Help contents

Help index

A

About the program

B

C

Chipset details
Command line parameters
Create Report

D

Device capabilities
Device details

E

Expression of thanks

F

FAQ – Frequently Asked Questions

G

H

Help contents

I

Install / Uninstall
IRQ Routing Table

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z

About the program

On the tab *About* the business logo of the program developer *Devid Espenschied* and the *registration details* are shown. The registration details are presented in the professional version of *ChipInfo* only if it is acquired with the matching keyfile. It is prohibited to pass on files of the professional version to third parties !

Under the logo the *internet* and *email* address of the program developer are displayed. With one click of your mouse you can either enter our internet site or send us an email message.

The registration details are:

- name (client name)
- address (street, house number, postal code and city)
- version type (D*emo* or P*rofessional*)
- registration number (unique customer number which is individually assigned to each customer).

In the demo version all customer data are replaced by *no answer* (n/a), with exception of the version type field.

See also:
Help contents
Help index

Program introduction

ChipInfo is a Windows based program which returns extensive and detailed data on the used mainboard chipset, the PCI bus and all connected devices. To achieve this the PCI bus is searched for all installed devices and data are returned in several conveniently arranged windows and lists.

The chipset as such has been introduced with the 286 processor and took over the functions of dozens of chips on older mainboards. Neat-Chipset from Chips & Technologies was the first to offer data on all important control functions of some chipsets. The control functions of documented registers were programmable. After that other manufacturers very soon changed the chipset market with Opti, Symphony, SiS and Via. With the introduction of the PCI bus Intel introduced the distinction between the so-called North and South Bridge. These bridges monitored different control instructions. The North Bridge mainly controlled the processor, while the South Bridge contained a set of different identifiable control units that every now and then partly work on the PCI bus as stand alone devices. Control units for AGP, IDE, SATA, SMBus, USB (1.1 and 2.0) and AC97 for audio and modem belong to that group. From the 8xx chipsets on Intel doesn't use the denotations North Bridge and South Bridge anymore; they are now called Memory Controller Hub (MCH) and I/O Controller Hub (ICH).

ChipInfo only shows chipset specifications based on the PCI register area. Other sources like the I/O registers or the so-called Memory Mapped Registers are not implemented yet. Because some manufacturers often consider the specifications of their chipsets confidential, some of those specifications are not available to us and therefore the program does not support those chipsets. For an exact overview of the program features, please see our feature list and the supported chipsets are here listed. *ChipInfo* runs under all Windows 9x/ME and Windows NT/200/XP/2003 operating systems.

ChipInfo has been developed for a multi lingual environment; therefore the chosen language has effect on all windows and lists. Chipset details are always given in English, because translation in German for example would not be meaningful. Further more the use of the terms register is avoided completely by using the real register names.

ChipInfo returns very extensive data and is therefore of interest to everyone who wants to know exactly which chipset and PCI devices are installed in his system and with which specifications. For the ambitious user the returned internal data can provide new insights regarding the used computer.

See also:
Help contents
Help index

Program features

In the next list an overview is given of the program features of the *ChipInfo* version which comes with this help file. Here you can get further information on the differences between the demo and professional version.

- Determination of detailed chipset information and matching chipset devices (list of supported chipsets)
- Determination of all installed PCI and AGP devices with details on every device, device-specific headers and used registers
- Opening up of device capabilities with Power Management, AGP, slot identification, Message Signaled Interrupts (MSI), PCI-X and Debug Port details
- Display of the IRQ routing table with unlocking of the PCI slots and matching interrupts
- Internal system data with summary of all processor data (processor, system, operating system, physical and virtual memory and program databases)
- Interactive or command line driven report generation
- Three different command line driven reports (standard with all details, only chipset details or only system internal)
- German and English language available in one package (operating system language is used by *ChipInfo* to establish the current language of the program, manual setting of the language is possible by using command lines)
- Debug mode with storage of debug data in the *debug.txt* file
- Several command line parameters for batch mode, language choice, driver management and debug mode
- Online help and manual in PDF format.

See also:
Help contents
Help index

System requirements

ChipInfo's system requirements are quite modest. The mandatory requirements are:

- x86-compatible mainboard with PCI bus
- x86-compatible processor with CPUID support
- Windows operating system with version 95, 98, ME, NT4, 2000, XP or 2003
- Administrator rights under Windows NT4, 2000, XP and 2003
- Hard disk with about 1.9 Mb of free space for program files and documentation
- Monitor with a minimum resolution of 800x600 to display all data on one screen.

See also:
Help contents
Help index

Install / Uninstall

Install

Installation by using a setup program is not necessary, because *ChipInfo* can be launched by clicking the *CHIPINFO.EXE* file. No entry is made in the start menu of the system.

The enclosed file *CHIPINFO.KEY* is mandatory for the professional version and contains customer data. To recognize the installed PCI devices, *ChipInfo* uses the file *PCI.DAT* and to examine the mainboard it also needs the *MAINBRD.DAT* database. Any other file is optional and only used for documentation or online help purposes.

Under the Windows NT4, 2000, XP and 2003 operating systems an additional possibility exists to manually install the drivers *ChipInfo* needs to access hardware (the driver file *HWACCESS.SYS* is part of *CHIPINFO.EXE*). In principle this will be taken care of during program startup. The */INSTALL* command line parameter performs this process without starting up *ChipInfo*. Performing this installation process will only be necessary when *ChipInfo* encounters startup problems during installation of these drivers.

Uninstall

Because no setup program has been used, *ChipInfo* can easily be removed by deleting all its files.

The hardware driver *HWACCESS.SYS*, which is installed with the first program startup, can be removed manually by using the */UNINSTALL* command line parameter. In principal however this driver will be uninstalled and deleted automatically by *ChipInfo* when shutting down the program.

See also:
Help contents
Help index

Program files

In the next list the files delivered with each version of *ChipInfo* are described with exception of the keyfile *CHIPINFO.KEY* that comes with the professional version only.

The concept of *ChipInfo* basically requires the present of the *CHIPINFO.EXE* and *CHIPINFO.KEY* files only. The first file is the program itself and the second one is the keyfile with customer data. Optionally the databases in the files *PCI.DAT* and *MAINBRD.DAT* are useful because they contain device names used by the program. The other files are only used for documentary and help purposes.

File	Needed for startup	Description
CHIPINFO.EXE	yes	program file
CHIPINFO.KEY	yes	keyfile with customer data
PCI.DAT	no	database with PCI device names
MAINBRD.DAT	no	database with mainboard specifications
CHIPINFO_D.HLP	no	german help file
CHIPINFO_E.HLP	no	english help file
CHIPINFO_D.PDF	no	german manual
CHIPINFO_E.PDF	no	english manual
HISTORIE.TXT	no	german history file
HISTORY.TXT	no	english history file
LIZENZ.PDF	no	german license agreement
LICENSE.PDF	no	english license agreement
ORDER.TXT	no	german/english order details

See also:
Help contents
Help index

Command line parameters

ChipInfo can be started up with several command line parameters. These commands can be used to either install or uninstall the *ChipInfo* driver or to set the desired language.
Furthermore a report generator can be activated from the command line directly which suppresses the startup of *ChipInfo's* graphical interface. This functionality is ideally applicable in network environments. The command line parameters are:

/H or /?

displays a window with possible command line parameters

/LANG=Language

manually sets the desired language (DEU = German, ENG = English)

/INSTALL

installs the *ChipInfo* driver manually (only for Windows NT4/2000/XP/2003, see Install/Uninstall).

/UNINSTALL

uninstalls the *ChipInfo* driver manually (only for Windows NT4/2000/XP/2003, see Install/Uninstall).

/DEBUG

activates the debug mode and creates the file *Debug.txt* in the current folder. This file can be used by the program developer to determine further actions.

/REPORT=file.txt

activates a report generator that creates a report file *file.txt* with all options. When the equality sign or the file name is omitted, *ChipInfo* automatically creates a report file named *chipinfo.txt*.

/REPORTCHIP=file.txt

activates a report generator that creates a report file *file.txt* with chipset data only. When the equality sign or the file name is omitted, *ChipInfo* automatically creates a report file named *chipinfo.txt*.

/REPORTSYS=file.txt

activates a report generator that creates a report file *file.txt* with system internal data only. When the equality sign or the file name is omitted, *ChipInfo* automatically creates a report file named *chipinfo.txt*.

See also:
Help contents
Help index

Expression of thanks

We wish to thank all beta testers and users whose suggestions, criticisms and error reports have helped us to improve the program.

Especially we wish to thank Dolf Westerveld for translating the documentation into English.

See also:
Help contents
Help index

Chipset details

ChipInfo automatically detects which chipset is operational on the mainboard and which devices are connected. In the first place the extracted chipset details in this area contain a header area with chipset names and stepping/revision details.

Next the proper details are presented in groups of the extracted chipset registers. The register *MCH Configuration* for instance, often found in Intel systems, is displayed in one separate line, followed by the extracted data on this register. A blank line is inserted between different registers.

With the exception of the header area all chipset details, like register specifications, options and results, are presented in English. Translation of these English oriented concepts in German for instance would be meaningless. Further more the use of the terms register is avoided completely by using the real register names.

If data is available about a non chipset PCI device, *ChipInfo* renames this category to *device details*. *ChipInfo* nevertheless tries to uncover chipset details based on vendor and device-ID data, when the device database *PCI.DAT* does not exist in the *ChipInfo* folder. In this case presentation of the chipset name, which is located in the database, is not possible.

ChipInfo displays details of the following chipsets (this list can vary in different versions of the program):

Intel:

Intel 434LX
Intel 434NX

Intel 430VX
Intel 430FX
Intel 430TX
Intel 430HX
Intel 430MX

Intel 440FX

Intel 440LX/440EX
Intel 440LX/440EX- Virtual PCI-to-PCI Bridge

Intel 440GX
Intel 440GX- PCI-to-PCI-Bridge

Intel 440BX/440ZX
Intel 440BX/440ZX- PCI-to-PCI-Bridge

Intel 450KX/GX- PCI Bridge
Intel 450KX/GX- Memory Controller

Intel 450NX- Memory & I/O Controller
Intel 450NX- PCI Expander Bridge

Intel 810(E/E2)
Intel 810(E/E2)-Graphics Device

Intel 815(E/P/EP/EM/G/EG)
Intel 815- AGP Bridge
Intel 815- Graphics Device

Intel 820
Intel 820- AGP Bridge

Intel 830M
Intel 830M- Graphics Device
Intel 830M- AGP Bridge

Intel 840
Intel 840- Hub Interface B Bridge
Intel 840- AGP Bridge

Intel 845(E/G/GL/GV/GE/PE/MP/MZ)
Intel 845- Host-to-AGP Bridge
Intel 845- Integrated Graphics Device

Intel 848(P/PE)
Intel 848- PCI-to-AGP Bridge
Intel 848- PCI-to-CSA Bridge
Intel 848- Overflow

Intel 850
Intel 850- AGP Bridge

Intel 852
Intel 852- GMCH Host-Hub Interface Bridge
Intel 852- GMCH Main Memory Control
Intel 852- GMCH Configuration Process
Intel 852- GMCH Integrated Graphics Device

Intel 855GM/GME
Intel 855GM/GME- GMCH Host-Hub Interface Bridge
Intel 855GM/GME- GMCH Main Memory Control
Intel 855GM/GME- GMCH Configuration Process
Intel 855GM/GME- GMCH Integrated Graphics Device

Intel 855PM
Intel 855PM- Host-Hub Interface Bridge
Intel 855PM- AGP Bridge
Intel 855PM- Power Management

Intel 860
Intel 860- Host-Hub Interface_A Bridge
Intel 860- Hub Interface_B Bridge
Intel 860- Hub Interface_C Bridge
Intel 860- AGP Bridge

Intel 865(P/PE/G/GV)
Intel 865- PCI-to-AGP Bridge
Intel 865- PCI-to-CSA Bridge
Intel 865- Overflow
Intel 865- Integrated Graphics

Intel 875P
Intel 875P- PCI-to-AGP Bridge
Intel 875P- PCI-to-CSA Bridge
Intel 875P- Overflow

Intel E7205
Intel E7205- Chipset Host RAS Controller
Intel E7205- PCI-to-AGP Bridge

Intel E7500
Intel E7500- DRAM Controller Error Reporting
Intel E7500- HI_B Virtual PCI-to-PCI Bridge

Intel E7501
Intel E7501- Host RASUM Controller
Intel E7501- HI_B PCI-to-PCI Bridge
Intel E7501- HI_B PCI-to-PCI Bridge Error Reporting
Intel E7501- HI_C PCI-to-PCI Bridge
Intel E7501- HI_C PCI-to-PCI Bridge Error Reporting
Intel E7501- HI_D PCI-to-PCI Bridge
Intel E7501- HI_D PCI-to-PCI Bridge Error Reporting

Intel E7505

Intel E7505- Chipset Host RAS Controller
Intel E7505- PCI-to-AGP Bridge
Intel E7505- HI_B PCI-to-PCI Bridge
Intel E7505- HI_B PCI-to-PCI Bridge Error Reporting

Intel MISA
Intel MPCI2

Intel MPIIX

Intel PIIX/PIIX3
- PCI to ISA Bridge
- IDE Controller
- USB Controller

Intel PIIX4
- PCI to ISA Bridge
- IDE Controller
- USB Controller
- Power Management

Intel ICH(0)
- Hub Interface to PCI Bridge
- LPC Interface Bridge
- IDE Controller
- USB 1.1 Controller
- SMBus Controller
- AC'97 Audio Controller
- AC'97 Modem Controller

Intel ICH2(M)
- LAN Controller
- Hub Interface to PCI Bridge
- LPC Interface Bridge
- IDE Controller
- USB 1.1 Controller
- SMBus Controller
- AC'97 Audio Controller
- AC'97 Modem Controller

Intel ICH3-M/S
- LAN Controller
- Hub Interface to PCI Bridge
- LPC Interface Bridge
- IDE Controller
- USB 1.1 Controller

- SMBus Controller
- AC'97 Audio Controller
- AC'97 Modem Controller

Intel ICH4
- LAN Controller
- Hub Interface to PCI Bridge
- LPC Interface Bridge
- IDE Controller
- USB 1.1 Controller
- USB 2.0 Controller
- SMBus Controller
- AC'97 Audio Controller
- AC'97 Modem Controller

Intel ICH5
- LAN Controller
- Hub Interface to PCI Bridge
- LPC Interface Bridge
- IDE Controller
- SATA Controller
- USB 1.1 Controller
- USB 2.0 Controller
- SMBus Controller
- AC'97 Audio Controller
- AC'97 Modem Controller

Intel PCI 64 Hub
- Hub Interface to PCI Bridge
- Advanced Interrupt Controller (APIC)

Intel PCI/PCI-X 64-Bit Hub 2
- Hub Interface to PCI Bridge
- I/OxAPIC Interrupt Controller
- Hot Plug Controller

AMD:

AMD 751
AMD 751- AGP and PCI-to-PCI Bridge

AMD 756
AMD 756- PCI-ISA Bridge
AMD 756- EIDE Controller
AMD 756- Power Management
AMD 756- USB Controller

AMD 761
AMD 761- PCI-to-PCI Bridge

AMD 762
AMD 762- PCI-to-PCI Bridge

AMD 766
AMD 766- PCI-ISA Bridge
AMD 766- EIDE Controller
AMD 766- Power Management
AMD 766- USB Controller

Via:

Via PIPC (VT82C586A/B)
Via PIPC- PCI-to-ISA Bridge
Via PIPC- IDE Controller
Via PIPC- USB Controller
Via PIPC- Power Management

Via Apollo VP (VT82C580VP)
Via Apollo VPX (VT82C580VPX)
Via Apollo VP2 (VT82C595)

Via Apollo VP3 (VT82C597)
Via Apollo VP3- PCI-to-PCI-Bridge

Via ProSavage PM133 (VT8605)
Via ProSavage PM133- PCI-to-PCI-Bridge (VT8605)

Via Apollo MVP3 (VT82C598MVP)
Via Apollo MVP3- PCI-to-PCI-Bridge (VT82C598MVP)

Via Apollo Pro (VT82C691)
Via Apollo Pro- PCI-to-PCI-Bridge (VT82C691)

Via Apollo PLE133 (VT8601A)
Via Apollo PLE133- PCI-to-AGP-Bridge (VT8601A)

Via Apollo PM601 (VT8601)
Via Apollo PM601- PCI-to-AGP-Bridge (VT8601)

Via Apollo Pro266 (VT8633)
Via Apollo Pro266- PCI-to-PCI-Bridge (VT8633)

Via KT133 (VT8363)
Via KT133- PCI-to-PCI Bridge (VT8363)

Via KX133 (VT8371)
Via KX133- PCI-to-PCI Bridge (VT8371)

Via CLE266
Via CLE266- PCI-to-PCI Bridge
Via CLE266- Graphics Controller

SiS:

SiS 630
SiS 630- PCI-to-PCI Bridge
SiS 630- IDE Controller
SiS 630- GUI Accelerator
SiS 630- LPC Bridge
SiS 630- Ethernet Controller
SiS 630- Audio Accelerator

Please understand that a presentation of all individual chipset details is not possible within the context of this help file. The huge amount of data would make this help file too large to handle.

See also:
Help contents
Help index

Device details

As well as the mainboard *ChipInfo* recognizes the connected devices, which are mostly marked as South Bridge components or, by Intel, as ICH components and represent a mass of different devices. It often concerns devices like the PCI-to-ISA Bridge, integrated graphic cards, IDE-bus controllers, USB-bus controllers, SM-bus controllers and the AC97 interface for audio and modem. First of all the extracted device details presented in this area contain a header area with device names and stepping/revision details.

Then the proper details are displayed in groups of the extracted device registers. A blank line is inserted between different registers.

With the exception of the header area, all device details, like register specifications, options and results, are presented in English. Translation of these English oriented concepts in German for instance would be meaningless. Further more the use of the terms register is avoided completely by using the real register names.

If the device is a Host-to-PCI bridge and chipset details are available then *ChipInfo* renames the data to *chipset details* in this area. If the device database *PCI.DAT* does not exist in the *ChipInfo* folder the program will nevertheless uncovers device details based on vendor and device ID data. In this case presentation of the device name, which is located in the database, is not possible.

Please understand that a presentation of all individual device details is not possible within the context of this help file. The huge amount of data would make this help file too large to handle.

See also:
Help contents
Help index

Device capabilities

The *Device Capabilities List* is useful when details about the corresponding PCI device are not available. However important details about this kind of devices can be obtained on the basis of the defined structures in the PCI specifications. For known devices with extensive device details the *Device Capabilities List* often provides a useful addition of information.

In the case of chipsets the switch *Device Capabilities* is located in the group *chipset details* and with non chipsets in the group *device details*. When this function is not supported by the corresponding device the switch cannot be selected. This function is deactivated in the demo version of *ChipInfo*.

ChipInfo displays details of the following device capabilities:

- PCI Power Management Interface
- AGP - Accelerated Graphics Port
- VPD - Virtual Product Data
- Slot Identification
- Message Signaled Interrupts
- CompactPCI - Hot Swap
- PCI-X
- Reserved for AMD
- Vendor Specific
- Debug Port
- CompactPCI - Central Resource Control
- PCI Hot-Plug

Please regard that only few details can be provided for the areas *CompactPCI - Hot Swap*, *Reserved for AMD*, *CompactPCI - Central Resource Control* and *PCI Hot-Plug*, because they are either not known or the corresponding specifications could not be obtained. In the future we will try to extend these areas with data.

The fact that each device can support each device capability area in more than one way is considered in the layout of the list. The area group names are displayed in brackets which makes distinction between different groups easy.

Please understand that a presentation of all individual group details is not possible within the context of this help file. The huge amount of data would make this help file too large to handle.

See also:
Help contents

Help index

PCI details

For each PCI device *ChipInfo* delivers general details mainly used for device identification in the corresponding register area before the *PCI header* (00h to 0Fh). These details are grouped in the next four clusters:

Device identification:

Unambiguous identification of devices is mainly based on the *vendor id* and *device id*. Vendor identifications are supplied by PCISIG (www.pcisig.com), device identifications can be chosen freely by the manufacturer. These data often suffice to recognize the exact device denotation in the device database *PCI.DAT*. Additionally the sub identifications *SubVendor* and *SubDevice-ID* exist. Although both features are part of the PCI header they are equally important to the identification and are therefore displayed in this area.

In the next line *device revision*, *device number*, *device function*, *device bus*, *header type*, *multi-functional device* check (i.e. the PCI-to-ISA bridge) and *device type* are presented. The *device type* is subdivided into 3 groups: *base class*, *sub-class* and *interface*. Returned data on *base* and *sub-class* are displayed in brackets.

Device command:

On the basis of the *device command* some data on device capabilities can be obtained. Some control data relate to *wait cycle control*, *parity errors*, the function *VGA Palette Snoop*, *memory write and invalidate*, *special cycles*, whether the actual device is a *bus master* and whether *memory* or *I/O-areas* are accessed.

Device status:

The *device status* represents the actual state of the concerning PCI device. An example is whether the *Capabilities List* which *ChipInfo* uses in the *Device Capabilities* window is supported. Other status values whether a *parity error* appeared in the Bus-Master device or whether *system failures* were observed and recognized and in which *speed mode* a device operates are also returned.

Other:

In this area details are presented which do not match to other categories. The *CacheLine size* implemented from the master device with memory write permission is one of these. *Cache line size* at the same time also serves as starting point for determining with which read commands with various speeds memory is accessed.

Latency time is expressed in PCI Bus Clocks and defines the time span between two accesses with PCI bus master devices.

The so-called *Build-In Self Test* (BIST) offers a testing mechanism which is implemented in the device. While testing the device normally does not operate on the bus, but switches to a special test mode. *ChipInfo* shows whether this self test is supported and, if so, the test has been performed successfully.

Unlike the chipset details *ChipInfo* shows these data in the chosen language. If German is selected,data will be returned in the German language.

See also:
Help contents
Help index

PCI header details

While the PCI register area 00h to 0Fh covers the registered standard area and is primarily responsible for device identification, the 10h to 3Fh area serves the so-called *PCI header*.

Each PCI device is classified in one of three different header types. For each header type a different register area is reserved and each is mostly specialized in different types of devices. The following header types are available:

- Type 0: standard header for all current devices
- Type 1: header for PCI-to-PCI bridges
- Type 2: header for PCI-to-CardBus bridges

A bridge in this case is logically considered to be a connection between two chipset components. A PCI-to-ISA Bridge for instance is a connection between PCI- and ISA-bus. If a device cannot be categorized as a type 1 or type 2 header, it will automatically be assigned to header type 0.

All 3 headers are divided into the columns *description* and *result* and first of all contain the range *general information* with *header type*, followed with *base addresses*, *latencies*, *BUS data* and *interrupt assignments*. Above all the latter is interesting to determine the assigned PCI- and ISA interrupt.

Headers 0 and 2 contain data for the *subsystem identification*. Here it concerns the *SubVendo*r and *SubDevice-ID*, which permit more exact device recognition similar to the regular Vendor- and Device-ID. *ChipInfo* accesses the database *PCI.DAT* for device recognition, however it cannot always supply results due to the freely adjustable ID's by the manufacturer.

Different from the chipset details, the data determined here are translated according to the chosen program language. If the German language is selected, the indicated details also appear in German.

Please understand that a presentation of all individual header details is not possible within the context of this help file. The huge amount of data would make this help file too large to handle.

See also:
Help contents
Help index

PCI registers

Each PCI device has a 256 byte address area, out of which *ChipInfo* extracts the majority of the indicated information. The window *PCI Register* lists these registers completely and divides the data into the following columns:

Register

Numbers the device registers sequentially in hexadecimal, starting with 00h.

Decimal

The register content in decimal.

Hexadecimal

The register content in hexadecimal.

Binary (7-0)

The register content in binary. Bits 7 to 0 are displayed from left to right.

Description

Contains the register designation, which comes from the defined structures within the PCI specification up to register 3Fh, and is additionally device specific.

Due to the fact that some device registers are laid out as Word or LongWord registers (16 or 32 bits), these registers are divided into the byte format. Behind the description text is displayed, between brackets, in how many parts the register is subdivided (beginning with 1).

If for example the specification of the respective PCI device is available in PDF format, you yourself can examine which options are active and not active on the basis of the listed registers.

In the demo version of *ChipInfo* the registers are only listed up to 7Fh.

See also:
Help contents
Help index

System internal

In the register tab *System internal* all important computer data are summarized and clearly presented on one page. During program startup the basic data are determined, so that no additional CPU time is needed during program execution. The data are divided into the following ranges:

Processor:

Contains details about the installed processor including *Manufacturer*, *Type*, *Frequency*, *Front Side Bus (FSB)*, *Multiplyer*, *CPUID command* data and the used hardware caches *L1*, *L2* and *L3*. A majority of the data within this category is returned on the basis the CPUID-command, which is available from later 486-processors on and enables exact conclusions about the processor and its characteristics. Other results as for example the *Front Side Bus* or *multiplyer* are determined with help of the so-called *Machine-specific Registers (MSR)*.

System:

This range provides a survey on the system component *mainboard*. It shows the *mainboard type* which first of all is obtained from the *MAINBOARD.DAT* device database on the basis of the *BIOS-ID*. If the device database is not available or the BIOS-ID does not exist the *mainboard type* is retrieved from the *Desktop Management Interface (DMI)*. Finally *ChipInfo* tries to retrieve this data by means of investigating the *BaseBoard.Manufacturer* and *BaseBoard.Product* attributes. In the line *PCI-devices* a statistical overview is given of the PCI-bus devices. It contains *all connected devices*, *multi-function devices*, *PCI-to-PCI devices* and *PCI-to-CardBus* devices. With regard to the installed BIOS *ChipInfo* shows the *BIOS type* with *version* and in a next line the *BIOS-ID*. This identification exclusively exists with AMI- and Award-BIOS's and contains manufacturer-specific details - *ChipInfo* uses it additionally for the recognition of the mainboard.

Operating system:

Here *ChipInfo* summarizes the general operating system data, like *Operating System Name*, *Suite*, *Version*, *Service Pack* and *Build*. The latter is frequently used to determine the difference between various beta-versions of the operating system.

physical Memory:

Physical memory is located on the mainboard in one or more memory banks. Data returned in this range are *Total*, *Used* and *Free*. Two or more memory banks are represented in one overall memory size because Windows see them as one.

virtual Memory:

Virtual memory is an extension of physical memory to gain more memory space. On the basis the paging file Windows uses hard disks and adds them to the operating system memory space. Application programs do not notice whether they reside in the main memory or on hard disk, since both areas represent a coherent space in the sense of virtual memory.

Data returned in this range are *Total*, *Used* and *Free*.

Program databases:

For statistics and version control *ChipInfo* delivers some data from the device databases *PCI.DAT* and *MAINBRD.DAT* if they are available in the current folder. In the field after the database name *yes* means that this database is available and connected. In other fields the *Database version*, *date* and the *size* in bytes are displayed. If a database is not available and therefore cannot be accessed, *ChipInfo* will report this situation immediately at program startup.

See also:
Help contents
Help index

IRQ routing table

Mainboards with a PCI-bus contain physical PCI-slots in which PCI-cards can be put and virtual PCI-slots which connect chipset components. Each slot contains 4 interrupt pins which are indicated *INTA#*, *INTB#*, *INTC#* and *INTD#*.

Within a x86 computer system 16 regular interrupts (PIRQ, Programmable Interrupt Request) are available which are already predefined for usage of basic technical system components (i.e. interrupt 4 for COM1 or interrupt 12 for PS/2 mice). Free interrupts are divided by the BIOS during the Power-On Self Test (POST) in such a way that the available interrupt pins are assigned to free interrupts. This process can be performed manually in many BIOS versions at one's own whishes.

The *IRQ Routing Table* represents an area within the BIOS in which the interrupt assignments are stored. *ChipInfo* reads these stored values and displays them in the register tab *IRQ routing table*.

These data are arranged in the following ranges:

General information:

Contains general details symbolizing the header of the *IRQ Routing Table*. It contains the signature *$PIR* which indicates the beginning of the table and is located in the memory range F0000h to FFFFFh. After that the *version* and table *size* (in bytes) are displayed, followed by the IRQ's exclusively assigned to PCI devices, the so-called *Miniport data* and a *checksum*.

PCI Interrupt Router device:

Here the corresponding PCI device is shown that primarily manages the interrupt assignments, containing the *Bus number* (mostly Bus 00h), *Device* and *Function number* and *Device name*. The latter is extracted from the *PCI.DAT* database.

Compatible PCI Interrupt-Router:

In this area the *Vendor-* and *Device-ID* are shown as well as the corresponding *Compatible PCI Interrupt Router name*. This device is often part of the chipset (i.e. PCI-to-ISA Bridge) and is able to assign PCI based interrupt pins to regular interrupts. Compatible because more interrupt routers can exist and the compatible router uses the same assignment method and interrupt management as the *PCI Interrupt Routing Device*. The description is fetched from the *PCI.DAT* database.

Slot Entries:

In this core area the *PCI slots* are listed by number and used. Data are presented in the

following parts:

Nr:

Number of the corresponding PCI slot, starting with 1.

Bus/Device/Function:

Contains the *Bus number* of the PCI slot (mostly 00h), the *Device number* and *Function number*.

Slot:

Represents the *Slot number* where zero-value represents a virtual PCI slot. Values from 1 on represent the numbered real physical PCI slots.

Type:

The 2 slot types are *physical* (PCI-slot) and *virtual* (mainboard). Physical slots are used for connection of external PCI-cards (i.e. network-card) and virtual slots, also located on the PCI-bus, are used to connect chipset components (i.e. PCI-to-ISA Bridge).

IRQ:

Here the 4 *Interrupt pins* are presented (*INTA#*, *INTB#*, *INTC#* and *INTD#*).

Link value:

Represents a value for the connection of the corresponding *Interrupt pin* and the regular interrupt. If this field contains a value 00h the corresponding *Interrupt pin* is not connected to an interrupt. If the value is higher it means assignment to other *Interrupt pins*, not to regular interrupts.

Link Bitmap:

The *Link Bitmap* is a hexadecimal value representing the interrupt assignment of *Interrupt pins* in a coded form.

IRQ Bitmap:

The *IRQ-Matrix* contains those interrupts for which assignment to the corresponding *Interrupt pin* was possible. Further circumstances are ignored in this result, i.e. free interrupts, whether multi-assignment is possible or interrupt-sharing is implemented.

See also:

Help contents
Help index

Create Report

With the *Report function* the obtained data can be stored in a file or be sent to a printer. Under *Report destination* you can choose the destination of the report. By selecting *Choose file* or *Choose printer* you can enter either the *File name* and path or the *Printer name* and its driver specific setups.

Under *Chipset/PCI* and *Other* you can choose which analysis data *ChipInfo* has to deliver. Selecting *Chipset/PCI* will fundamentally return analysis results for each individual PCI device. More analysis data in the area *Other* are returned only once independent of the number of available PCI devices.

The option *All PCI devices* is important on many PCI devices, because it analyze all activated PCI devices and does the activated PCI analysis for each device. When this option is not activated only the presently opened PCI device are analyzed.

To conclude this procedure clicking the *Create report* button starts the report function. During report generation *ChipInfo* shows which analysis is progressing in a *Status window*. After the conclusion you can return to the program by confirming in the *Status window*.

You can use 3 additional start parameters as command line parameters to generate reports. These options can very well be applied in network environments because *ChipInfo* will then be launched without the graphical user interface (GUI).

See also:
Help contents
Help index

Program analyses overview

Analyses performed on each individual PCI device are:

Chipset details / Device details
Device capabilities
PCI details
PCI header details
PCI registers

Analyses performed once for a computer system are:

System internal
IRQ routing table

See also:
Help contents
Help index

FAQ - Frequently Asked Questions

Question: At startup under Windows 95, 98 or NT the error message "LOADER ERROR: The procedure entry point VarNot could not be located in the dynamic link library oleant32.dll " (or similar) is returned. What does that mean?

Answer: The used programming language Borland Delphi 6 apparently has been changed basically with Service Pack 2 (see Borland's internet site: http://bdn.borland.com/article/0,1410,28841,00.html). Therefore an update from DCOM has to be installed for some Windows 95/98/NT systems.

The following links can be used to get to Microsoft's web site.

Windows 95 DCOM Update: http://www.microsoft.com/com/dcom/dcom95/dcom1_3.asp
Windows 98 DCOM Update: http://www.microsoft.com/com/dcom/dcom98/dcom1_3.asp
Windows NT4 ServicePack 4: http://www.microsoft.com/ntserver/nts/downloads/recommended/NT4SvcPk4/NT4SvcPk4.asp

These updates are necessary to run our program on these systems.

Question: How can I get updates as fast as possible?

Answer: Registered customers and therefore users of the professional version have a user name and password at their disposal for entering the protected customer area of our website by means of the delivered keyfile. When a new version of *ChipInfo* is available an email is sent to the registered users with an overview of the improvements and new program functions. After logging in they can download the latest professional version. After unpacking the professional version can be started up together with the initial keyfile.

Question: How can I activate the debug mode?

Answer: The */DEBUG* command line entry activates the debug mode and creates the text file *debug.txt* in the current folder. This function makes sense when *ChipInfo* encounters problems during startup or when it cannot recognize some hardware components. If problems occur please send this file to the developers (chipinfo@pcanalyser.de) to make solving of the problem possible. Please consider that startup in the activated debug mode takes longer. This is caused by opening the debug file at every write operation and closing it afterwards to guarantee the file integrity.

See also:
Help contents

Help index

Professional version

The demo version provides about half of the functionality of the commercial professional version. Fundamentally the demo version should be used only for evaluation purposes where the limited functionality provides sufficient insights into the program.

The fully functional professional version provide extensive chipset data, the IRQ routing table and 3 different batch modes for automatic program startup.

Thus the professional version is interesting for users who need stocktaking of a network. On the other hand even more details are presented to the ambitious user who wants to examine further system sources.

By obtaining the professional version you support further development efforts to follow market changes in the chipset sector as close as possible.

All differences between the demo and the professional version are presented in the following list:

Program function	**Demo version**	**Professional version**
Complete chipset information	no	yes
Device capabilities (if supported)	no	yes
More than 128 showable PCI registers	no	yes
IRQ Routing Table	no	yes
Batch mode for automatic program start	no	yes
Demo windows	yes	no

See also:
Help contents
Help index