the most part PC compatible, the CPU card can be programmed just like a normal DOS PC. That is to say, the usual programming tools for the PC, so far as they are suitable for DOS programs, can also be used for the MicroPC. Basic, Pascal and "C" are particularly apropos.

Certain restrictions must be considered regarding console (video) output. The respective compiler must be configured in such a way that the output is made by DOS or BIOS calls. Only then can the output be redirected to the serial interface, which represents the MicroPC's console port. Programs that access the video memory directly — a popular method of DOS compilers because it is fast — are normally not possible on the MicroPC (for some LCDs there are BIOS extensions that use the 386 processor’s virtual 8086-mode to intercept accesses to the VGA chip and video memory, converting them in a appropriate way).

4.2. Accessing PIF Peripherals

The PIF bus works "I/O mapped", meaning that it is addressed by using the 386EX processor’s IN and OUT commands. The address space of the PIF-Bus comprises the 64 I/O addresses from 300 to 33Fh. The data bus of the PIF bus is 8 bits wide.

4.3. Testing Programs

4.3.1. Debugging on the PC

Programs for the MicroPC should be tested as far as possible on the development PC, since PC development environments and debuggers can be used there without limitations. Accessing peripheral hardware that needs the PIF-bus interface can be done with the PIF-ISA-Base interface card. This card occupies one ISA-bus slot (a PCI card is not currently available). Its hardware addresses and IRQ can be flexibly configured. This method works for all PIF I/O cards.

4.3.2. Remote debugging

An effective method for debugging source code directly on the MicroPC is the remote kernel of Borland’s Turbo-Debugger. For this method, one of the two serial interfaces of the MicroPC must be reserved exclusively for the debugger — any other activities on the same interface disturbs communication between the remote kernel and the host program. The remote kernel (tdremote.exe) is copied to the Flash disk of the MicroPC and started from there.

5. Hardware

5.1. 386EX Core

The CPU core of the Intel 386EX constitutes a fully static 386SX. Its data bus is 16 bits wide, and its address bus 26 bits (386SX: 24 bits) wide. An address space of 64 Mbyte memory and 64 kByte I/O is available. The processor core supports protected-mode applications.

5.2. Memory

5.2.1. RAM

5.2.1.1. RAM layout

The MicroPC can be equipped with 512KB or 1MB SRAM. The starter-kit MicroPC is equipped with 1MB SRAM. This memory is accessible in the processor's “Real Mode”, and thus in the lowest MB. The range between 896KB and 1MB is excluded — this is the area where the part of the flash memory with the BIOS (48KB), the ROM-DOS kernel (48KB), and optionally a BIOS extension (maximally 32KB) reside. If necessary, this flash range can also be set to 256KB or 512KB, so that only a maximum of 768KB or 512KB RAM are available in the lowest MB.

So, when equipped with 1MB SRAM, at least 128KB are not addressable in Real Mode. The entire RAM is however also made available in the second MB of the address space (as "Extended Memory") and can be addressed there with protected-mode functions (BIOS INT 15h, function 87h).

5.2.1.2. Battery backup

The static RAM and the real time clock IC (RTC) can be powered by a backup battery. The RAM maintains its contents when the operating voltage is turned off, and the real time clock keeps running. This backup power is supplied by means of the I/O connector's Vbatt pin.

The minimal voltage for the SRAM and RTC is 2V. With a lithium cell of 3.2 V, SRAM and RTC together draw about 2 µA (typical value). No battery current is drawn when the normal operating voltage of 3.3V is on.

5.2.1.3. Goldcap

Instead of a battery a "Goldcap" can also be used, in other words a capacitor with particularly large capacity (e.g. 1F). Data can be held over several hours. The advantage over a battery is in the relatively maintenance-free operation. The Goldcap's charging current may be supplied from Vcc during normal operation, e.g. by way of a diode a with series resistor to the limit the voltage.

5.2.2. Flash memory

5.2.2.1. Flash memory layout

The MicroPC can be equipped with 1MB, 2MB or 4MB Flash memory. This is organized in blocks of 64KB, which can be erased individually. The BIOS and ROM-DOS kernel occupy 96KB of the flash memory, while 32KB are reserved for optional BIOS extensions. The remainder is available as flash disk, ROM disk, or as freely usable memory for application programs. The BIOS functions of Int 5Fh exist for this purpose.

A part of the Flash memory, namely 128KB, 256KB or 512KB (including BIOS and ROM-DOS), can be made accessible within the real mode address space. Program code residing residing within this location can be executed directly from flash memory. Of course, this part of the address space is no longer availble for RAM.

5.2.2.2. Limited number of erase cycles

Flash memory consists of "Large Sector Flash-ICs" (e.g. AMD's 29LV800 or compatible). Only a limited number of erase cycles per block are tolerable for these devices (usually one million erase cycles are guaranteed by the manufacturer). This means that the flash memory, in particular the flash disk, is not suitable for permanent write operations of a program, since the permissible number of erase cycles per block might be exceeded in a relatively short time. A RAM disk must be used for such purposes.

5.2.2.3. Flash disk

The largest section of the Flash memory is usually occupied by the Flash disk. This is organized like a hard drive, and is accessed by the ROM-DOS by means of the BIOS functions of Int 13h. By default, the files COMMAND.COM of the ROM-DOS operating system as well as some utility programs for serial data communication (RDRIVE, XLOAD, XSEND) are stored on the flash disk. Typically, the application program and application data will also reside on the flash disk.

The MicroPC's Flash file system is extremely stable against sudden power failures, to which battery-operated devices particularly easily succumb. The flash file system will work reliably even if a power failure took place during flash disk operations. However, "lost clusters" may possibly arise, if files were still open under DOS. These can be deleted with the DOS utility program CHKDSK.

5.2.2.4. Creation of a new Flash disk

- Create a directory on the PC for all files which are to appear later on the Flash disk.
- Copy all necessary files into this directory. Among these should be at least COMMAND.COM and a program for data communication (RDRIVE.EXE, RMAP.EXE or XLOAD.COM). In addition one can create the appropriate CONFIG.SYS and AUTOEXEC.BAT files here.
- Create a flash image file with the program FLASHHDD.EXE according to the description in the chapter "PC Programs".
- Start the terminal program and the MicroPC, and press the < S > key during the memory test. You will arrive at the CPU card's setup menu.
- By pressing < L > , choose the ***FLASH*** menu. From here, select the function ***Update Flashdisk*** and confirm with < Enter >. The confirmation question must be answered with "Y". The BIOS deletes the flash disk range and then waits for the file transfer (this can be recognized by appearance of the "§" characters). Then send the flash image file with the terminal program (ALT+ <S> with Vterm). To do so, select the transmission protocol „X-Modem" and enter the path of the flash image file. A few seconds after completion of the transfer the card will reboot.

5.3. Real-Time Clock

The real-time clock ("RTC") provides date and time for application programs. Leap years and a 24-hour mode are both accounted for. As on a PC, the RTC can generate an interrupt (IRQ 8). Besides the usual scheduled interrupt (interrupt at a certain pre-programmed time), the RTC can also release a cyclical interrupt at every minute, hour, or day, or with 1Hz or 4096 Hz. As on a PC, the BIOS functions for the RTC are available at Int 1Ah for reading and programming.

The RTC can be supplied with backup power via the Vbatt pin of the MicroPC's connector so that it keeps running if the MicroPC's power supply is switched off.

The active-low RTC interrupt signal (open-drain output) is on the I/O connector of the MicroPC (Wakeup) and can be used to trigger actions of the peripherals. This also functions in the deep power-down mode (with the CPU oscillator turned off), but the power supply to the MicroPC must remain switched on. The Wakeup signal can also be used as an external interrupt input if the RTC interrupt is not needed.

5.4. TCU (Timer/Counter Unit)

The TCU is widely compatible with the 8254 from Intel (and thus with PC standards). Further details about the timer can be found in the 386EX processor manual from Intel.

Properties of the 8254:

- three 16-bit timers,
- six counter modes,
- BCD or binary numbers,
- separate interrupts for each timer (IRQ 0, 10 and 11),
- clock source internally or externally selectable (only for Timer 0 and 1).

Timer 0 is programmed by the BIOS according to PC standard in mode 3 to achieve an output frequency of 18,206 Hz (periodic timer interrupt, system timer). Its output is connected to IRQ0. The BIOS interrupt routine increments the 32-Bit timer variable at RAM address 0040h:006Ch each time it is called. This variable serves as the basis for the DOS system time.

Timer 1 (responsible for the DRAM Refresh on a normal PC) and likewise also **Timer 2** (for the loudspeaker on a PC) are freely available. On the MicroPC, Timer 1 and Timer 2 can also generate an interrupt (IRQ 10 and 11).

Timer 0 and Timer 1 can be operated with an internal or external clock. Timer 2 can be operated only with internal clock. The internal clock is produced by a prescaler from the CPU clock (25, 20, 8 or 4 MHz). The prescaler can be adjusted to values from 2 to 513. The BIOS sets the value of the prescaler in such a way that the clock is somewhat slower than the timer clock of a PC (1.193182 MHz), i.e. than the next higher whole number to the quotient from 25 MHz and 1.193182 MHz. This gives a prescaler of 21 and a timer input clock of 1.19047 MHz. When downclocking the CPU clock by the BIOS Int 15h function C311h to 20, 8 or 4 MHz, the prescaler is adapted to 17, 14 or 7.

The following timer signals are sent out on the I/O connector X1:

Timer	Output	Clock	Gate
Timer 0	Pin 14	Pin 24	Pin 45
Timer 1	Pin 39	Pin 16	Pin 46
Timer 2	---	---	---

The signals of Timer 0 and Timer 1 are multiplexed with other functions. The appropriate settings can be made in the BIOS I/O Setup (see 8.3).

5.5. Watchdog Timer

The 386EX watchdog timer is a 32-bit timer that the MicroPC uses as a programmable watchdog. The input corresponds to the CPU clock rate (25 MHz at full speed); the maximum watchdog timeout period therefore amounts to about 172 seconds. The watchdog timer releases a hardware reset at the end of the timeout period. The application program must persistantly reset the watchdog timer before the timeout is reached. If an application program has crashed for some reason, the watchdog timer will (usually) reset the system, thereby reproducing a well defined state once again.

For activating and resetting the watchdog, BIOS functions of Int 15h are available. Once activated, the watchdog can be deactivated only by a hardware reset.

The use of the watchdog timer as a universal timer (with IRQ15) is possible only with special versions of the MicroPC hardware (another reset generator and relinquishment of the watchdog function), to prevent the timer output from producing a system reset.

5.6. Interrupt Controller

5.6.1. General

Like a normal PC, the 386EX processor has two on-chip 8259-compatible programmable interrupt controllers (PICs). Thereby 15 interrupt requests (IRQs) are available, some of which are already taken up by the MicroPC. IRQs 1, 5, 7, 9, 13 and 14 are free for applications. If no RTC interrupt is needed, IRQ8 can also be used for other purposes (RTC on signal, RTC open-drain output). If the first or second serial interface are not needed or can be operated without interrupts, then IRQ4 and/or IRQ3 are also available. Note that IRQ8 is inverted on the connector, and is thus active-low.

The PIC2 output connects to the IRQ2-input of the PIC1. The IRQ2-input is therefore configured as a slave input, so that there is no IRQ2 in the literal sense. Thus, furthermore, no IRQ is assigned to the vector Int 0Ah. The CPU core has a general interrupt input, to which the output of the PIC1 is attached. There are also the NMI and the SMI interrupt inputs, but these are not used on the MicroPC.

Programming details for the interrupt controllers can be taken from Intel's 386EX-processor manual or the usual PC literature. Therefore only some references are specified here.

5.6.2. Edge and level triggering

The interrupt controllers of a PC traditionally operate with edge triggering. A reprogramming to level triggering is not recommended, since problems will arise with the BIOS interrupt functions (especially the RTC and Timer 0). If the RTC interrupt is not needed, the PIC2 can be adjusted to level triggering. The advantage of level triggering: several units can share an IRQ by wired OR and/or wired AND functions (this does not work well with edge triggering, since edges are lost if two interrupt signals on a line occur at the same time.) The disadvantage of level triggering is the fact that the interrupt source must be reset immediately within the service routine (which is, for example, simply not possible with the timer). Otherwise further interrupts will arise.

5.6.3. Assigned interrupt vectors

A range of eight consecutive interrupt vectors can be assigned to each PIC by software initialization. On a PC, the vectors Int08 to Int0Fh are traditionally assigned by the BIOS to the PIC1 and the vectors Int70h to Int77h are assigned to the PIC2. According to the PC tradition, the PIC1's vectors therefore lie within the range of Int0 to Int 1Fh, designated by Intel as "reserved." In other words, they share the vector with certain processor exceptions. In practice this does not usually lead to problems.

5.6.4. Masking of interrupts

In order to activate an IRQ, the associated bit in the mask register of the PIC must be set to 0. For the IRQs of the PIC2, the PIC1's mask bit 2 (belonging to IRQ2) must also be set to 0. The mask registers are located at I/O addresses 21h for PIC1 and at A1h for PIC2.

5.6.5. Resetting the interrupt controller

Each IRQ input has an in-service bit in the interrupt controller. In principle the interrupt service routines must reset the in-service bit of the relevant IRQ, since otherwise no further IRQ with the same or lower priority can be generated. This is normally done via the "nonspecific reset instruction"

```
out [20], 20
```

and/or for the PIC2:

```
out [A0], 20
```

or in "C":

```
_outp(0x20, 0x20); _outp(0xA0, 0x20);
```

thus through output of byte 20h at I/O address 20h and/or A0h. For the IRQs of PIC2, the in-service bit of IRQ2 must also always be reset; both of the indicated “Out” commands must therefore be made.

It must be noted that upon entering an interrupt routine the CPU disables all interrupts by first resetting the Interrupt Enable flag. This applies to IRQ service routines as well as to software interrupts (only in protected mode a different setting can be selected for each interrupt). Therefore the interrupts should be re-enabled within the interrupt routine as soon as possible (also depending on the requirements of the application).

5.6.6. Non-Latching of IRQs

On the 8259 IRQs are **not latched** (stored), even though the Intel manual gives this impression. If a peripheral drives its IRQ signal low again before the CPU can service the IRQ, then this interrupt is lost. This may happen if interrupts are disabled in the CPU, or if the CPU is servicing another IRQ of higher priority at this particular moment. Only directly during an interrupt acknowledge CPU cycle will the states of the IRQ inputs be frozen, in order to generate an unequivocal interrupt vector. The Interrupt Request Register (IRR) otherwise shows only the state of the IRQ inputs. This applies independently of whether or not an IRQ is masked out. If, in the edge-triggered mode, an IRQ is just being processed (in-service bit set), the IRR's relevant bit will be read as 0, since the Edge Sense Latch disables the input. The Edge Sense Latch is reset by a low impulse on the IRQ input, even if the in-service bit is still set, so that the IRQ input via IRR can be read in again.

5.6.7. Spurious interrupt

The non-latching of IRQ impulses also makes default or spurious interrupts necessary. This situation arises with "dirty" IRQ signals, e.g. with IRQs from bouncing keyboards. If the CPU executes an INTA cycle in reaction to an IRQ, but the PIC has already forgotten the associated IRQ input (since the signal was reset), the PIC must nevertheless send an interrupt vector to the data bus (a random value on the data bus could otherwise make the computer hang). This is then the "spurious IRQ7". It is even activated if the IRQ7 is masked out. The IRQ7's in-service bit is not set in the case of a spurious IRQ. A spurious IRQ15 can also occur with the PIC2. Whether an IRQ7 or an IRQ15 arises depends on the timing of the input signal. Because of the PIC2's delay the IRQ signal remains valid on the PIC1 somewhat longer, so that the PIC1 could potentially regard the IRQ as not "spurious" and activate the PIC2, which would then produce a spurious IRQ15.

5.6.8. IRQ priority

On a PC, the priority of the IRQs is usually specified in such a way that IRQ0 has the highest priority, and the other IRQs follow in numerical order. Since the PIC2 is attached to IRQ2, the IRQs of the PIC2 take priority over the IRQ3. However the service routines of the PIC2 cannot be interrupted by higher priority PIC2 IRQs so long as the in-service bit of the IRQ2 is not reset. But if this bit is reset, they can also be interrupted by lower priority PIC1 IRQs.

The priorities can also be changed, however only cyclically within the individual PICs. In other words, one specifies the IRQ with the lowest priority, from which the other priorities within this PIC are automatically determined. The definition of lowest priority for the PIC1 and/or PIC2 is made via the command

```
out [20], C0+ IRQ-Nr
```

and/or

```
out [A0], C0+ IRQ-Nr
```

For example,

```
out [20], C3
```

sets IRQ3 to the lowest priority whereby the IRQ4 (of COM1) gets the highest priority.

5.7. Asynchronous Serial Interfaces

Two UARTs compatible to the well-known 16C450 are integrated into the 386EX processor. In contrast to the UARTs that are integrated these days into PC main boards (which are compatible to the 16C550) they have no FIFOs.

The first serial interface (COM1) is used as the MicroPC's standard input/output port (DOS "con" device). For “con,” COM2 or an external serial interface can also be employed (COM1 through COM4 are possible).

5.7.1. Signals of the serial interfaces

Both serial interfaces have the eight usual PC signals: data lines (RXD, TXD), modem status inputs (DSR, CTS, DCD and RI) and modem control outputs (RTS, DTR). Frequently one of the pairs DTR/DSR or RTS/CTS is used for handshake operation.

The eight COM1 signals can be reconfigured individually as input or output ports (see the chapter, "I/O Ports").

In addition, the COM2 signals RTS, DTR, DSR and RI can be used alternatively as signals of the synchronous serial interface (see relevant chapter).

5.7.2. BIOS functions

The BIOS offers the usual PC INT 14h functions for operation of the serial interface. Deviating from the PC tradition, these are operated with receive interrupts in the case of the MicroPC. This setting can be deactivated in the BIOS Setup. The buffer size specified in the Setup is effective only in interrupt mode. In order to reduce computing time, many applications program the serial interfaces directly by accessing the UART registers. In such a case, at least the receive interrupt is usually used.

5.7.3. Hardware interrupt of the serial interfaces

As on a PC, COM1 and COM2 use IRQ4 and IRQ3. In the event that the interrupts are not needed for the serial interfaces, IRQ4 and IRQ3 can be used externally for other purposes.
There are four different interrupt sources for each UART, which however all use the same IRQ line.

- Line-status interrupt:	overrun, parity or framing errors, or break ;
- Receive interrupt:	a character was completely received (receive buffer full);
- Transmit interrupt:	a character was completely sent (transmit buffer empty);
- Modem-status interrupt:	the status of DCD, RI, CTS or DSR has changed*.

Definitions:

Overrun Error: The Receive Buffer register was overwritten by a further character. This means that the previous character was not retrieved by the processor in time;

Parity Error: The parity or the parity bit (with forced parity) of the received character was unequal to the complement of the Even Parity Select (EPS) bit in the line control register. This happens only with a switched-on parity bit. With "forced parities" the parity bit can be used in a way similar to a ninth data bit, since one can set it by means of the EPS bit when sending, and interpret it by means of the line-status interrupt when receiving.

Framing Error: A character did not have a stop bit (or had a stop bit too few, if 1.5 or 2 stop bits are set). For a valid stop bit the RXD line must remain in the high state for the duration of one bit.

Break condition: The RXD line went on “low” for the duration of more than one character. This also always results in a framing error. A break condition can be generated when sending by the setting of the line control register’s break bit. The minimum time duration of a character can be achieved by sending a character and waiting afterward until the Transmit Shift register is empty. Afterwards the break bit is again reset. One can signal a special condition to the receiver in this way, without having to fall back on certain byte values or byte sequences.

* with RI: only rising edges must generate an IRQ.

5.7.4. UART register

Divisor Latch low (DLL, Address 0)

Divisor Latch high (DLH, Address 1)

Interrupt Enable Register (IER, Address 1):
- Bit 0: Receive Interrupt
- Bit 1: Transmit Interrupt
- Bit 2: Line Status Interrupt
- Bit 3: Modem Status Interrupt
- Bit 4..7: 0

Line Control Register (LCR, Address3):
- Bit 0: Word Length Bit 0
- Bit 1: Word Length Bit 1
- Bit 2: No. of Stop Bits (1 or 2)
- Bit 3: Enable Parity Bit
- Bit 4: Select Even Parity
- Bit 5: Select Forced Parity
- Bit 6: Set Break
- Bit 7: Divisor Latch Enable

Interrupt Identification (Status) Register (IIR or ISR, Address 2):

Bit 0:	0 = Interrupt Pending
Bit 1..2:	0 = Modem Status Interrupt
	1 = Transmit Interrupt
	2 = Receive Interrupt
	3 = Line Status Interrupt
Bit 3..7: 0	

Modem Control Register (MCR, Address 4):
- Bit 0: /DTR
- Bit 1: /RTS
- Bit 2: OUT1: Test Bit for /RI in Loop-Back Mode
- Bit 3: OUT2: Test Bit for /DCD in Loop-Back Mode; activates the UART interrupts*
- Bit 4: Set Loop-Back Mode
- Bit 5..7: 0

Line Status Register (LSR, Address 5):
- Bit 0: Received Data Ready
- Bit 1: Overrun Error

Bit 2: Parity Error
Bit 3: Framing Error
Bit 4: Break Condition
Bit 5: Transmitter Hold Register Empty
Bit 6: Transmitter Shift Register Empty
Bit 7: 0

Modem Status Register (MSR, Address 6):
Bit 0: Delta CTS
Bit 1: Delta DSR
Bit 2: Delta RI
Bit 3: Delta DCD
Bit 4: /CTS
Bit 5: /DSR
Bit 6: /RI
Bit 7: /DCD

Scratch Register (SCR, Address 7)

*OUT1 and OUT2 are originally universal digital outputs of the UARTs 8250 and 16C450. On a PC, the UART interrupt output is traditionally connected to the interrupt controller by means of OUT2 through a tri-state buffer, or it is separated (deactivated). Certain UARTs already have this gate internally, such as the Exar/Startech ST16C552. On the 386EX, a multiplexer is controlled by OUT2, which can switch between the UART's IRQ output and an external signal (pin).

5.8. Synchronous Serial Interface

The 386EX-Prozessor has a synchronous serial interface (SSIO) in addition to the two asynchronous ones. There is a data line and a clock line for sending and another pair for receiving. These four signals are located on the X1 connector on the same pins as the COM2 signals DTR, DSR, RI and RTS.

The SSIO can be operated at a maximum of half the processor clock speed (CLK2/4), for example at most 12.5 Mbaud for the MicroPC with a 25 MHz CPU. 16-bit words are transmitted.

The transmitter and receiver can both operate in either master or slave mode. The IRQ9 interrupt is optionally produced for the "Transmit Buffer Empty" and "Receive Buffer Full" states.

When using the synchronous serial interface one must note these two known bugs, documented by Intel, which they have never fixed:

- Auto-Transmit Mode. This would actually be the normal master transmit mode. However it is correctly usable only at the maximum Baud rate, since the first bit of a data word is output at the maximum rate regardless of the Baud rate setting.
- The status register’s Transmit Buffer Empty bit does not return a correct value. Before turning off the transmitter, one must poll the Baud-rate counter and wait until the first bit is completely sent.

5.9. I/O Ports

The MicroPC has a maximum of 32 freely programmable digital I/O ports on one connector. For the most part these can be configured independently of each other as input or output. Some of these pins are also used by the timer, the interrupt controller and the first serial interface (see BIOS I/O Setup).

Seventeen (17) of the I/O ports belong to the processor’s internal port registers. The PIF bus’s data, address and control signals constitute an additional 15.

5.9.1. Processor ports

These ports are addressed as part of the 386EX’s port registers (P1, P2 and P3). The direction (input or output) is specified in the BIOS I/O Setup or by means of the port’s direction register (0 = output, 1 = input or open-drain output). If the inputs are to be used, the respective bits of the output port must be set to 1 (because of the open drain outputs).

	Configuration	Pin State	Output	Direction
P1	F820h	F860h	F862h	F864h
P2	F822h	F868h	F86Ah	F86Ch
P3	F824h	F870h	F872h	F874h

Itemized below are the bits of these registers that are accessible on the connector (see also the table in chapter 10):

Port/Bit	Alternative Funktions
P1.0	COM1 DCD or IRQ14 or Timer 1 Gate
P1.1	COM1 RTS
P1.2	COM1 DTR
P1.3	COM1 DSR or IRQ9 or Timer 0 Gate
P1.4	COM1 RI
P2.0	PIF bus /CS0
P2.1	PIF bus /CS1
P2.2	PIF bus /CS2
P2.3	PIF bus /CS3
P2.5	COM1 RXD
P2.6	COM1 TXD
P2.7	COM1 CTS
P3.0	IRQ4 or Timer 0 output (TOUT0)
P3.1	IRQ3 or Timer 1 output (TOUT1)
P3.2	IRQ1
P3.3	IRQ5
P3.5	IRQ7 or I2C bus SCL

The remaining bits of the registers for ports P1, P2 and P3 should not be modified by application programs, since they are used within the MicroPC.

5.9.2. PIF-Bus Ports

PIF-bus signals D0..D7, A0..A3, –CS0..–CS3, –RD, –WR and Ready can be used as digital ports can be used alternative (in part simultaneously) to bus usage. The chip selects -CS0 to -CS3 can be used as processor ports (see the preceding chapter). The remaining pins are addressed by I/O addresses 100h to 103h (see also the table of I/O addresses in the "Tables" chapter). The signals -RD, -WR and Ready can be individually tri-stated; for D0..D7 and A0..A3 this can be done in groups of four (D0..D3, D4..D7, A0..A3). The signals are read or written via the I/O addresses 100h (D0..D7) or 101h (A0..A3, RD, WR, Ready).

For PIF bus usage all signals except Ready must be configured as outputs by using the PIF configuration register 102h. E. g. in ' C ' by the command

```
_outp(0x102, 0x3F)
```

Bit 7 of the PIF configuration register is used for activating fast PIF mode, which reduces the number of wait states the processor performs during PIF-bus cycles (beware: not every peripheral hardware will work with fast mode).

The states of pins –RD, -WR and Ready can also be changed via the Toggle Register 103h. Sending “1” to the bit associated with this register inverts the signal from its previous state.

5.10. I2C Bus

The BIOS provides some simple functions for accessing the I2C bus (see BIOS Reference). All signals of ports P1, P2 and P3 which are available on the MicroPC connector can be used by BIOS functions as the I2C bus’s SDA and SCL signals. The desired port pins are selected in the BIOS setup I/O menu. They must be configured as inputs (which in our case is equivalent with open drain outputs). Port pin P3.5 is already used for the internal I2 bus as an SCL signal (for the RTC). It is therefore particularly well suited for use with the external I2C bus.

Pull-up resistors in the range of 1kOhm to 4.7 kOhm must be provided, except in the case of P3.5, which already has one on board.

Since P1, P2 and P3 are CPU ports, the maximum permissible signal level is only 3.3V. In order to also connect 5V signals to the I2C bus, external N-channel MOSFETs can be used (one each for SDA and SCL). Their gate is attached to the 3.3V supply voltage, the source to the MicroPC-side bus signal, and the drain to the peripheral-side (5V) bus signal.

5.11. PIF Bus

5.11.1. Overview

The PIF bus is simple 8-bit extension bus for connecting peripheral cards to the MicroPC. The bus architecture is derived from the interfaces of various LCDs (although their plug allocations are not uniform). Thus LCDs with the Toshiba T6963C controller can even be operated directly on the PIF bus.

The address space consists of 64 I/O addresses. However, it is not 6 address lines that are used, but 4 chip-select lines and 4 address lines. Of the chip-select lines, only one is ever active (1 from 4 code). Thus 16 I/O addresses are assigned to each chip select. This principle simplifies address decoding.

In many cases, address coding is completely unnecessary. For instance, the well-known PIO component 82C55 can be operated directly on the PIF bus by using the signals /CS0, /RD, /WR, A0, A1, the data lines and the operating voltage. Thus of the 16 addresses that belong to chip select 0 (/CS0), effectively only four would be used, although all 16 are occupied. This "disappearance" of addresses is not a problem in many systems (those which need little in the way of peripherals), and simplifies the design.

Substantial are the active-low read (/RD) and write lines (/WR), of which exactly one is active for each PIF bus access, according to whether it is a read or a write cycle. The data lines are sampled in each case on the **rising** edge, thus toward the end of the bus cycle.

5.11.2. Hardware design for the PIF bus

The following points must be noted when designing hardware to connect to the PIF bus:

1. Access to the PIF peripherals is through I/O commands. Memory-mapped accesses are not possible.
2. The four address lines of the PIF bus correspond to the lowest four address lines of the CPU bus. They can therefore accept any offset value from 0 to 0Fh.
3. Exactly one chip-select line is active (low) during a valid PIF bus access.
4. The four chip-select lines are decoded from address lines A4 and A5 of the CPU bus. Therefore they correspond to offset values of 0h, 10h, 20h and 30h.
5. The base address of the PIF bus is added to the offset values. For the MicroPC it is 300h. Other CPU cards may vary.
6. Exactly one of the signals /RD or /WR is active (low) during a valid PIF bus access. The peripheral must evaluate these signals and the chip-select signals - otherwise incorrect bus cycles can occur.
7. The data lines are sampled during both the reading and writing on the rising edge of the /RD or /WR signals.
8. The duration of a PIF bus cycle can be set to 1µs or 320ns (for a 25 MHz CPU clock-speed) in the PIF bus configuration register. A change of the CPU clock-speed causes a corresponding change of these values.
9. Ready signal: This signal is generated by the peripheral hardware in order to extend PIF bus cycles. Addresses, chip selects and /RD or /WR remain valid until the peripheral releases the Ready signal again (switches to high). The signal has a pull-up resistor on the CPU card. The peripherals must use open collector (open drain) outputs if more than one peripheral is designed to use the Ready signal.

5.11.3. PIF bus mechanics

The PIF bus signals for the MicroPC are located on a 50-pin, double-row header 2.54mm spacing. The pinout of the first 26 pins is compatible with the 26-pin connectors used by various *taskit* CPU cards. Thus the usual flat ribbon cables can be used. When using flat ribbon cables the cable length should not exceed 30 cm, in order to minimize disturbances by crosstalk and line reflections.

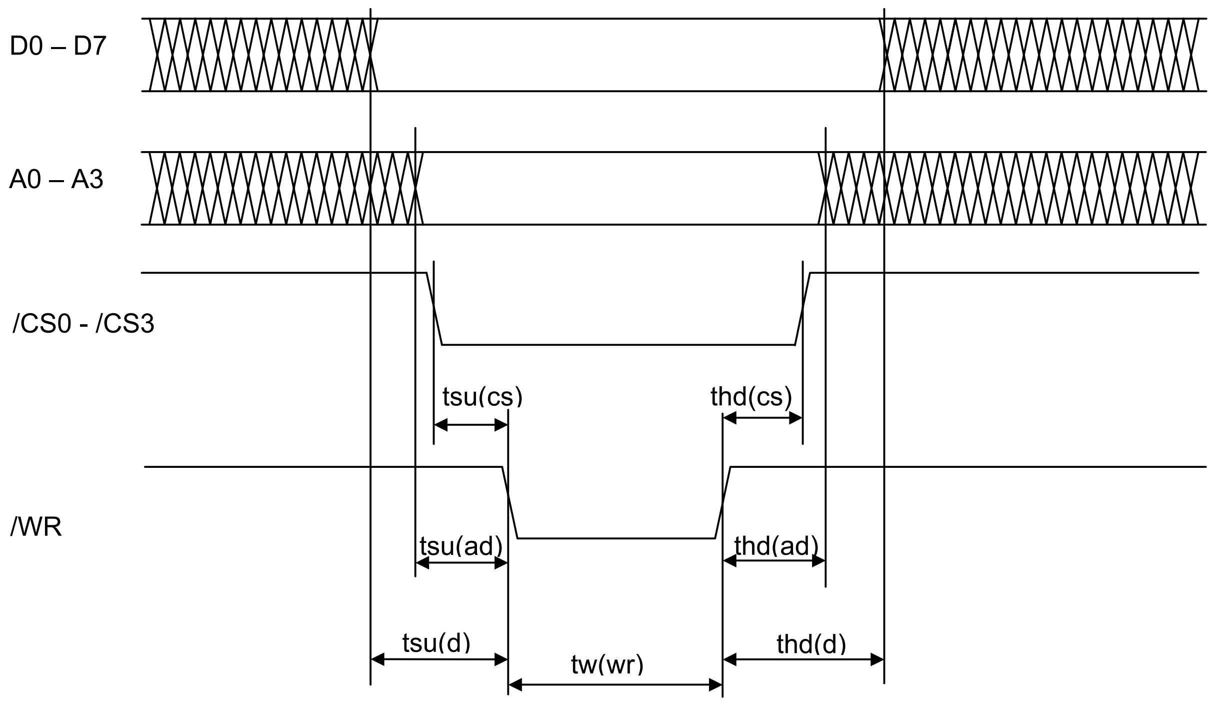
Lengths of up to approx. 1.5m are possible if additional GND lines are used. /RD and /WR signals in particular should be shielded from each other (and from other signals) by GND lines. These GND lines should be connected at both cable ends. The CPU card must be located on one end of the cable. Output termination resistors of 39 ohms to /RD and/WR are recommended.

Various available PIF cards are equipped with a header/socket combination, which makes it possible to stack several cards.

5.11.4. PIF bus signals

Signal	**Pin No. X2 Starter-Kit Board**	**I/O**	**active**	**Description**
D0 ... D7	11 ... 18	I/O	high	Data lines
/CS0 ... /CS3	7, 22, 23, 24	O	low	Chip select. For each PIF-bus cycle exactly one chip select is active.
A0 ... A3	8, 9, 20, 21	O	high	Address lines
/RD	6	O	low	Read Signal. Is active for each read access.
/WR	5	O	low	Write Signal. Is active for each write access.
/RESET	10	O	low	Reset Signal. This is the output signal of the MicroPC reset generator.
/INT	25	I	low	Interrupt Request. On the MicroPC this signal is inverted and connected to IRQ1. It has a 10kOhm pull-up resistor.
READY	19	I	high	Permits peripherals to extend PIF bus cycles (low = not ready). The bus cycle is terminated by the CPU once the ready signal is high again.
VCC	3			5V supply voltage
VEE	4			Negative supply voltage for the contrast current used by certain LCDs (not used by the MicroPC)
GND	1, 2, 26			Ground (negative connection for the supply voltage)

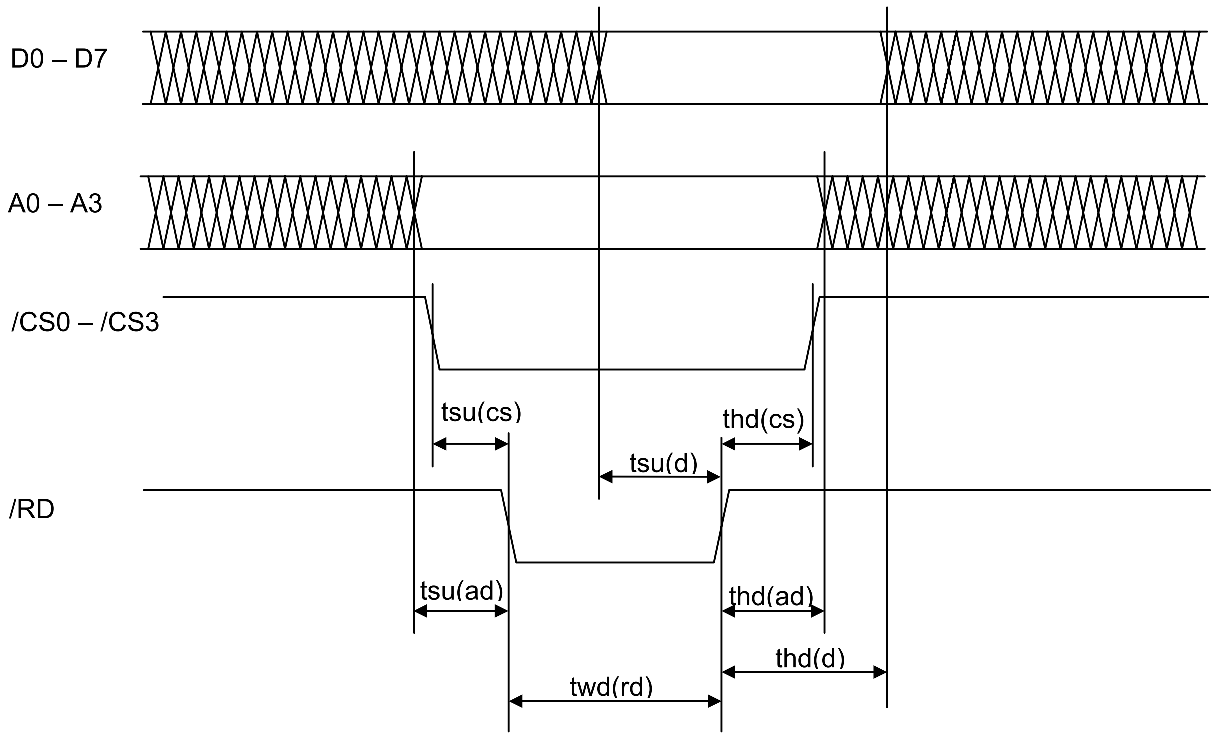
5.11.5. PIF bus timing (Write)



Symbol	Name	Description	min.	typical	max.
tsu(cs)	Chip Select setup time	/CSn low to /WR low	120 ns	4 CLK cycles	160 ns
thd(cs)	Chip Select hold time	/WR high to /CSn high	70 ns	2 CLK cycles	80 ns
tsu(ad)	Address setup time	A0..A3 valid to /WR low	120 ns	3 CLK cycles	
thd(ad)	Address hold time	A0..A3 valid after /WR high	70 ns	2 CLK cycles	
tsu(d)	Data setup time	D0..D7 valid to /WR low	120 ns	3 CLK cycles	
thd(d)	Data hold time	D0..D7 valid after /WR high	70 ns	2 CLK cycles	
tw1(wr)	Write pulse width	/WR low to /WR high (normal mode)	800 ns	20 CLK cycles	
tw2(wr)	Write pulse width	/WR low to /WR high (fast mode)	160 ns	4 CLK cycles	
tclk	Clock cycle	Clock cycle length at CLK2 = 50 MHz		40 ns	

Note: The relatively generous setup and hold times realized with the MicroPC are not standard for all CPU cards with PIF-bus. Setup and hold times of the PIF-bus are generally only guaranteed to be larger than 0. In other words, the Chip Selects, addresses and data are stable only during the /WR and /RD pulses.

5.11.6. PIF bus timing (Read)



Symbol	Name	Description	min.	typical	max.
tsu(cs)	Chip Select setup time	/CSn low to /RD low (Chip Select valid to Read valid)	120 ns	4 CLK cycles	
thd(cs)	Chip Select hold time	/RD high to –CS high (Chip Select hold after Read invalid)	70 ns	2 CLK cycles	
tsu(ad)	Address setup time	A0..A3 valid to /RD low	120 ns	3 CLK cycles	
thd(ad)	Address hold time	A0..A3 valid after /RD high (address hold after Read invalid)	70 ns	2 CLK cycles	
tsu(d)	Data setup time	D0..D7 valid to /RD high (Data valid to Read invalid)	10 ns		
thd(d)	Data hold time	D0..D7 valid after /RD high (data hold after Read invalid)	0 ns		
tw1(rd)	Read pulse width	Normal Mode	800 ns	20 CLK cycles	
tw2(rd)	Read pulse width	Fast Mode	160 ns	4 CLK cycles	
tdly(d)	Data delay time	READY valid to data valid delay			20ns
tdly(rd)	Read delay time	READY valid to Read invalid delay	1 CLK cycle		2 CLK cycles
tclk	Clock cycle	Clock cycle length at CLK2 = 50 MHz		40 ns	

5.11.7. Characteristics of the MicroPC PIF-bus

The data bus is switched to high impedance only during read operations. The rest of the time it has low impedance and holds the byte that was last read or written.

The address lines A0..A3 as well as the signals /RD and /WR change their level only by accesses to the PIF-bus or to port register 101h (which is assigned to them). With the help of register 101h one can therefore increase the address setup time for PIF bus access.

The four Chip Selects /CS0... / CS3 can be flexibly applied. Since these signals are generated directly by the chip-select unit of the CPU, the location as well as the size of the PIF address area can be changed with the help of the address and mask registers. So one can configure these in such a way that they function like additional address lines, instead of the usual 1-out-of-4 codes of the Chip Selects. An address space of 256 addresses may thereby be obtained compared to only 64 addresses in normal chip select mode.

In addition, /CS0.../CS3 can also be used as port pins (setting in BIOS Setup). Chip select functionality (low pulse during PIF-bus read and write cycles with simultaneous /RD or/WR pulse) is then no longer possible.

5.11.8. 386EX Bus Monitor And PIF-Bus

Since the MicroPC uses the internal watchdog timer of the CPU as a watchdog, it can not be used as a bus monitor. Accesses to unused I/O addresses then result in a "ready hang condition" (the processor hangs). An example is port 61h (NMI status register in a PC, missing in the MicroPC), which is read by the Ethernet packet driver (and also by different network drivers) in order to produce a defined delay based on the duration of the I/O Read command. Affected are fundamentally programs which are already finished and cannot be changed.

Therefore an otherwise unused chip select of the 386EX processor’s chip-select unit is used as a bus monitor. Only the four PIF-bus chip selects can be used for this purpose. Out of these, we use one whose pin is configured as a port pin (if any). The associated chip select signal is then available only internally within the processor. It is configured in such a way that it covers all I/O addresses. A "ready hang" condition can no longer occur through I/O commands.

If all PIF chip-select signals are to be used as chip selects, the BIOS initialization does not account for a bus monitor. It is nevertheless possible to mask critical I/O addresses with the help of CS0..CS3 by using the mask registers of the chip selects (see the Intel 386EX manual). An access to such an address (e.g. port 61h) then invokes a low pulse of the relevant chip-select line. That is not normally a problem, since a valid PIF bus cycle requires a low impulse on the /RD or /WR line, which should be decoded by the peripherals. Chip-select impulses resulting from addresses outside of the valid PIF bus range will not invoke a /RD or /WR pulse.

5.12. CompactFlash

CompactFlash (CF) cards are comman, internationally standardized memory modules. They are used in digital cameras and other products. CompactFlash cards are addressed by the BIOS as hard drives. They can be read and written by the PC through PCMCIA slots (PC-card slots) with a PCMCIA adapter. The media can currently (March 2003) hold a maximum of 1GB. There are also CompactFlash-compatible 1-inch hard drives from IBM and Hitachi with 320MB or 1GB (Microdrive™).

The MicroPC BIOS supports up to two CompactFlash cards attached to the PIF bus.

CompactFlash cards are registered in the BIOS setup as hard drives. Usually, LBA or CHS mode is used (but beware: if the CF module is to be used in a PC-card slot, it must be formatted compatibly to the respective PC). The CF card is assigned DOS drive letter D:. By selecting the "Swap Hdd0 and Hdd1" option in the setup, it is possible to to assign "C:" to the CF module, so that the MicroPC can boot from it. The DOS programs FDISK and FORMAT can be used for partitioning and formatting.

A schematics for a CompactFlash adapter for the PIF bus is given in the appendix.

5.13. Power Management

Power consumption can be drastically reduced in many cases by the power management functions of the BIOS. This applies whenever the full CPU performance at 25 MHz is not needed permanently.

5.13.1. Changing the CPU clock rate

The CPU clock speed can be lowered by BIOS functions to 20 MHz, 8 MHz or 4 MHz. The BIOS adjusts the divisor values for Timer 0.

Note that switching the clock rate does not happen instantly. To switch from the lowest to the maximum clock rate (4 MHz to 25 MHz) the clock generator needs about 4ms. The accuracy of Timer 0 is thereby affected too. Its divisor values (prescalers and timer registers) are re-loaded immediately by the BIOS function. Also, the current timer count is not taken into account by the BIOS function.

5.13.2. Idle mode

An application program can always switch, when no activity is taking place, into the idle or power-down mode. Only upon the appearance of an interrupt will the CPU resume execution of the program.

In idle mode only the clock for the CPU core is internally turned off, while the clock for the serial interfaces and the timers keeps running.

The normal operating condition is restored by each hardware interrupt that is not masked out (see also the chapter concerning interrupt controllers). Note that Timer 0 normally generates an IRQ every 55ms and thus automatically terminates the idle mode. The application program must ensure that the idle mode is restored as required after each IRQ.

The BIOS function for idle mode accounts for the simultaneous down-clocking of the processor. In idle mode the clock rate is given to the function as a parameter. Here again one must consider switching time (see above).

5.13.3. Power-down mode

The clock for the CPU core and internal peripherals of the 386EX is stopped. The timers continue to run only if they are operated with an external clock. The SSIO functions only in slave mode. Since the UARTs possess their own clock, they also function in power-down mode (though not in deep power-down mode).

As with idle mode, the BIOS function for the power-down mode accounts for the simultaneous down-clocking of the processor. The clock rate is given to the function as a parameter. Yet again, the switching time must be considered (see above). Deep power-down mode has its own BIOS function.

Restarting from power-down mode is done by an interrupt from the RTC, the serial interfaces, or the timer, or via external IRQs. These IRQs may not be masked out. Interrupts from the timers can terminate the power-down mode only if they are operated with an external clock (Timers 0 and 1) and/or with COMCLK (Timer 2), since the normal clock pulse for the timers (PSCLK) is turned off in power-down mode.

5.13.4. Deep power-down mode

In deep power-down mode the oscillator chip is turned off. The power input is thereby reduced to less than 1 mA.

The time necessary for restarting the clock to 25 MHz amounts to 9ms.

Return from deep power-down mode can take place only via an IRQ8. This is normally the IRQ of the RTC. The signal –IRQ8/TCLK0 on the MicroPC connector is used by the open drain output of the RTC. External open drain signal sources can be attached here and likewise produce an IRQ8.

6. PC Programs

6.1. VTERM

VTERM.EXE is the standard terminal program for the MicroPC and thus the most appropriate connection program for the MicroPC during software development.

6.1.1. Command-line parameters

VTERM can be called with the following command-line parameters:

-?	: Command-line parameter overview
-b(baud)	: Set data transmission rate
-c(1-4)	: Select serial port
-m	: Select black/white display
-o	: Open log file
-t(AHT)	: Select terminal emulation

6.1.2. VTERM commands

The following keys are assigned commands:

ALT-B	: Set data transmission rate
ALT-C	: Select serial port
ALT-D	: Assign remote drives
ALT-E	: Turn local ECHO on/off
ALT-F	: Set handshake
ALT-H	: Help
ALT-O	: Open/close log file
ALT-P	: Set data transmission parameter
ALT-R	: Receive file
ALT-S	: Send file
ALT-T	: Set terminal emulation
ALT-W	: Save settings
ALT-X	: Exit VTERM
ALT-Y	: Clear screen
ALT-Z	: DOS command

In addition to the usual terminal functions (output to the screen, input through the host PC's keyboard, and file transfer) VTERM permits direct access from the MicroPC to the PC's drives with the help of the TSR program RDRIVE.

6.1.3. File transfer with VTERM

Apart from file transmission via RDRIVE, which runs automatically, VTERM must be instructed explicitly to start other kinds of file transfers. This particularly affects file transfers by the BIOS Setup (flash update/backup) and communication with XSEND and XLOAD. Sending or receiving a file in VTERM is started with ALT-R or ALT-S, respectively. VTERM then asks for a transmission protocol and a file name.

Transmission to the MicroPC generally takes place via Xmodem protocol, therefore VTERM must also be set to Xmodem. Then one enters the name of the file to be sent, or the name of the file which should be received (Xmodem does not transfer the file name).

6.2. FLASHHDD

Flashhdd.exe is used to create a flash image file based on the contents of any directory. This file is then transferred via BIOS setup. Its contents constitute drive C: of the MicroPC. In order to be able to boot DOS, the affected directory must contain at least the file command.com.

Call:

FLASHHDD [/B<n>] [/S<m>] [/ V][/?] *<Source Directory> [<Destination File>]*

Options:

/B<n>	n = Number of blocks (default 14).
/S<m>	m = Block size in kB (64 in the case of the MicroPC).
/ V	Verbose (Show files and directories)
/?	Help
/M	MS-DOS compatible Flash-disk format (include MS-DOS system files)

The maximum value for n depends on the configuration, in accordance with the following table:

Total capacity	n
1 MB	14
2 MB	29
4 MB	61
8 MB	125

Exceeding the maximum value for n should be avoided, since DOS computes the capacity from the boot sector of the Flash disk. If DOS computes more capacity than is physically present, a BIOS error will occur when DOS attempts to access non-existing sectors, leading to a system hang-up.

Make sure that the target file is not inadvertently in the source directory (to avoid an error by recursion).

6.3. ROMDRV

Romdrv.exe permits the creation of a ROM disk image file based on the contents of any directory. This file is assigned drive A: in the MicroPC's Flash memory as per BIOS Setup.

Call:

romdrv <Source Directory> [<Destination File>]

Make sure that the target file is not inadvertently in the source directory (to avoid recursion errors).

6.4. Bin2hex

Bin2hex is a program for generating a Intel Hex86 file from a binary file (e.g. ".COM" file).

6.5. Hex2bin

Hex2bin generates a binary file from an Intel Hex86 file.

7. MicroPC Programs

7.1. Mapping Remote Drives with RDRIVE, RMAP and RMCWD

RDRIVE enables integration of the PC drives as drives of the MicroPC. The program is made resident after loading. Files can be transferred to and from the host PC like over a network, e.g. by the "copy" command. This is the MicroPC's standard technique for transmitting files. In addition, programs can be loaded directly from the host PC, without first copying them to a local MicroPC drive.

Call:

```
rdrive [-?] [-c<n>] [-u]
```

```
-?            : Help
-c<1..4>      : select COM port 1 to 4 (default: COM1)
-u            : remove RDRIVE from RAM
```

To change the serial interface with -c, RDRIVE must be cleared beforehand with -u.

With the program RMAP, drives and directories of the host PC can be mapped as drives of the MicroPC. LOCAL designates the drive letter of the MicroPC to be used, and REMOTE designates the drive or directory of the host PC to be used.

```
RMAP   /LOCAL=D   /REMOTE=C
```

makes, for example, drive C: of the host PC available as drive D: of the MicroPC. There is no firm rule about which letter must be used in which order -- independently of whether the remote drives are local or network drives of the host PC.

Mapping directories is just as easy:

```
RMAP   /LOCAL=E   /REMOTE=C:\Programs\Files386
```

The assignments can be overwritten at any time, or can be deleted by indicating the local drive letter:

```
RMAP   /LOCAL=D
```

The command RMAP alone generates a list of current drive assignments.

The program RMCWD.EXE automatically assigns the indicated drive letter to the directory from which Vterm was started.

7.2. XLOAD

Xload is a simple program for transferring files from the host to the RAM or Flash disk. The transmission protocol XMODEM is used. After the call of:

xload [com port] file

Xload waits until the transfer is started on the host side.

7.3. XSEND

Xsend is a simple program for transferring files from the MicroPC to the host. The transmission protocol XMODEM is used. After the call of:

xsend [com port] file

the transfer must be started from the host side.

7.4. ZTRANS

Ztrans offers extended functionality in relation to Xsend and Xload, in particular the transmission of the file name as well as the transmission of several files with one command. The underlying protocol is ZMODEM. This is not supported by VTERM. A ZMODEM-capable terminal program (e.g. Windows HyperTerminal) must be used instead, or Ztrans must also be started on the host.

Call:

ztrans *[/R] [/Bn] [/Cn] [/?] <File(s)>*

Options

/R	Receive instead of send
/Bn	Set baud rate
/Cn	Select interface
/?	Help

Wildcards are possible for < *File(s)*>.

8. BIOS Setup

The MicroPC offers a variety of settings for adapting it to the needs of the application. In order to enter setup, the **<S>** key must be pressed during the memory test. The memory test is then interrupted and the setup menu appears. A menu item can be selected here with the cursor keys, or with the TAB or Ctrl-E keys.

8.1. Main Setup

Date and Time: Setting the real-time clock (RTC). These values are stored after power-down only if a lithium battery is installed or another power supply is attached to the I/O connectors Vbatt pin.

Console Port: A serial interface to be addressed by the BIOS or ROM-DOS ("CON" device) is registered here. **Note:** if "none" or a nonexistent interface is set, then the setup and DOS prompt cannot be accessed after the next system start. This sometimes does make sense, in order to prevent unauthorized access to the MicroPC. Resetting the Console Port is however still possible by short-circuiting the ready pin during startup to GND (ground). The BIOS then automatically sets default values for the first serial interface and jumps to the setup menu.

Real Mode Flash: Here one sets how much CPU address space in the lowest megabyte is to be made available to the Flash memory. This memory range is then accessible in the 386EX CPU's Real Mode. Possible settings: 128KB (default), 256KB or 512KB. The associated address range is located at the end of the lowest megabyte. 128kB are used here by the operating system (BIOS and ROM-DOS).

Enable ROM Disk: The Flash memory range set here can be used for a non-writeable ROM disk or also as user addressable flash memory.

Enable RAM Disk: The RAM disk uses the free RAM range above 1MB, to the extent that it is not already made available below 1MB. With 1MB of RAM, at least 128KB are available for the RAM disk (according to the default size of the Real Mode Flash). This range can be increased by setting more Real Mode Flash.

CompactFlash 1 and 2: Up to two CompactFlash^{TM} memory cards or Microdrive^{TM} hard drives can be attached to the MicroPC. These can be operated in LBA mode or in auto-CHS mode. The mode set when formatting a CompactFlash card must always be used later, including when the CompactFlash card is inserted in other computers. Changing the mode requires reformatting the card.

CPU Clock (CLK2) in MHz: Available clock speeds are 50, 40, 16 and 8 MHz.

8.2. Advanced Setup

Power-on messages: When "disabled," the copyright message, RAM test messages, and the BIOS configuration box are suppressed.

System Configuration Box: Display of the BIOS Config box can be suppressed.

Display "Hit <S>..." : Display of the message "Hit <S> ..." can be suppressed.

Wait For Key on Error : The BIOS waits for a command if any error is detected during the system test at boot time. The BIOS setup can then be started to fix any inconsistency of setup settings.

Fast Boot: The BIOS conducts only a shortened RAM test (saves booting time).

ROM-DOS : activates or deactivates the ROM-DOS.

ROM-DOS Bootdrive : Determines the drive from which the ROM-DOS reads the command.com, config.sys and autoexec.bat files.

Flash File System : turns off the BIOS's Flash File System for the on-board Flash disk.

DOS/Non-DOS Flashdisk: "FAT monitoring" by the BIOS achieves a substantial speed increase for the on-board Flash disk's Flash File System. This however functions only under DOS. The FAT monitoring must be turned off when using another operating system.

SRAM and Flash Waitstates: the default value should not be changed usually. However, if the maximum CPU clock is not used, then another value can be given. This is calculated as follows:

$n \geq 0$

and $n \geq (t_R + 10ns) * f_{CPU} – 1{,}5$

where: $\mathbf{t_R}$ = RAM / Flash access time (as per data sheet),
n = number of wait states
$\mathbf{f_{CPU}}$ = oscillator frequency / 2 (≤ 25 MHz)

For t_R = 55 ns and f_{CPU} = 25 MHz the following applies:

$(t_R + 10ns) * f_{CPU} – 1{,}5 = 0{,}125$

The next-larger integral value for n = 0.125 is 1. The 55ns RAM thus functions with 1 wait state.

For t_R = 70 ns and f_{CPU} = 25 MHz the formula reads:

$(t_R + 10ns)* f_{CPU} – 1{,}5 = 0{,}5$

One wait state must therefore also be set for 70ns RAM.

For t_R = 120 ns and f_{CPU} = 25 MHz:

$(t_R +10ns) * f_{CPU} – 1{,}5 = 1{,}75$

Two wait states must thus be set for 120 ns Flash.

For t_R = 90 ns and f_{CPU} = 25 MHz:

$(t_R +10ns) * f_{CPU} – 1{,}5 = 1{,}0$

One wait state must thus be set for 90 ns Flash.

8.3. I/O Configuration Setup

Setting the serial interface:

Note: Changing the configuration settings for the serial interfaces can completely block access to the MicroPC (either intentionally or inadvertently). One should therefore know exactly which settings one changes for which purposes.

The BIOS and DOS support a maximum of four serial interfaces. These can be addressed by the user program through the BIOS via Int 14h or through DOS as devices COM1 to COM4. Additional serial interfaces are not supported by the BIOS and DOS; these must be programmed by direct access to the hardware.

COM PORTS: The respective UART type for each of the four possible COM ports is set with TYPE. BASE indicates the base address of the UARTs. The baud rate of the interface is set with BAUDRATE. The number of the data bits, parity and number of the stop bits are set with SETTING. Under INTERRUPTS one can indicate whether the interface should function in polling mode or with receipt interrupt. In this case, the correct interrupt line must be set. If one uses the interrupt mode, the size of the receive buffer used by the BIOS interrupt handler can be indicated in bytes with BUFFER.

PRINTER PORTS: The base address of the optional printer interfaces is set here.

X1 Connector Configuration: Definition of the function of individual I/O plug X1 pins. These can be configured as digital I/O (input, open drain output, or output), as timer functions, or as interrupts.

PORT INIT : It is possible here to implement circuit-dependent initializations of I/O ports (thus also of peripherals attached to the PIF bus.) very early in the boot process. One can indicate start-up values for a maximum of four I/O addresses. These I/O accesses are executed a few microseconds after a reset.

8.4. Flash Setup

Flash Update Setup

In the Flash Update Setup different ranges of the MicroPC's Flash memory can be reprogrammed. Existing content is deleted and reprogrammed with data loaded via the serial interface. Loading of new data takes place with the transmission protocol XMODEM. After selection of a menu option, the card begins to send a protocol character (§). After that, transmission with XMODEM must be started from the terminal program (in the case of VTERM: Alt-S, XMODEM, file name). After a successful download, depending upon selected range, the card is re-booted or returns to the setup menu.

Flash Backup Setup

Individual ranges of the Flash can be sent to the PC. After selecting a menu option, the "receive file" function of the terminal program must be started using XMODEM protocol (in the case of VTERM: ALT-R, XMODEM, file name).

Flash Erase Setup

Individual ranges of the MicroPC Flash can be erased selectively. The erasure takes place in a block by block fashion (a flash block sector includes 64kB). Erasing a block normally lasts less than a second.

8.5. Exit Setup

Exit and Save changes: End setup and save changes.

Exit and discard changes: End setup without saving changes.

Reset to previous values: All changes made since the start of the BIOS Setup are discarded.

Reset to default values: All BIOS Setup settings are changed to the standard values.

9. BIOS - Reference

9.1. INT 10h - Video Service

9.1.1. INT 10h Function 00h - Set video mode

Call:	AH	= 00h
	AL	= Video Mode
Return:	none	
Description:	Since the MicroPC does not have video hardware, this function serves only to clear the screen of the terminal PC.	

9.1.2. INT 10h Function 02h - Set cursor position

Call:	AH	= 02h
	DH	= line
	DL	= column
Return:	none	

9.1.3. INT 10h Function 03h - Get current cursor position

Call:	AH	= 03h
Return:	AX	= 00h
	DH	= line
	DL	= column

9.1.4. INT 10h Function 06h/07h - Scroll current page up/down

Call:	AH	= 05h/06h
	BH	= New color attribute
Return:	none	
Description:	Since the MicroPC does not have video hardware, this function serves only to clear the screen of the terminal PC with the given color attribute.	

9.1.5. INT 10h Function 09h - Write char/attribute to screen

Call:	AH	= 09h
	AL	= Character
	BL	= Color attribute
	CX	= Number of characters
Return:	none	
Description:	The character is output CX times with the indicated color attribute. The cursor position is not changed. Different than on a normal PC, the color attribute applies to all following outputs from the functions 00h, 0Ah and 0Eh.	

9.1.6. INT 10h Function 0Ah - Write character to screen

Call:	AH	= 0Ah
	AL	= Character
	CX	= Number of characters
Return:	none	
Description:	The character is output CX times. Other than on the screen of a normal PC, the cursor position is incremented since a terminal program does so automatically.	

9.1.7. INT 10h Function 0Eh - Write teletype to screen

Call:	AH	= 0Eh
	AL	= Character
Return:	none	
Description:	The character in AL is output, whereby the control characters 07h (Beep), 08h (backspace), 0Ah (line feed) and 0Dh (carriage return) are interpreted. This is the fastest way to send a character, since escape sequences do not have to be sent.	

9.2. INT 11h - Equipment Check Service

Call:	none		
Return:	AX	= Contents of 40:10h	
		Bits 15 -14	= Number of printers
		Bits 13 - 12	= Reserved
		Bits 11 - 9	= Number of diskettes
		Bit 8	= Reserved
		Bits 5 - 4	= Video mode
		Bit 3	= Reserved
		Bit 2	= Mouse installed
		Bit 1	= Coprocessor
		Bit 0	= Boot disk present
Description:	This function returns the content of memory cells 40:10h.		

9.3. INT 12h - Memory Size

Call:	none	
Return:	AX	= Contents of 40:13h
Description:	This function returns the content of memory cells 40:13h. This gives the free memory in Kilobytes.	

9.4. INT 13h - Disk Services

Since the MicroPC Flash disk is organized like a hard drive, the following also applies to it.

9.4.1. INT 13h Function 01h - Read disk status

Call:	AH	= 01h
	DL	= Drive (0 oder 1)
Return:	AH	= 0 No error
		= or error code
	CF	= 0 No error
		= 1 Error
Description:	Reads, and then resets, the last error code.	

9.4.2. INT 13h Function 02h - Read disk sectors

Call:	AH	= 02h
	AL	= Number of sectors
	CH	= Track
	CL	= Sector
	DH	= Head
	DL	= Drive (0 oder 1)
	ES:BX	= Pointer to the sector buffer
Return:	AH	= 0 No error
		= or error code
	AL	= Number of sectors read

	CF	= 0 No error
		= 1 Error
Description:	This function reads the indicated number of sectors into a buffer.	

9.4.3. INT 13h Function 03h - Write disk sectors

Call:	AH	= 03h
	AL	= Number of sectors
	CH	= Track
	CL	= Sector
	DH	= Head
	DL	= Drive (0 oder 1)
	ES:BX	= Pointer to the sector buffer
Return:	AH	= 0 No error
		= or error code
	AL	= Number of sectors read
	CF	= 0 No error
		= 1 Error
Description:	This function writes the given number of sectors to the disk.	

9.4.4. INT 13h Function 08h - Read drive parameter

Call:	AH	= 08h
	DL	= Drive (0 oder 1)
Return:	AX	= 0
	CH	= Last Track
	CL	= Last Sector
	DH	= Number of heads
	DL	= Number of installed drives
	ES:DI	= Pointer to Diskette Parameter Table
	CF	= 0 No error
		= 1 Error
Description:	This function returns a drive's parameters.	

9.5. INT 14h – Functions of the Asynchronous Serial Interfaces

9.5.1. INT 14h Function 00h – Initialize Serial Interface

Call:	AH	= 00h				
	AL	= Parameter				
		Bits 7 - 5	=	Baud rate:		
				000	=	110 Baud
				001	=	150 Baud
				010	=	300 Baud
				011	=	600 Baud
				100	=	1200 Baud
				101	=	2400 Baud
				110	=	4800 Baud
				111	=	9600 Baud
		Bits 4 - 3	=	Parity		
				x0	=	none
				01	=	uneven
				11	=	even
		Bit 2	=	Stop bits		
				0	=	1 stop bit
				1	=	2 stop bits
		Bit 1 - 0	=	Data word length:		
				00	=	5 bits
				01	=	6 bits
				10	=	7 bits

		11	= 8 bits
	DX	= Number of the serial interface (0 - 3)	
Return:	AH	= Line status as with function 1	
	AL	= Modem status	

9.5.2. INT 14h Function 01h – Send character

Call:	AH	= 01h	
	AL	= Character to be sent	
	DX	= Number of the serial interface (0 - 3)	
Return:	AL =	Character sent	
	AH =	Line status:	
		Bit 7 = 1 :	Timeout error
		Bit 6 = 1 :	Transmitter shift register empty
		Bit 5 = 1 :	Transmitter buffer register empty
		Bit 4 = 1 :	Break condition = RXD low for the duration of more than one word length
		Bit 3 = 1 :	Framing error = invalid stop bit
		Bit 2 = 1 :	Parity error
		Bit 1 = 1 :	Overrun error = receive buffer overwritten
		Bit 0 = 1 :	Receive data ready

9.5.3. INT 14h Function 02h – Receive character

Call:	AH = 02h	
Return:	AL =	Received character
	AH =	Line status as with function 1
	DX=	Number of the serial interface (0 - 3)

Remark: A timeout occurs after approximately 1 second. The serial interfaces are operated with interrupts or in polling mode, depending upon the applicable setting in the BIOS setup.

9.5.4. INT 14h Function 03h – Query the status of a serial interface

Call:	AH	= 03h
	DX	= Number of the serial interface (0 - 3)
Return:	AH	= Line status as with Function 1

9.5.5. INT 14h Function 04h - Extended Initialization

Call:	AH	= 04h		
	BH	= Parity		
			00h	= no parity.
			01h	= odd parity.
			02h	= even parity.
	BL	- Stop bits		
			00h =	1 stop bit
			01h =	2 stop bits or 1,5 in the case of 5 databits
	CH	- Data length		
			00h =	5 bits
			01h =	6 bits
			02h =	7 bits
			03h =	8 bits
	CL	- Baud rate		
			00h =	110 Baud
			01h =	150 Baud
			02h =	300 Baud
			03h =	600 Baud
			04h =	1200 Baud
			05h =	2400 Baud
			06h =	4800 Baud

		07h =	9600 Baud
		08h =	19200 Baud
		09h =	38400 Baud
		0Ah =	57600 Baud
		0Bh =	115200 Baud
	DX	= Number of the serial interface (0 - 3)	
Return:	AH	= Line status as with function 1	
	AL	= Modem status	

Remark: This function permits higher baud rates than function 00h.

9.6. INT 15h - System Services

9.6.1. INT 15h Function 24h - A20 gate control

9.6.2. INT 15h Function 87h - Move memory block

9.6.3. INT 15h Function C0h - Get system config table

Call:	AH	= C0h
Return:	AH	= 00h
	ES:BX	= Address of System Config Table

Description: This function delivers the address of the System Configuration Table.

9.6.4. INT 15h Function A1h - Int 10h / Int 16h redirect I/O

This function redirects the BIOS Int 10h video output and the Int 16h keyboard input functions to a serial interface. The parameter 0 (none) turns off all Int 10h and Int 16h input and output operations.

Call:	AH	= A1h
	BX	= COM port (1 = COM1, ..., 4 = COM4, 0 = none)

9.7. INT 15h Function C3h - Functions Specific to the MicroPC

9.7.1. INT 15h Function C301h – Enable Watchdog

Call:	AH	= C3h
	AL	= 01h
	BX	= Duration of the Watchdog timeout in milliseconds
Return:	--	

Description: This function activates the watchdog. Subsequently, the watchdog must be reset at least once within the watchdog timeout limit, otherwise a system reset is triggered.

Once released, the watchdog can be deactivated only by a system reset. The timeout duration can no longer be changed, either.

9.7.2. INT 15h Function C302h – Reset Watchdog

Call:	AH	= C3h
	AL	= 02h
Return:	--	

Description: After activation of the watchdog, this function must be called at least once within the watchdog timeout limit, otherwise a system reset is triggered.

9.7.3. INT 15h Function C310h – Query processor clock rate

Call:	AH	= C3h
	AL	= 10h

Return:	AL	= 3 : CPU clock rate= 25 MHz (CLK2 = 50 MHz)
		= 2 : CPU clock rate= 20 MHz (CLK2 = 40 MHz)
		= 1 : CPU clock rate= 8 MHz (CLK2 = 16 MHz)
		= 0 : CPU clock rate= 4 MHz (CLK2 = 8 MHz)

Beginning with board revision 2, the following clock rates are also possible:

= 7 : CPU clock rate=	16.67 MHz	(CLK2 = 33.33 MHz)
= 6 : CPU clock rate=	12.5 MHz	(CLK2 = 25 MHz)
= 5 : CPU clock rate=	10 MHz	(CLK2 = 20 MHz)
= 4 : CPU clock rate=	6.5 MHz	(CLK2 = 13 MHz)

9.7.4. INT 15h Function C311h – Set processor clock rate

Call:	AH	= C3h
	AL	= 11h
	BL	= Clock rate as defined for Function C310h above

Return: --

Remark: The prescaler (CLKPRS) for the input clock rate of Timer 0 is adjusted according to the new clock rate, so the Timer 0 interrupt period will not change.

9.7.5. INT 15h Function C312h – Set CPU in idle mode

Call:	AH	= C3h
	AL	= 12h
	BL	= *Clock rate*

Remark: The CPU core is stopped, but the 386EX's internal peripherals (in particular the timers and the SSIO) keep running. Return from idle mode is possible only by a hardware interrupt.

9.7.6. INT 15h Function C313h – Set CPU in power-down mode

Call:	AH	= C3h
	AL	= 13h
	BL	= Clock rate as with Function C310h, or
	BL	= 80h: Deep Power-Down Mode

Remark: The CPU core and the 386EX's internal peripherals CPU core are stopped. The timers continue to run only if they are operated with an external clock. The SSIO functions only in slave mode. In the MicroPC, the UARTs work with their own clock, and thus will still work in power-down mode.

Return from the power down mode is done by RTC interrupts, serial interface interrupts, or external IRQs. The interrupts of Timer 0 and Timer 1 terminate the power-down mode only if operated with an external clock. Since Timer 2 can be operated only with internal clock on the MicroPC, it cannot trigger a return from power-down mode.

In deep power-down mode the clock is completely turned off. This provides the lowest power consumption. The return from deep power-down mode can be accomplished only by the RTC IRQ, or likewise an external signal connected with the RTC IRQ.

9.7.7. INT 15h Function C314h – Synchronous serial interface: Send

Call:	AH	= C3h
	AL	= 14h
	BX	= Data word
	CL	= 4..127: Divisor for Baud rate

Remark: For this and the following function the synchronous serial interface works exclusively in master mode without interrupts. The baud rate results from the CPU input clock rate CLK2 (normally 50 MHz for the MicroPC) according to the formula:

Baud rate = CLK2 / (8 x (CL + 1))

There are also possible baud rates between 1.25 MBaud and 48.8 kBaud. Changing the CPU clock causes an according change of the baud rate.

9.7.8. INT 15h Function C315h – Synchronous serial interface: Receive

Call:	AH	= C3h
	AL	= 15h
	CL	= 4..127: Divisor for Baud rate
Return:	AX	= Data word

9.7.9. INT 15h Function C320h – Reading EEPROM

Call:	AH	= C3h
	AL	= 20h
	BH	= Address in EEPROM
Return:	AL	= Data byte

Description: This function reads the data byte at the given address in EEPROM.

9.7.10. INT 15h Function C321h – Writing EEPROM

Call:	AH	= C3h
	AL	= 21h
	BH	= Address in EEPROM
	BL	= Data byte

Description: This function writes the data byte to the given address in EEPROM.

9.7.11. INT 15h Function C322h - I2C bus: Read byte with address byte

Call:	AH	= C3h
	AL	= 22h
	BH	= I2C bus Address
	CH	= I2C-chip Address
Return:	AL	= I2C bus data byte reading
	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.12. INT 15h Function C323h - I2C bus: Write byte with address byte

Call:	AH	= C3h
	AL	= 23h
	BH	= I2C bus address
	BL	= I2C bus data byte
	CH	= I2C-chip address
Return:	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.13. INT 15h Function C324h - I2C bus: Read data block with 2 address bytes

Call:	AH	= C3h
	AL	= 24h
	BX	= I2C bus address (BH = first address byte, BL = second address byte)
	CH	= I2C-chip address
	CL	= Number of bytes to be read (0 == 256)
	ES:DI	= Far pointer to read buffer in RAM
Return:	ES:[DI]	= data bytes read
	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.14. INT 15h Function C325h - I2C bus: Write data block with 2 address bytes

Call:	AH	= C3h
	AL	= 25h
	BX	= I2C bus address (BH = first address byte, BL = second address byte)
	CH	= I2C-chip address
	CL	= Number of bytes to be written (0 == 256)
	ES:SI	= Far pointer to write buffer
Return:	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.15. INT 15h Function C326h – Request I2C bus

Call:	AH	= C3h
	AL	= 26h
Return:	Carry Flag = 0: I2C bus successfully requested	
	Carry Flag = 1: I2C bus was in use	

Description: Since I2C bus cycles may not overlap in the MicroPC, the bus should be requested before each transmission. The associated I2C bus flag is set and prevents further I2C bus functions from interrupting the current function. This is of importance only if I2C functions are to be implemented within interrupt routines.

9.7.16. INT 15h Function C327h – Enable I2C bus

Call:	AH	= C3h
	AL	= 27h

Description: The I2C bus flag is reset. If the I2C bus was requested before with the function C326h, it must be reset with this function, otherwise no further I2C bus access is possible.

9.7.17. INT 15h Function C328h – I2C Bus: Read byte with 2 address bytes

Call:	AH	= C3h
	AL	= 28h
	BH	= I2C bus address 0
	CH	= I2C chip address
	CL	= I2C bus address 1
Return:	AL	= I2C bus data byte reading
	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.18. INT 15h Function C329h - I2C bus: Write byte with 2 address bytes

Call:	AH	= C3h
	AL	= 29h
	BH	= I2C bus address 0
	BL	= I2C bus data byte
	CH	= I2C chip address
	CL	= I2C bus address 1
Return:	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.19. INT 15h Function C32Ah – I2C bus: Read byte

Call:	AH	= C3h
	AL	= 2Ah
	CH	= I2C chip address
Return:	AL	= I2C bus data byte reading
	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.20. INT 15h Function C32Bh - I2C bus: Write byte

Call:	AH	= C3h
	AL	= 2Bh
	BL	= Data byte
	CH	= I2C chip address
Return:	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.21. INT 15h Function C32Ch - I2C bus: Read two bytes

Call:	AH	= C3h
	AL	= 2Ch
	CH	= I2C-Chip Address
Return:	AX	= Data word reading (AH = first byte, AL = second byte)
	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.22. INT 15h Function C32Dh - I2C bus: Write two bytes

Call:	AH	= C3h
	AL	= 2Dh
	BH	= First data byte
	BL	= Second data byte
	CH	= I2C chip address
Return:	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.23. INT 15h Function C32Eh - I2C bus: Read data block

Call:	AH	= C3h
	AL	= 2Eh
	CH	= I2C chip address
	CL	= Number of bytes to be read (0 == 256)
	ES:DI	= Far pointer to read buffer in RAM
Return:	[ES:DI]	= Data read
	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.24. INT 15h Function C32Fh - I2C bus: Write data block

Call:	AH	= C3h
	AL	= 2Fh
	CH	= I2C-Chip Address
	CL	= Number of bytes to be read (0 == 256)
	ES:SI	= Far pointer to write buffer
Return:	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

9.7.25. INT 15h Function C330h – Query hardware serial number

Call:	AH	= C3h
	AL	= 30h
Return:	AX	= lower serial number data word
	BX	= middle serial number data word
	CX	= upper serial number data word

Description: This Function reads the six bytes of the MicroPC's hardware serial number. Each MicroPC unit has its own unique serial number.

9.8. INT 16h - Keyboard Service

9.8.1. INT 16h Function 00h - Read keyboard input

Call:	AH	= 00h
Return:	AH	= Scan code extended keys
	AL	= Key value

Description: This function reads in a key. Only some extended keys are supported, since ANSI escape sequences are used for this (see chapter 10).

9.8.2. INT 16h Function 01h - Read keyboard status

Call:	AH	= 01h
Return:	ZF	= 1 - No character present
		= 0 - Character present

Description: This function ascertains whether a character is in the keyboard buffer. Different from a normal PC, the character in the buffer is not returned.

9.8.3. INT 16h Function 05h – Simulate pressing of a key

Call:	AH	= 05h
	CH	= Key's scan code
	CL	= Key's ASCII code
Return:	AL	= 0 – Function succeeded
		= 1 – Keyboard buffer was full

Description: A user program can write values to the BIOS keyboard buffer with this function.

9.9. INT 17h - Parallel Service

9.9.1. INT 17h Function 00h - Print character

Call:	AH	= 00h	
	AL	= Character	
	DX	= LPT port (0 - 2)	
Return:	AH	= Printer status	
		Bit 7	= 1 Printer available
		Bit 6	= 1 Acknowledgment
		Bit 5	= 1 Out of paper
		Bit 4	= 1 Printer selected
		Bit 3	= 1 Printer error
		Bit 2 - 1	= Reserved
		Bit 0	= Timeout error

Description: This function outputs a character.

9.9.2. INT 17h Function 01h - Initialize printer

Call:	AH	= 01h	
	DX	= LPT Port (0 - 2)	
Return:	AH	= Printer Status	
		Bit 7	= 1 Printer available
		Bit 6	= 1 Acknowledgment
		Bit 5	= 1 Out of paper
		Bit 4	= 1 Printer selected
		Bit 3	= 1 Printer error
		Bit 2 - 1	= Reserved
		Bit 0	= Timeout error

Description: This resets the printer.

9.9.3. INT 17h Function 02h - Get printer status

Call:	AH	= 02h
	DX	= LPT Port (0 - 2)
Return:	AH	= Printer Status
		Bit 7 = 1 Printer nicht besetzt
		Bit 6 = 1 Acknowledgment
		Bit 5 = 1 Out of paper
		Bit 4 = 1 Printer selected
		Bit 3 = 1 Printer error
		Bit 2 - 1 = Reserved
		Bit 0 = Timeout error
Description:	The status of the printer is read out.	

9.10. INT 18h - Boot Failure

Description: This function is activated after unsuccessful boot attempts.

9.11. INT 19h - Boot System

Description: This function is activated after a complete initialization of the BIOS. It is activated to execute its own routine by the ROM-DOS. If ROM-DOS is not present or is deactivated, the BIOS default handler tries to load the operating system from the flashdisk or ROM disk. If this fails, an INT 18h is executed.

9.12. INT 1Ah - Clock and Timer Functions

9.12.1. INT 1Ah Function 00h - Read system timer

Call:	AH	= 00h
Return:	AH	= 00h
	AL	= 24h Overrun Flag
	CX:DX	= System ticks since midnight

Description: This function reads the system timer variable at address 40h:6Ch. This 32-bit variable is incremented by the timer 0 interrupt 18.2 times per second. It is set to zero every 24 hours (which is supposed to happen at midnight). At boot time, the system timer variable is synchronized with the real time clock (see function 9). The DOS system time is derived from this variable.

9.12.2. INT 1Ah Function 01h - Set system timer

Call:	AH	= 01h
	CX:DX	= new value for timer variable
Return:	AH	= 00h

Description: This function sets the system timer variable at address 40h:6Ch.

9.12.3. INT 1Ah Function 02h - Read real-time clock

Call:	AH	= 02h
Return:	AH	= 00h
	AL	= Hours BCD
	CH	= Hours in BCD
	CL	= Minutes in BCD
	DH	= Seconds in BCD
	CF = 0: OK	
	CF = 1: Error	

Description: This function reads out the time from the RTC.

9.12.4. INT 1Ah Function 03h - Set real-time clock

Call:	AH	= 03h
	AL	= Hours in BCD
	CH	= Hours in BCD
	CL	= Minutes in BCD
	DH	= Seconds in BCD
Return:	AH	= 00h
	CF	= 0: OK
	CF	= 1: Error

Description: This function sets the time of the RTC.

9.12.5. INT 1Ah Function 04h – Read RTC date

Call:	AH	= 04h
Return:	CH	= Century (19 or 20)
	CL	= Year
	DH	= Month
	DL	= Day

Description: This function reads out the date from the RTC.

9.12.6. INT 1Ah Function 05h – Set RTC date

Call:	AH	= 05h
	CH	= Century (19 or 20)
	CL	= Year
	DH	= Month
	DL	= Day
Return:	CF	= 0: OK
	CF	= 1: Error

Description: This function sets the date of the RTC.

9.12.7. INT 1Ah Function 06h – Set / Enable RTC interrupt

Call:	AH	= 06h
	CH	= Hour
	CL	= Minute
	DH	= Second

Return: CF = 0: OK
CF = 1: Error (e.g., an interrupt is already programmed)

Description: The real-time clock generates an interrupt at the programmed time on the same day. This is an impulse of approximately 10…40ms duration. The user program can set the 4Ah interrupt vector to a function of the program, which will then be called every time the RTC IRQ is activated. If this interrupt is only used to return from power-down mode, linking of the 4Ah interrupt is not necessary.

The values for hour, minute and second must be coded in BCD. Only one single interrupt time is valid at any time. If an interrupt time is already programmed, it must first be disabled with Function 07h.

9.12.8. INT 1Ah Function 07h – Disable RTC interrupt

Call: AH = 07h

Return: CF = 0: OK
CF = 1: Error

Description: A programmed interrupt time can be cleared with this function. The RTC interrupts will then cease to be generated. This function must also be called in order to change the interrupt time; a new time can be programmed with Function 06h only after calling this function.

9.12.9. INT 1Ah Function 08h : Synchronize system timer

Call:	AH = 08h

Return:	CF = 0: OK
	CF = 1: Error

Description:	This function sets the system timer variable at address 40h:6Ch according to the real time clock. This 32-bit variable is incremented by the timer 0 interrupt 18.2 times per second.

9.12.10. INT 1Ah Function 09h : Set cyclical interrupt

Call:	AH = 09h AL = 0: Interrupt every 1/4096 seconds 1: Interrupt each second 2: Interrupt each minute 3: Interrupt each hour 4: Interrupt each day

Return:	CF = 0: OK
	CF = 1: Error

Remark:	This function is hardware-dependent, and thus not compatible to a normal PC's BIOS.

9.13. INT 1Bh to 1Fh

These interrupt vectors do not point to an executable function, but to various BIOS tables.

9.14. INT 5Fh - Flash Services

9.14.1. INT 5Fh Function 00h - Erase Flash block

Call :	AH	= 00h
	DX:DI	= 32-bit block start address
		= 32-bit Flash start address + 10000h * block number

depending on whether one or two Flash ICs are installed

Return:	Carry Flag = 0: No error Carry Flag = 1: Error
Description:	(See also Function 02h). Deletion of a 64kB Flash block.

9.14.2. INT 5Fh Function 01h - Read Flash block

Call:	AH	= 01h
	DX:DI	= 32-bit source address (first byte to be read)
		= Flash start address + 10000h * block number + offset
	ES:BX	= Target address (Far pointer to data buffer in RAM)
	CX	= Number of bytes to be read
Return:	[ES:BX]	= Data reading
	Carry Flag = 0: No error	
	Carry Flag = 1: Error	

Description:	"Offset" indicates the start address relative to the beginning of the block. The source address is a 32-bit address, since the Flash memory is addressed in protected mode. The target address (in RAM) is a real-mode address, and thus in the form *Segment:Offset*. Flash start depends on the configuration: 1MB: 3F00000h, 2MB: 3E00000h, 4MB: 3C00000h, 8MB: 3800000h.

9.14.3. INT 5Fh Function 02h - Write Flash block

Call:	AH	= 02h
	DX:DI	= 32-bit target address

	ES:BX = source address (Far pointer to data buffer) CX = number of bytes to write
Return:	Carry Flag = 0: No error Carry Flag = 1: Error
Description:	(See also Function 02h). This function does not implement a delete operation. If the range to be written is not cleared, the function returns an error.

9.14.4. INT 5Fh Function 03h - Erase and write Flash block

Call:	AH = 03h DX:DI =32-bit target address ES:BX = Source address (Far pointer to data buffer) CX = Number of bytes to be written
Return:	Carry Flag = 0: No error Carry Flag = 1: Error
Description:	(See also Function 02h). The relevant Flash block will be cleared before being written to.

9.14.5. INT 5Fh Function 04h - Read Flash chip and manufacturer ID

Call:	AH = 04h DX:DI = 32-bit source address = 03F0:0000 ES:BX = target address (Far pointer to data buffer in RAM)

Return:	[ES:BX] = Device ID
	[ES:BX + 2] = Manufacturer ID
	Carry-Flag = 0: No error
	Carry-Flag = 1: Error

Description:	Reads out the Flash-IC's ID code. Depending upon the specific installation, the following ID codes are, for example, possible (the list is not complete): 29LV800BT: 22DAh 29LV160BT: 22C4h 29LV320DT: 22F6h 29LV641D : 22D7h Manufacturer: AMD: 01, Fujitsu: 04, ST: 20h

10. Tables

10.1. I/O Addresses

Port Address	Function
000h – 00Fh	DMA Controller *
020h – 021h	1. Interrupt Controller (Master)
022h – 023h	386EX Control Register
040h – 043h	Timer 0, 1 and 2
080h – 083h	DMA Page Register *
092h	A20 Gate, CPU Reset
0A0 – 0A1	2. Interrupt Controller (Slave)
100h – 103h	**PIF-Bus Configuration:**
100h	Data port (PD0..PD7)
101h	Control port (PA0..PA6)
102h	I/O Configuration (1 = Output, 0 = Input) Bit0: PD0..PD3, Bit1: PD4..PD7, Bit2: PA0..PA3, Bit3: reserved Bit4: PA4 (PIF-RD), Bit5: PA5 (PIF-WR) Bit6: PA6 (PIF-RDY) Bit7: PIF-Bus Fast Mode For valid PIF-bus Cycles, set port 102h to 3Fh or BFh
103h	Bit 4: Toggle PA4, Bit 5: Toggle PA5, Bit 6: Toggle PA6 Bit 0..3, Bit 7: reserved
2F8h – 2FFh	COM2
300h – 30Fh	PIF CS0
310h – 31Fh	PIF CS1
320h – 32Fh	PIF CS2
330h – 33Fh	PIF CS3
3F8h – 3FFh	COM1
F000h – FFFFh	**386EX Peripherals:**
F480h – F48Bh	Synchronous Serial Interface
F860	I/O-Port P1 Input
F862	I/O-Port P1 Output
F864	I/O-Port P1 Direction
F868	I/O-Port P2 Input
F86A	I/O-Port P2 Output
F86C	I/O-Port P2 Direction
F870	I/O-Port P3 Input
F872	I/O-Port P3 Output
F874	I/O-Port P3 Direction

* DMA is not supported by the MicroPC .

10.2. Interrupt Table

Vector	Address	Usage	Service Routine via	Type
00	000	Divide by Zero	BIOS or Application	CPU Exception
01	004	Single-Step Debug Interrupt	Debugger	CPU Exception
02	008	Non-Maskable Interrupt (NMI)	Application	Hardware NMI
03	00C	One-Byte Debug Breakpoint	Debugger	CPU Command
04	010	Overflow (called with INTO command)	Application	CPU Exception
05	014	Array Bounds Check (called with BOUNDS command)	Application	CPU Exception
06	018	Invalid Command Code	Debugger	CPU Exception
07	00C	Coprocessor not available	Application	CPU Exception
08	020	Timer 0 (System Timer)	BIOS	Hardware IRQ0
09	024	PIF-bus IRQ (PIF INT)	Application	Hardware IRQ1
0A	028	Reserved	BIOS	Software
0B	02C	Serial Interface (COM2)	BIOS or Application	Hardware IRQ3
0C	030	Serial Interface (COM1)	BIOS or Application	Hardware IRQ4
0D	034	Available for Application	Application	Hardware IRQ5
0E	038	Available for Application	Application	Hardware IRQ6
0F	03C	Available for Application	Application	Hardware IRQ7
10	040	Video Functions	BIOS	Software
11	044	System Configuration (Equipment Check)	BIOS	Software
12	048	RAM Size	BIOS	Software
13	04C	Disk Functions	BIOS	Software
14	050	Serial Port Functions (COM1-4)	BIOS	Software
15	054	Diverse, including MicroPC-specific functions	BIOS	Software
16	058	Keyboard Functions	BIOS	Software
17	05C	Parallel Port Functions (LPT)	BIOS	Software
18	060	Boot Error	BIOS	Software
19	064	Boot Loader	BIOS or DOS	Software
1A	068	System-Timer and RTC Functions	BIOS	Software
1B	06C	Reserved	BIOS	Software
1C	070	Timer User Function (called from Int 08h)	Application	Software
1D	074	Reserved	BIOS	Software
1E	078	Disk Parameter Table	--	BIOS Table
1F	07C	Reserved	BIOS	Software
20 – 3F	080 – 0FC	Reserved for DOS	DOS	Software
40 – 49	100 – 124	Reserved for BIOS	BIOS	Software
4A	128	RTC User Function (called from Int 70h)	Application	Software
4B – 5E	12C – 178	Reserved for BIOS	BIOS	Software
5F	17C	Flash Functions	Application	Software
60 – 6F	180 –1BC	Available for Application	Application	Software

Interrupt Table (cont.)

Vector	Address	Usage	Service Routine via	Type
70	1C0	Real-Time Clock (RTC) Alarm	BIOS	Hardware IRQ8
71	1C4	Synchronous Serial Interface or Available for Application	BIOS or Application	Hardware IRQ9
72	1C8	Timer 1	LCD-BIOS Extension or Application	Hardware IRQ10
73	1CC	Timer 2	Application	Hardware IRQ11
74	1D0	Reserved (DMA)	--	Hardware IRQ12
75	1D4	Available for Application	Application	Hardware IRQ13
76	1D8	IDE (Compact-Flash)	BIOS	Hardware IRQ14
77	1DC	386EX Watchdog Timer	Application	Hardware IRQ15
78 – FF	1E0 – 3FC	Available for Application	Application	Software

10.3. Pin Allocation

Pin No.	Functions								
	PIF bus	Timer	SIO	SSIO	Digital Port	IRQ	I²C bus[1]	JTAG	Other
01	GND								
02	D0				PD0				
03	D2				PD2				
04	D4				PD4				
05	D6				PD6				
06	/WR				PA5				
07	A0				PA0				
08	A2				PA2				
09	/CS0				P2.0				
10	/CS2				P2.2				
11	/RESET								
12					P3.3	IRQ5	x		
13	VCC								
14		TOUT0			P3.0	IRQ4[2]	x		
15			TXD1						
16		TCLK1	CTS1			IRQ13			
17			DSR1	STXCLK					
18			RI1	SSIORX					
19			RXD0		P2.5		x		
20			CTS0		P2.7		x		
21			DTR0		P1.2		x		
22			RI0		P1.4		x		
23								TMS	
24		/TCLK0				/IRQ8[3]		TDI	Wakeup
25	GND								
26	GND								
27	D1				PD1				
28	D3				PD3				
29	D5				PD5				
30	D7				PD7				
31	/RD				PA4				
32	A1				PA1				
33	A3				PA3				
34	/CS1				P2.1		x		
35	/CS3				P2.3		x		
36					P3.2	IRQ1	x		
37					P3.5	IRQ7	SCL		
38	VCC								
39		TOUT1	DCD1		P3.1	IRQ3[4]	x		
40			RXD1						
41			RTS1	SSIOTX					
42			DTR1	SRXCLK					
43			TXD0		P2.6		x		
44			RTS0		P1.1		x		
45		TGATE0	DSR0		P1.3	IRQ9[5]	x		
46		TGATE1	DCD0		P1.0	IRQ14[6]	x		
47									VBATT
48	READY				PA6			TCK	Emergency Boot
49		TGATE2						TDO	
50	GND								

[1] Digital ports usable by BIOS as I²C-bus port (SCL, SDA). Pin 37 is also used for the internal I²C bus as SCL signal.
[2] Occupied by SIO-0 (COM1)
[3] Occupied by RTC
[4] Occupied by SIO-1 (COM2)
[5] Occupied by SSIO
[6] Occupied by external CompactFlash card

10.4. ANSI Escape Sequences Used by the BIOS

The following escape sequences are used by the video BIOS (INT 10h):

Escape Sequence	Meaning	INT 10h
ESC[# ; # H	Set cursor position	02h
ESC[6 n	Status request	03h
ESC[s	Save cursor position	09h, 0Ah
ESC[u	Restore cursor position	09h, 0Ah
ESC[2J	Erase screen	00h, 0Eh
ESC[# ; ... ; # m	Set colors	06h, 07h, 09h

The following escape sequences are recognized by the keyboard BIOS (INT 16h):

Taste	Normal	+ Shift	+ Ctrl	+ Alt
F1	ESC[M	ESC[Y	ESC[k	ESC[w
F2	ESC[N	ESC[Z	ESC[l	ESC[x
F3	ESC[O	ESC[a	ESC[m	ESC[y
F4	ESC[P	ESC[b	ESC[n	ESC[z
F5	ESC[Q	ESC[c	ESC[o	ESC[@
F6	ESC[R	ESC[d	ESC[p	ESC[[
F7	ESC[S	ESC[e	ESC[q	ESC[\
F8	ESC[T	ESC[f	ESC[r	ESC[]
F9	ESC[U	ESC[g	ESC[s	ESC[^
F10	ESC[V	ESC[h	ESC[t	ESC[_

Left	ESC[D	Home	ESC[H
Right	ESC[C	End	ESC[F
Up	ESC[A	PgUp	ESC[I
Down	ESC[B	PgDown	ESC[G
Insert	ESC[L		

10.5. Electrical Data

Ambient temperature 25°C, unless otherwise indicated.

Symbol	Description	Parameter	Min.	Typ.	Max.	Unit
Vcc	Operating Voltage		3.0	3.3	3.6	V
Vres	Reset Threshold			2.9		V
tres	Duration of Reset Impulse		150		280	ms
V_{IH}	High-Level Input Voltage	PD0..PD7, PA0..PA3, PIF-RD, PIF-WR, PIF-RDY,	2		5.5	V
V_{IH}	High-Level Input Voltage	all other digital signals	2		Vcc+ 0.3	V
V_{IL}	Low-Level Input Voltage		0		0.8	V
Icc	Operating current in normal operation	CLK2 = 50 MHz *		135		mA
		CLK2 = 40 MHz		107		mA
		CLK2 = 16 MHz		48		mA
		CLK2 = 8 MHz		27		mA
	in idle mode	CLK2 = 50 MHz				mA
		CLK2 = 40 MHz				mA
		CLK2 = 16 MHz				mA
		CLK2 = 8 MHz		8.5		mA
	in power-down mode	CLK2 = 8 MHz		4.4		mA
		CLK2 = 0		0,30		mA
Vbatt	Battery Voltage (for SRAM and RTC)		2,0	3	3.6 (Vcc)	V
Ibatt	Battery current with operating voltage turned off	Equipped with 512kB SRAM Battery Voltage = 3V Ambient temp. = 25°C		3		µA
		Ambient temp. = 70°C			17	µA
		Ambient temp. = 85°C			22	µA

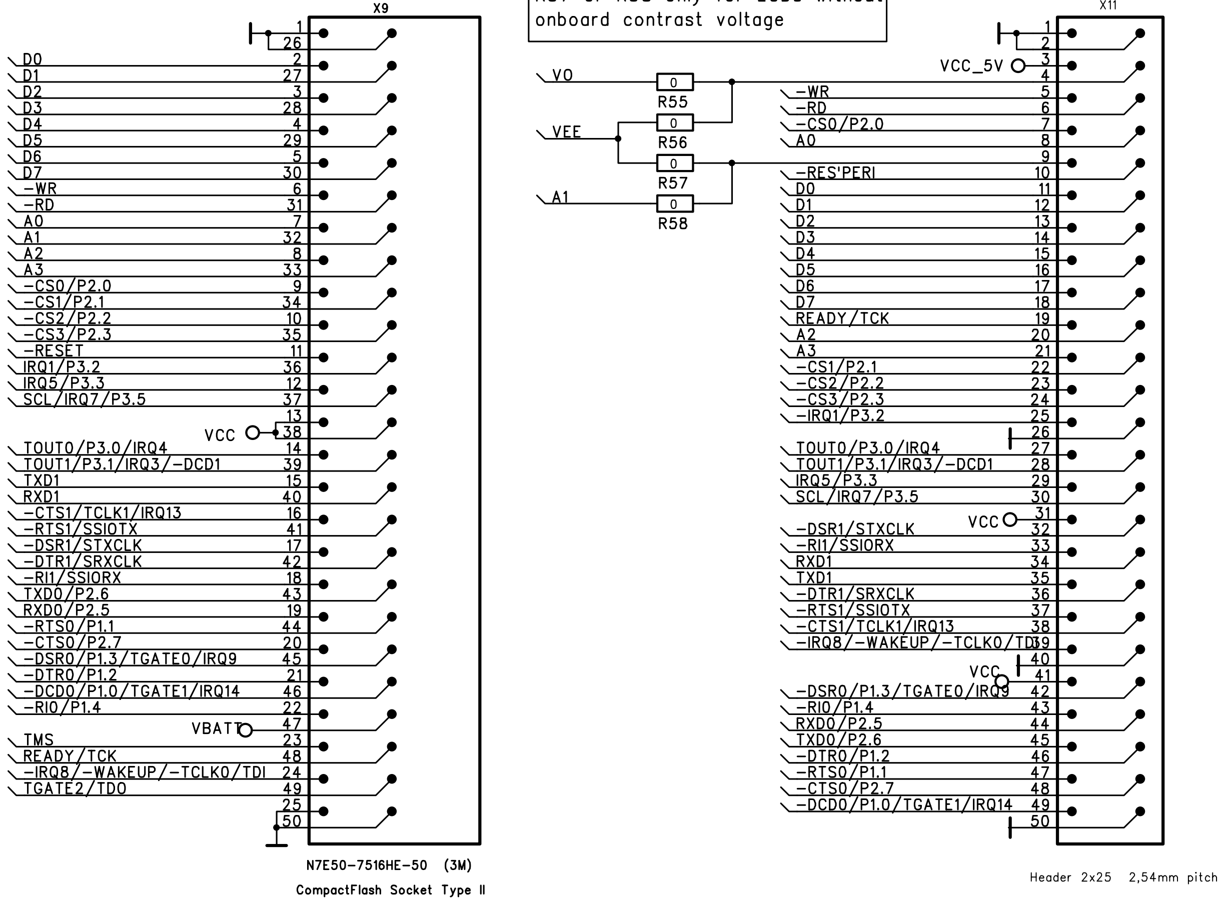
Remark:

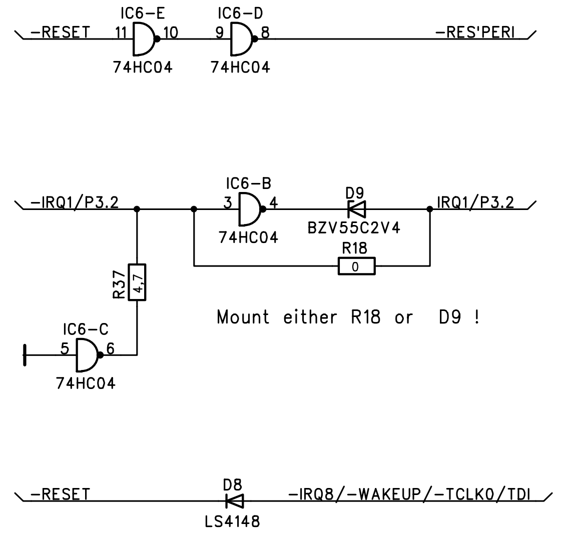
* The nominal clock rate – according to which the command times of the CPU are given – is 50% of the CLK2 oscillator frequency.

11. Circuit Diagram for MicroPC Starter-Kit Board

11.1. BUS Connector

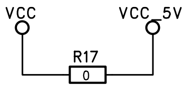
Normal Assembly: R55 and R58.
R57 or R58 only for LCDs without onboard contrast voltage



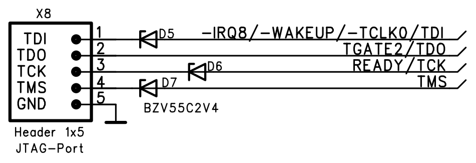


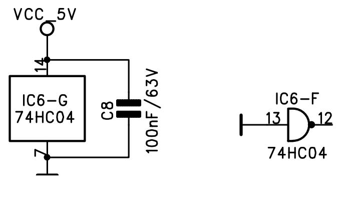
Mount either R18 or D9 !

D8 is not necessary for MicroPC PCB rev. 02



R17 is only mounted if 3.3V should be used for VCC_5V
(do not mount L2 in this case)



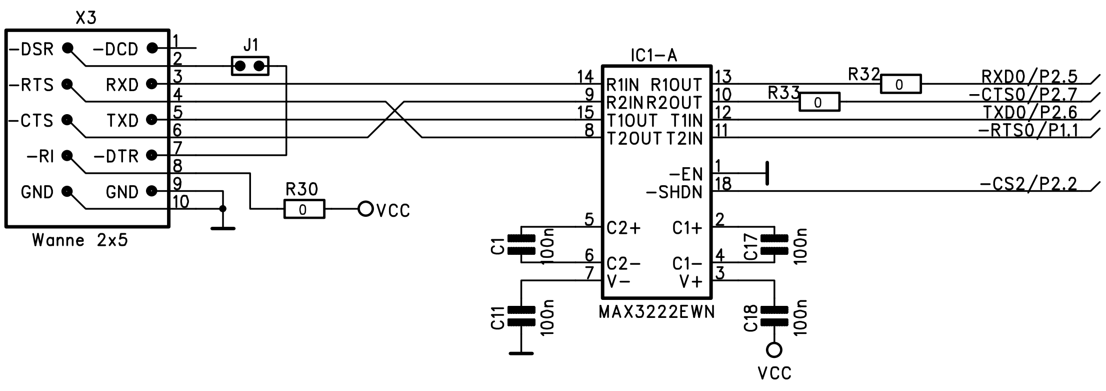


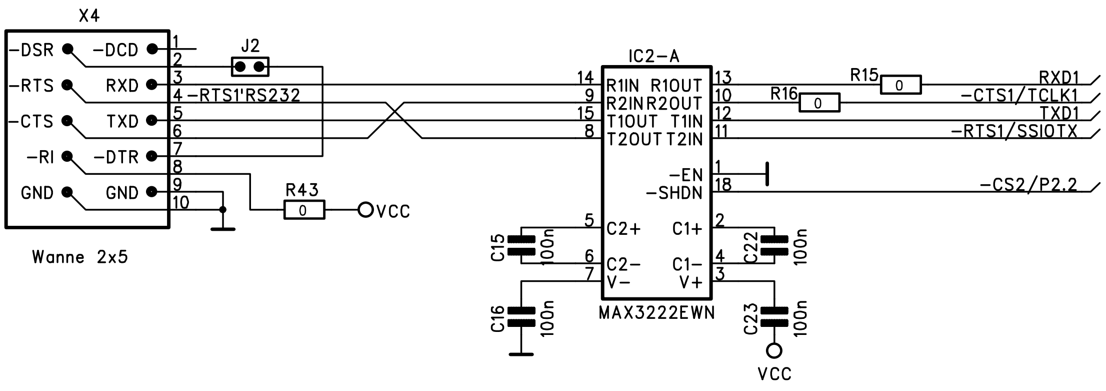
11.2. RS232 Driver

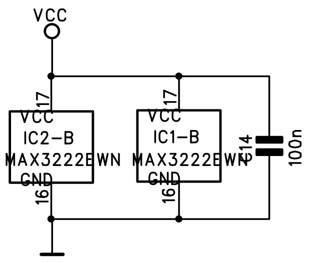
Both RS232 driver ICs (IC1 and IC2) can be switched into standby mode via a digital port output. Standby mode uses only a few µA current. Port -CS2 serves this purpose in connection with R35. In the event that -CS2 is used as a PIF-bus chip select, R36 should be equipped.

Bridges R30 and R43 make it possible to put a supply voltage of 3.3V on the SIO connector if the the RI signal is not used.

If the signals of a serial interface are to be used on the X1 connector, the signals RXD and CTS of the respective interface can be disconnected from the RS232 driver to avoid signal contention. This is done by means of bridges R32, R33, R15 and R16.







11.3. Keyboard, LCD

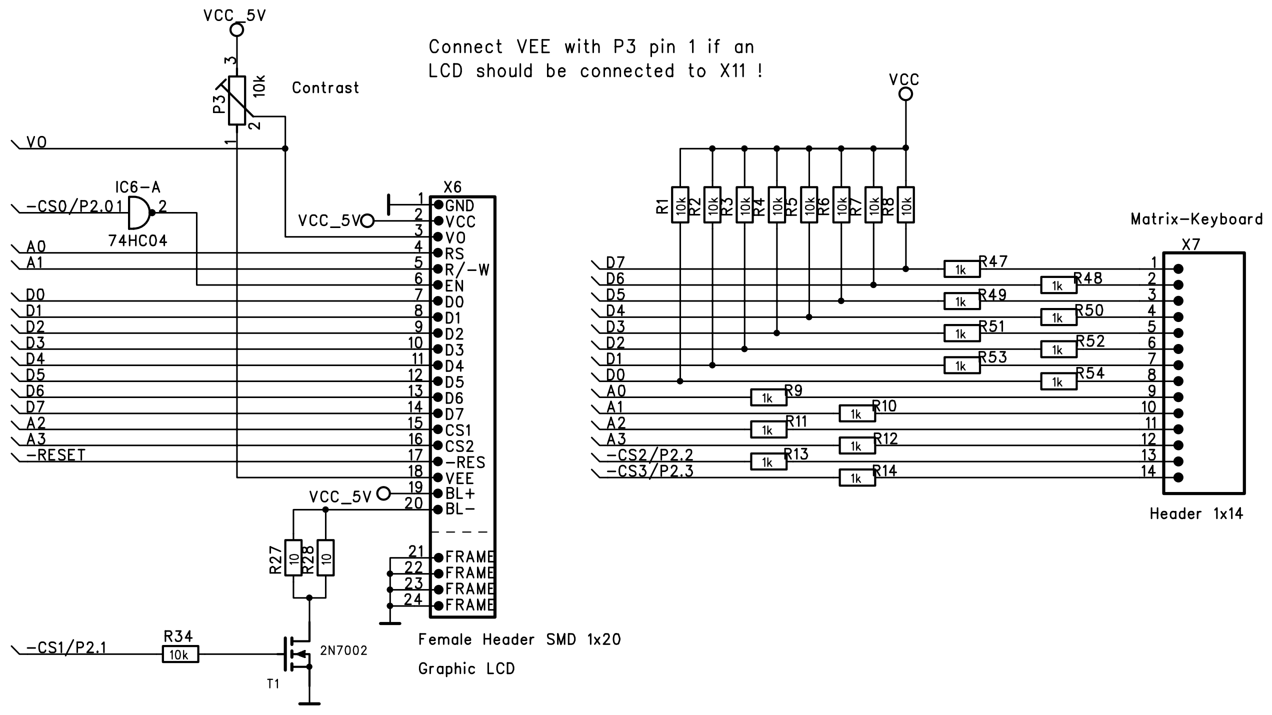
The X7 connector is intended for connecting LCDs with the Samsung KS0108 controller (also newly designated S6B0108), or the Hitachi HD61202. Since this controller's bus is not directly compatible to the PIF bus, it is adapted by means of an inverter and by use of additional, separate I/O addresses for reading and writing. An LCD then occupies I/O addresses 304h to 30Fh.

The P3 potentiometer serves for adjusting the LCD contrast.

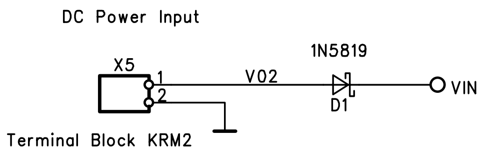
Scanning of a 4 x 8 matrix keyboard can be done simply by means of the PIF-bus signals D0..D7 and A0..A3. Since I/O addresses 100h..102h are used for this purpose, scanning the keyboard will not produce regular PIF-bus cycles (these would require that either -RD or -WR become active). The keyboard port accordingly also occupies no PIF bus I/O addresses. Series resistors are installed in the keyboard connections, so that pressing a key will not disturb the PIF bus.

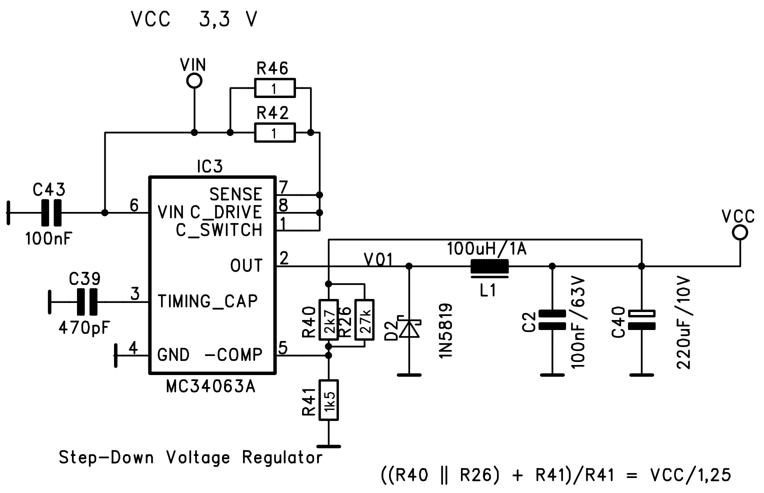
For 8 x 6 matrix keyboards, signals -CS2 and -CS3 can uses as additional ports for scanning.

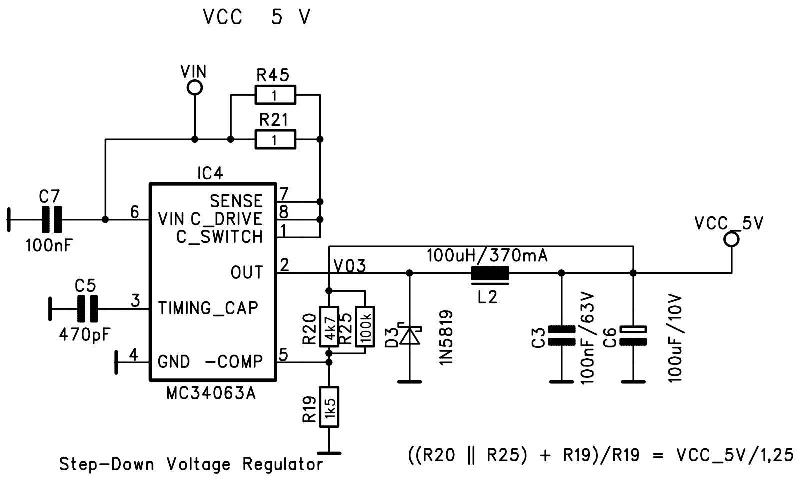
The keyboard scan is carried out within the interrupt routine of the system timer (Timer 0). Therefore a keyboard hardware interrupt is not necessarily needed. However, it *is* needed if the keyboard should be able to end the idle or power-down modes. A hardware interrupt must then be implemented, e.g by an eightfold NAND (74HC30) on D0..D7. In order to avoid that each PIF bus cycle triggers an interrupt, this IRQ must be masked out and only be unmasked when idle mode or power-down mode is activated.

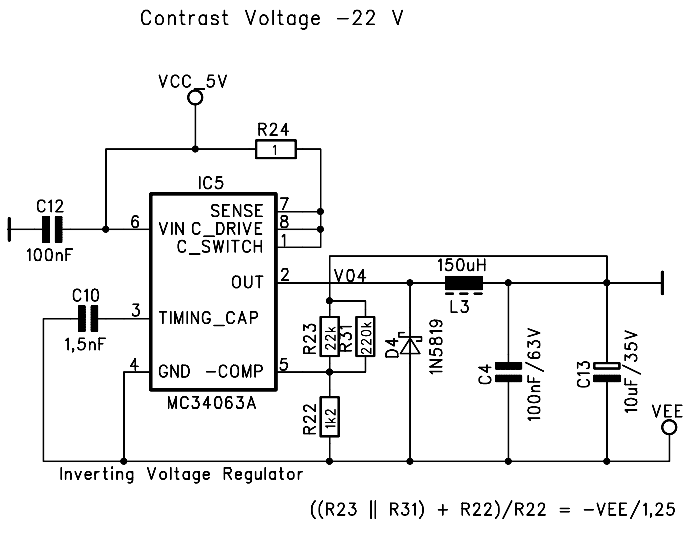

VCC_5V
P3 10k Contrast
Connect VEE with P3 pin 1 if an LCD should be connected to X11 !
VCC
VO
IC6-A 74HC04
-CS0/P2.0
A0 A1 D0 D1 D2 D3 D4 D5 D6 D7 A2 A3 -RESET
X6
GND VCC VO RS R/-W EN D0 D1 D2 D3 D4 D5 D6 D7 CS1 CS2 -RES VEE BL+ BL-
FRAME FRAME FRAME FRAME
R27 10 R28 10
-CS1/P2.1 R34 10k
2N7002 T1
Female Header SMD 1x20
Graphic LCD
R1 R2 R3 R4 R5 R6 R7 R8 10k
R47 R48 R49 R50 R51 R52 R53 R54 1k
R9 R10 R11 R12 R13 R14 1k
-CS2/P2.2 -CS3/P2.3
Matrix-Keyboard X7
Header 1x14

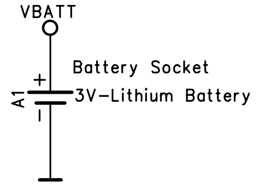
11.4. Power Supply







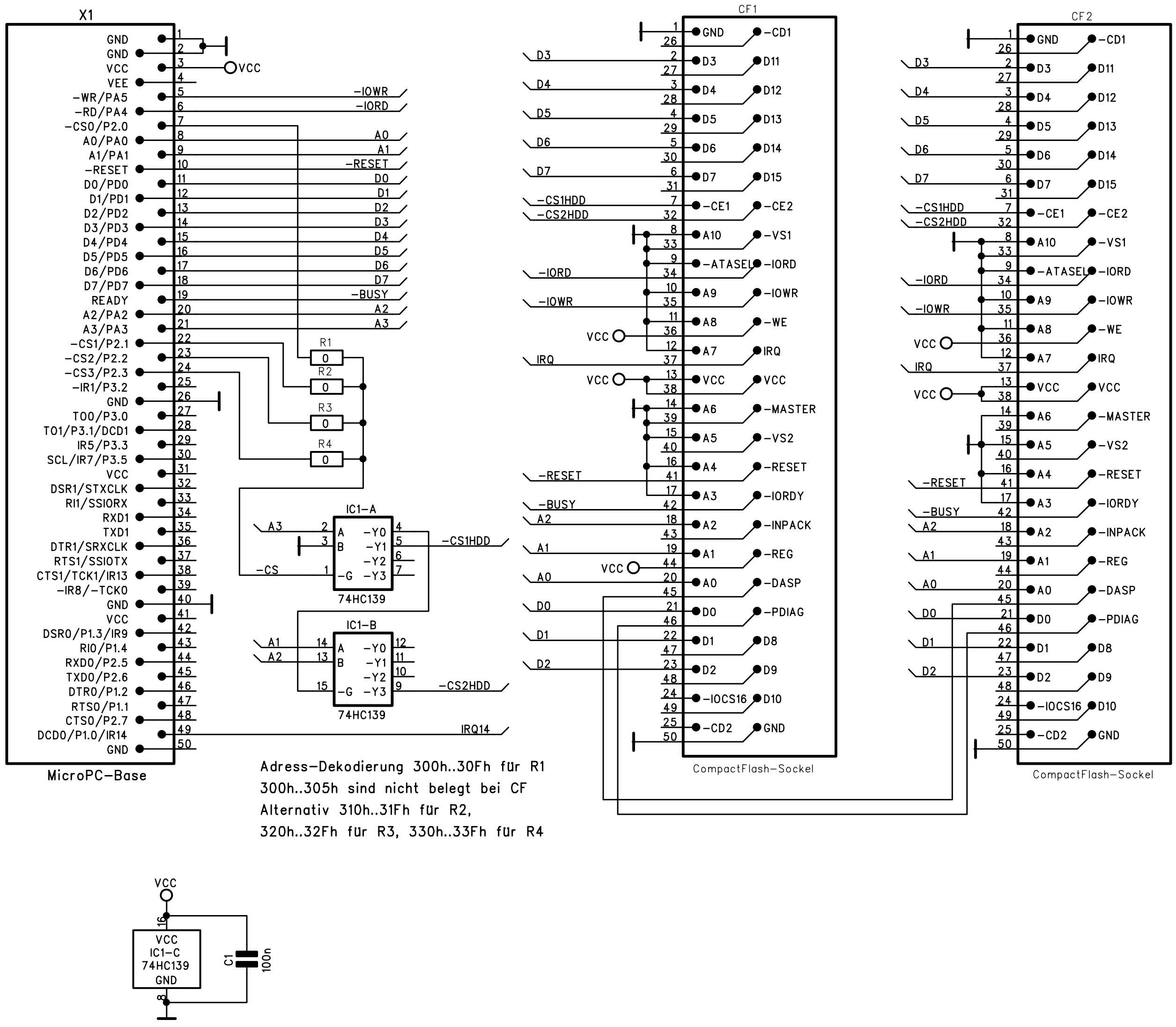




12. MicroPC CompactFlash Adapter Schematics

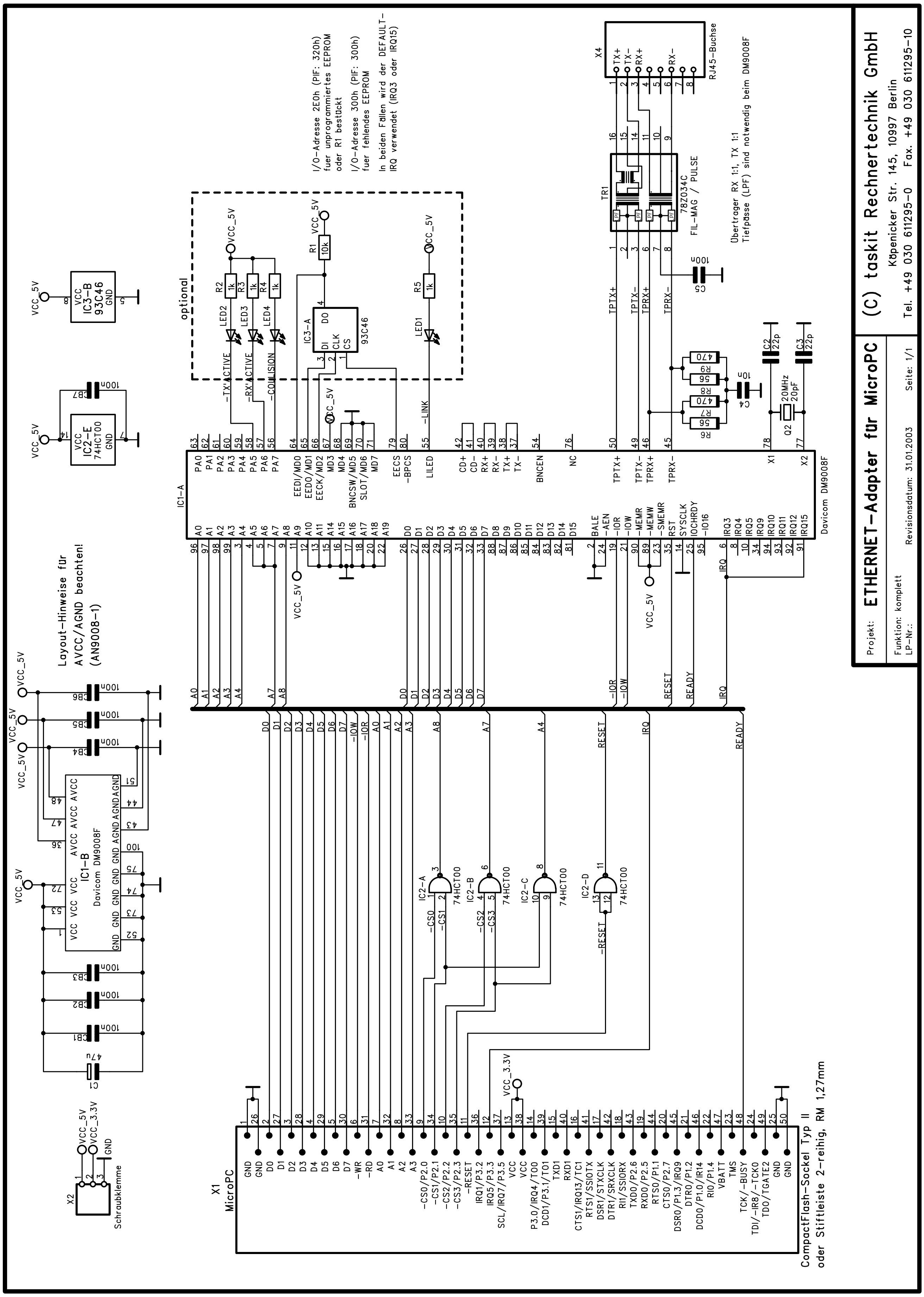
In accordance with the following connection diagram, an adapter for two CompactFlash cards can be configured. A 74HC139 or 74LCX139 IC is used here for address decoding. Thus the CompactFlash cards use only 10 I/O-addresses compared to 20 I/O-addresses on a completely passive adapter.

Only one of the four chip selects -CS0..-CS3 is needed, and accordingly only one of the bridges R1..R4 is installed. The BIOS functions for CompactFlash normally work with -CS0, using the address range 306h..30Fh.



Adress-Dekodierung 300h..30Fh für R1
300h..305h sind nicht belegt bei CF
Alternativ 310h..31Fh für R2,
320h..32Fh für R3, 330h..33Fh für R4

13. MicroPC Ethernet-Adapter Schematics


(C) taskit Rechnertechnik GmbH
Köpenicker Str. 145, 10997 Berlin
Tel. +49 030 611295-0 Fax. +49 030 611295-10
Projekt: ETHERNET-Adapter für MicroPC
Funktion: komplett
LP-Nr.:
Revisionsdatum: 31.01.2003
Seite: 1/1
I/O-Adresse 2E0h (PIF: 320h) fuer unprogrammiertes EEPROM oder R1 bestückt
I/O-Adresse 300h (PIF: 300h) fuer fehlendes EEPROM
In beiden Fällen wird der DEFAULT-IRQ verwendet (IRQ3 oder IRQ15)
Übertrager RX 1:1, TX 1:1
Tiefpässe (LPF) sind notwendig beim DM9008F
Layout-Hinweise für AVCC/AGND beachten! (AN9008-1)
optional
X4 RJ45-Buchse
TR1 78Z034C FIL-MAG / PULSE
IC1-A Davicom DM9008F
IC1-B Davicom DM9008F
IC2-A IC2-B IC2-C IC2-D 74HCT00
IC3-A 93C46
Q2 20MHz 20pF
X1 MicroPC
CompactFlash-Sockel Typ II oder Stiftleiste 2-reihig, RM 1,27mm
X2 Schraubklemme