



SPECTRUM
SYSTEMENTWICKLUNG MICROELECTRONIC GMBH

PCI.DIO32
100 MHz digital I/O
for PCI bus

Hardware Manual
Driver Manual

© Spectrum Systementwicklung Microelectronic GmbH - 2004
Ahrensfelder Weg 13-17, 22927 Grosshansdorf, Germany

SBench is a registered trademark of Spectrum Systementwicklung Microelectronic GmbH.

MS-DOS, Windows and Windows NT are trademarks or registered trademarks of Microsoft Corporation.

LabVIEW is a trademark of National Instruments Corporation.

MATLAB is a registered trademark of The MathWorks Inc.

Agilent VEE is a trademark of Agilent.

FlexPro is a registered trademark of Weisang & Co.

DASYLab is a registered trademark of DATALOG GmbH.

Spectrum reserves the right to make improvements and/or changes to the products and/or programs at any time in order to improve design and to supply the best product possible.

Table of Contents

Table of Contents	3
Preface	5
Versions	5
Product Introduction	6
General Information.....	6
Software	6
Additional information	6
Ordering.....	6
Installation	7
System Requirements	7
Hardware Installation	7
Driver Installation	7
DOS	8
Win 98/ME/2000/XP (WDM).....	8
Windows 95 (VXD)	8
Windows NT (Legacy)	9
Windows NT.....	9
Installation for Linux.....	10
Login	10
Select the right driver	10
Load Driver	10
Major Number.....	10
Installing the Device	10
End	10
Info	11
Utilities	11
SBench 5.x.....	11
DRVCONFIG.EXE	11
PCITEST.EXE.....	11
Hardware Description	12
Trigger Information.....	12
Option Multiple Recording	12
Option Gated Sampling.....	13
Option Synchronisation	14
Block Diagram.....	15
Technical data.....	15
Placement	15
Connector	16
Sync Bus	16
Software Description	17
General Information	17
Header files on CD	17
DLLTYP.H.....	17
SPECTRUM.H	17
REGS.H	17
ERRORS.H.....	17
Driver functions.....	17
int16 SpcInitPCIBoards (int16* count, int16* PCIVersion).....	18
int16 SpcInitBoard (int16 nr, int16 typ).....	18
int16 SpcSetParam (int16 nr, int32 reg, int32 value)	18
int16 SpcGetParam (int16 nr, int32 reg, int32* value).....	18
int16 SpcSetData (int16 nr, int16 ch, int32 start, int32 len, dataptr data)	18
int16 SpcGetData (int16 nr, int16 ch, int32 start, int32 len, dataptr data)	19
Error Codes.....	19
Valid Board Types	19
Hints for programming the boards.....	19
Software - Register of PCI.DIO32	21
PCI register.....	21
PCI Features register	21
Error registers	21
Status register	21

Command register	22
Synchronisation (Option)	22
Memory register	22
Posttrigger register	22
UserxIn/Out register	22
Features register	23
Modes of operation (from version 2.32 on)	23
Triggermode register	24
Triggeredge register	24
Triggermask register	24
Triggerpattern register	25
Pulsewidth register	25
Examples for trigger definition	25
Multiple Recording (option)	25
Gated Sampling (option from version 2.32 on)	26
Samplerate register	26
Input direction register	27
Input impedance register	27
Data (Read/Write)	27
Example of Driver use	28

Vorwort

Diese Anleitung enthält detaillierte Informationen über die Hardware Möglichkeiten der PCI.DIO32 von Spectrum Systementwicklung. In den Informationen sind die technischen Daten, die Spezifikationen und die Beschreibung der Schnittstellen enthalten.

Außerdem führt diese Beschreibung durch den Installationsprozess sowohl der Karte als auch der Treiber für das jeweilige Betriebssystem.

Zuletzt enthält dieses Handbuch die komplette Software Beschreibung der Karte und des zugehörigen Treibers. Der Leser wird in die Lage versetzt diese Karte in einem beliebigen PC System unter einem der unterstützten Betriebssysteme einzusetzen.

Achtung, in diesem Handbuch ist keine Beschreibung der speziellen Treiber für die Produkte von Drittherstellern wie LabVIEW oder MatLab enthalten. Diese Treiber sind nicht im normalen Lieferumfang enthalten.

Neuerungen der Karte, zusätzliche Optionen oder Speicher-ausrüstungen werden auf der Homepage <http://www.spec.de> bekannt gegeben. Hier kann ebenfalls die neueste Treiberversion mit den letzten Fehlerbereinigungen gefunden werden.

Versions

Version	Date	Changes/New options/Enhancements
1.0		First series delivered to customers.
1.16		new features implemented: DirectIO, 16 Bit data transferring
2.16	Until January 2000	printed circuit board redesign (PCI bus interfacechip)
2.32	Current version	new trigger-modes implemented

© Spectrum Systementwicklung Microelectronic GmbH 2002
Ahrensfelder Weg 13-17, 22927 Grosshansdorf, Germany

SBench is a registered trademark of Spectrum Systementwicklung Microelectronic GmbH.

MS-DOS, Windows and Windows NT are trademarks or registered trademarks of Microsoft Corporation.

LabVIEW is a trademark of National Instruments Corporation.

MATLAB is a registered trademark of The MathWorks Inc.

HP VEE is a trademark of Hewlett Packard Company

FlexPro is a registered trademark of Weisang & Co

Spectrum reserves the right to make changes at any time in order to improve design and to supply the best product possible.

Preface

This manual provides detailed information on the hardware features of the PCI.DIO32 from Spectrum Systementwicklung. This information includes specifications, block diagram, connector description.

In addition, this guide takes you through the process of installing your board and also describes the installation of the delivered driver package for each operating system.

Finally this manual provides you with the complete software information of the board and the related driver. The reader of this manual is able to integrate the board in any PC system with one of the supported operating systems.

Please note that in this manual there is no description for specific driver parts like LabVIEW or MatLab software that are not normally enclosed in the hardware.

For any new information on the board as well as new available options or memory upgrades please contact our website <http://www.spec.de>. You will also find the actual driver package with the latest bug fixes on our site.

Product Introduction

Allgemeine Information

Die schnelle digitale I/O Karte PCI.DIO32 arbeitet mit zwei 16 Bit Kanälen. Jeder Kanal kann individuell als Eingang oder Ausgang programmiert werden. So stehen maximal 32 Bit zur Aufzeichnung oder zur Wiedergabe zur Verfügung. Für Systemtests eignet sich der Modus mit 16 Bit Ausgabe synchron zu 16 Bit Aufzeichnung.

Zur Synchronisierung mit externen Systemen steht ein Trigger und ein Takt Ein-/Ausgang zur Verfügung. Dies erlaubt die universelle Zusammenarbeit mit einer Vielzahl anderer auch herstellerfremder Karten.

Ein spezieller SYNC-Bus ermöglicht die Kaskadierung mehrerer PCI.DIO32 zu einem Mehrkanalsystem. Genauso ist damit die Synchronisierung der Karte mit Transientenrekordern aus dem Hause Spectrum möglich. Dies erlaubt die Erstellung von gemischten Analog/Digital Aufzeichnungs- und Ausgabesystemen.

Anwendungsbeispiele: Bauteiltest, Systemtest, Mustererkennung.

Software

Kostenlos mitgeliefert werden Treiber für Linux, DOS und Windows 9x/ME/NT/2000/XP. Für die einfache Programmierung sind Beispiele in C/C++, Delphi und Visual Basic enthalten. Darüber hinaus steht zur komfortablen Steuerung die Signalverarbeitungssoftware SBench 5.2 kostenlos zur Verfügung. Außerdem sind Treiber für LabVIEW, DASYLab, MATLAB und VEE erhältlich.

Additional information

The PCI.DIO32 operates with components having very high power consumption. Therefore it is highly recommended to place the board near the cooling fan. Do not use the PCI.DIO32 in hermetic closed systems.

Ordering

PCI.DIO32 standard	PCI.DIO32 with 8 MSamples (32 MBytes) memory including drivers & cable	PCIDIO32
Option 32M	memory upgrading to 32 MSamples (128 MBytes)	PCIDIO32-32
Option Update	later memory update after delivery of board	PCIDIO32-up
Cascading	Synchronisation of several PCI.DIO32 for multi-channel-systems	PCIDIO32-ks
Multiple Recording	Memory segmentation for fast repetition rates	PCIDIO32-mr
Gate	Gated sampling with an external control signal	PCIDIO32-gs
DASYLab driver	Drivers for DASYLab 5.0 for Win 95/98, Win 2000 and Win NT	PCIDIO32-dl
HP-VEE driver	Drivers for HP-VEE 5.0 for Win 95/98, Win 2000 and Win NT	PCIDIO32-hp
LabVIEW driver	Drivers for LabVIEW 4.0 for Win 3.11, Win 95/98, Win 2000 and Win NT	PCIDIO32-lv
MatLab driver	Drivers for MatLab 5.0 for Win 95/98, Win 2000 and Win NT	MATLAB
Extra cable	80 pol flat ribbon cable with connector	PCIDIO32-cab

General Information

The fast digital I/O board PCI.DIO32 works with two 16 bit channels. Each channel may be individually programmed as input or output. This results in a maximum of 32 bit for output or 32 bit for input. A system test mode with 16 bit input synchronously to 16 bit output is also implemented.

For synchronisation purposes with external systems the board is equipped with a clock and trigger in/output. This allows the board to work with other boards from Spectrum as well as boards from other producers.

An internal synchronisation bus allows it to cascade multiple PCI.DIO32 as well to synchronise the board with transient recorders from Spectrum. This results in a fast system with mixed digital/analogue input/output.

Application examples: part testing, system testing, pattern recognition

Software

Drivers for Linux, DOS and Windows 9x/ME/NT/2000/XP as well as programming examples for C/C++, Delphi and Visual Basic are delivered with the board. Comfortable programming, initialising and data display are performed by the free-of-charge Windows program SBench 5.2. Software drivers for LabVIEW, DASYLab, MATLAB and VEE are available.

Installation

System Anforderungen

PCI basierter IBM kompatibler PC mit mindestens einem freien PCI Steckplatz in voller Länge. Der PCI-Bus muß mindestens der Revision 2.1. genügen. Die Karte arbeitet nicht in einem PCI-Bus Revision 2.0 oder früher. Wenn mehr als eine Karte im System installiert werden soll, so empfehlen wir einen zusätzlichen Lüfter für die Karten einzusetzen.

Hardware Installation

- (1) Schalten Sie den PC aus.
- (2) Öffnen Sie das Gehäuse.
- (3) Wählen Sie einen freien PCI Steckplatz der benötigten Länge aus. Wenn in Ihrem System kein zusätzlicher Lüfter installiert ist, so ist die beste Wahl ein Steckplatz, in dem die Karte nicht direkt neben einer anderen Karte platziert ist. Wenn Ihr System einen oder mehrere zusätzliche Lüfter besitzt, so platzieren Sie die Karte direkt in deren Luftstrom.
- (4) Installieren Sie die Karte in dem ausgewählten Steckplatz. Achten Sie dabei besonders auf den korrekten Sitz des PCI Steckers im Steckplatz.
- (5) Schrauben Sie die Karte an der Frontblende am Gehäuse fest.
- (6) Wenn Sie eine PCI Karte in voller Baulänge erstanden haben, so liegt Ihrer Karte ein Kartenhalter bei (bei Karten mit SMB Steckern ist dieser bereits montiert). Es wird empfohlen diesen Kartenhalter zu installieren, um die Karte fest im System zu fixieren. Wenn Sie eine Karte mit 9 mm BNC Steckern haben, so ist nur die nachträgliche Montage des Bügels an der bereits installierten Karte mit einem kurzen Schraubendreher möglich.
- (7) Starten Sie das System.
- (8) Wenn Ihr System nicht bootet, überprüfen Sie bitte den korrekten Sitz der Karte in ihrem Steckplatz. Starten Sie danach das System neu.
- (9) Wenn Ihr System immer noch nicht bootet kann es jetzt ein Problem in der Zusammenarbeit mit anderen PCI Karten geben. Deinstallieren Sie bitte alle anderen PCI Karten bis auf die Grafikkarte und versuchen Sie das System in dieser Konfiguration zu starten. Wenn diese Maßnahme zum Erfolg führt, so muß vermutlich die Reihenfolge der PCI Karten in Ihrem System geändert werden.

Treiber Installation

Spectrum liefert einen Kartentreiber aus, der alle Karten unterstützt. Dieser Treiber hat auf allen Betriebssystemen ein einheitliches Interface. Mit Vorstellung der Treiberversion 3.00, die jetzt einen WDM kompatiblen Treiber enthält, mußte eine Unterteilung in PCI und ISA Karten gemacht werden. Bitte wählen Sie den passenden Treiber anhand der Tabelle aus. Wenn Sie ISA und PCI Karten von Spectrum gemischt in einem System benutzen, so nutzen Sie bitte den ISA Treiber.

System Requirements

PCI based IBM PC compatible PC with at least one free full-length PCI slot. The PCI bus version must be at least revision 2.1. The boards will not work with older PCI buses of revision 2.0. If you are installing more than one board in your PC, an additional cooling fan is strongly recommended.

Hardware Installation

- (1) Power off your PC.
- (2) Open the cover.
- (3) Select a free PCI slot of the required length. If you are using a system with no additional cooling fans, it is the best decision to put the board in a slot not adjacent to any other board. If you have a system with additional cooling fans, place the PCI board in front of a cooling fan.
- (4) Install the board in this slot. Make sure that the PCI connector is right struck into the slot.
- (5) Use a screw to fix the bracket to the PC.
- (6) If your board has full PCI length a retainer is delivered with the board (on boards with SMB connectors this retainer is already installed). It is recommended to use this retainer to fix the board in the system. If you have a board with 9 mm BNC connectors, it is not possible to install the retainer before inserting the board in the system. You need to install the retainer with a short screwdriver to the already installed board.
- (7) Reboot the system.
- (8) If your system will not boot, please check whether the board is struck correctly into the PCI connector and reboot again.
- (9) If your system will not boot after this, there may be a problem with other PCI boards. Please de-install all other PCI boards and try to boot the system without them. If this works, you may have to change the order of the PCI boards in the system.

Driver Installation

Spectrum supplies one driver that supports all boards with an unique interface for all operating systems. With introduction of the new version 3.00 which includes a WDM style driver there has been a separation made between PCI and ISA boards. Please use the matching driver that is listed in the table. If you mix ISA and PCI boards from Spectrum in your system you need to use the ISA driver.

Operating System	PCI boards only	PCI and ISA mixed	ISA boards only
Windows XP	WDM driver (Windows 98/ME/2000/XP)	Legacy driver (Windows NT)	Legacy driver (Windows NT)
Windows 2000	WDM driver (Windows 98/ME/2000/XP)	Legacy driver (Windows NT)	Legacy driver (Windows NT)
Windows ME	WDM driver (Windows 98/ME/2000/XP)	VXD driver (Windows 95)	VXD driver (Windows 95)
Windows 98	WDM driver (Windows 98/ME/2000/XP)	VXD driver (Windows 95)	VXD driver (Windows 95)
Windows NT	Legacy driver (Windows NT)	Legacy driver (Windows NT)	Legacy driver (Windows NT)
Windows 95	VXD driver (Windows 95)	VXD driver (Windows 95)	VXD driver (Windows 95)

DOS	OBJ files (DOS driver)	OBJ files (DOS driver)	OBJ files (DOS driver)
Linux	Linux Kernel Module	Linux Kernel Module	Linux Kernel Module

DOS

Der Treiber für DOS besteht aus einem Satz Objektdateien zum Linken in ein DOS Programm. Die Treiber Dateien können auf der CD im Verzeichnis \DRIVER\DOS auf der Diskette gefunden werden. Beispiele zur Nutzung von Borland C++ 3.1 sind ebenfalls vorhanden. Zur Benutzung der Treiber müssen nur die Objekt Dateien *.OBJ und die Header Dateien *.H ins Arbeitsverzeichnis kopiert werden.

Wenn die Beispielprogramme bei der Arbeit mit DOS nicht laufen, so kann es hier zu einem Problem mit den im System installierten Software Treibern gekommen sein. Starten Sie das System erneut ohne irgendwelche installierten Treiber. Wenn das Programm so läuft, fügen Sie Schritt für Schritt Ihre Treiber wieder in das System ein, um den problematischen Treiber herauszufinden.

Auf einigen Motherboards kann es zu Problemen mit älteren Versionen der Datei EMM386.EXE kommen. Die Version 6.22 läuft hier korrekt. Es kann daher nötig sein, diese Datei gegen eine neuere Version auszutauschen.

Win 98/ME/2000/XP (WDM)

Wenn das Betriebssystem Windows 98, Windows ME, Windows 2000 oder Windows XP installiert ist, wird die PCI Karte nach dem nächsten Start automatisch erkannt. Das System bietet die direkte Installation eines Treibers für die Karte an. Wählen Sie hier als Installationsquelle die mitgelieferte CD. Die Treiberdateien befinden sich im Verzeichnis \Driver\Win98_2k_XP. Die Treiber stehen sofort nach der Installation ohne Neustart des Systems zur Verfügung.

Die Treiber bestehen aus einer 32 Bit DLL, die alle Funktionen des Treibers enthält, und einem WDM-Kernel-Treiber (SYS). Die DLL kann mit allen Systemen benutzt werden, die eine Schnittstelle zu 32 Bit Windows DLLs anbieten. Beispiele für Microsoft Visual C++, Borland Delphi und Microsoft Visual Basic sind ebenfalls enthalten.

Falls Sie Visual C++ benutzen, so ist es möglich, die Library Datei SPECTRUM.LIB mit in ein Projekt zu integrieren, um die Funktionen des Treibers auf einfache Weise in das Programm einzubinden. Die Library Datei arbeitet nicht mit Borland Compilern zusammen.

Die beiden DLL's unterscheiden sich nur im Aufruf der Funktionen. Die Datei SPECTRUM.DLL exportiert die Funktionen als _cdecl (für C, C++, Delphi), die Datei SPCSTD95.DLL als _stdcall (für Visual Basic). Je nach benutztem Compiler kann eine der beiden DLL's benutzt werden.

Windows 95 (VXD)

Die Treiber für Windows 95 bestehen aus einer 32 Bit DLL, die alle Funktionen des Treibers enthält, und einem Virtual Device Driver (VXD). Die DLL kann mit allen Systemen benutzt werden, die eine Schnittstelle zu 32 Bit Windows DLLs anbieten. Beispiele für Microsoft Visual C++, für Borland Delphi und für Visual Basic sind ebenfalls enthalten.

Zur Installation des Treibers benutzen Sie bitte die auf der CD enthaltene Installationsversion im Verzeichnis /Install/Win95Drv. Hiermit werden alle Treiberdateien in die vorgesehenen Verzeichnisse installiert.

Falls Sie Visual C++ benutzen, so ist es möglich, die Library Datei SPECTRUM.LIB mit in ein Projekt zu integrieren, um die Funktionen des Treibers auf einfache Weise in das Programm

DOS

The driver consists of a set of object files ready to link to a DOS program. The driver files are found on CD in the directory \DRIVER\DOS on the driver disk. Examples for the use with Borland C++ 3.1 are included. To use the driver files, just copy the object *.OBJ and header *.H files to your working directory. If the example files are not working when using DOS operating system, there may be problems with the installed software drivers. Start the system once again without any software drivers installed. After this install the drivers step by step to find out the problematic software driver. On some motherboards, there may be problems when using older versions of EMM386.EXE. The version 6.22 works correctly. It may be necessary to update this driver to a higher version.

Win 98/ME/2000/XP (WDM)

When the operating system Windows 98, Windows ME, Windows 2000 or Windows XP is installed, the PCI board will be automatically recognised after the rebooting. The system will ask for a driver to be installed. Select the install directory from the Spectrum driver CD. The driver files are placed in the directory \Driver\Win98_2k_XP. The driver is ready to use directly after installing, no reboot is necessary.

The driver consists of a 32 bit windows DLL which includes all functions of the driver and a WDM kernel driver (SYS). The DLL can be used with all systems which accept 32 bit windows DLL's. Examples for Microsoft Visual C++ 4.x, Borland Delphi and Microsoft Visual Basic are included.

If you are using Microsoft Visual C++, you may use the delivered library file SPECTRUM.LIB to access the driver functions easily. The library file will not work with Borland compilers.

The only difference between the both DLL's is the calling convention. The file SPECTRUM.DLL uses _cdecl definition (for C, C++, Delphi), the file SPCSTD95.DLL uses _stdcall definition (for Visual Basic). Depending on the used programming language, one of the two DLL's may be used.

Windows 95 (VXD)

The driver consists of a 32 bit windows DLL which includes all functions of the driver and a virtual device driver (VXD). The DLL can be used with all systems which accept 32 bit windows DLL's. Examples for Microsoft Visual C++ 4.x, Borland Delphi and Microsoft Visual Basic are included.

You need to use the install program for driver installation. The program is located on CD in the directory /Install/Win95Drv. The program installs all driver files in the correct directory.

If you are using Microsoft Visual C++, you may use the delivered library file SPECTRUM.LIB to access the driver functions easily. The library file will not work with Borland compilers.

The only difference between the both DLL's is the calling convention. The file SPECTRUM.DLL uses _cdecl definition (for C,

einzubinden. Die Library Datei arbeitet nicht mit Borland Compilern zusammen.

Die beiden DLL's unterscheiden sich nur im Aufruf der Funktionen. Die Datei SPECTRUM.DLL exportiert die Funktionen als `_cdecl` (für C, C++, Delphi), die Datei SPCSTD95.DLL als `_stdcall` (für Visual Basic). Je nach benutztem Compiler kann eine der beiden DLL's benutzt werden.

Windows NT (Legacy)

Der Treiber besteht aus einem Kernel Mode Treiber für Windows NT und einer 32 Bit DLL, die die Funktionen des Kernel Mode Treibers benutzt. Beispiele für Microsoft Visual C++, Borland Delphi und Microsoft Visual C++ werden ebenfalls mitgeliefert.

Windows NT

- (1) Loggen Sie sich als ADMINISTRATOR oder als ein Benutzer mit dem Recht Treiber zu installieren und die Registry zu ändern in Ihr System ein.
- (2) Starten Sie das Setup Programm auf der Treiber CD. Sie finden das Installationsprogramm im Verzeichnis `\Install\WinNTDrv`.
- (3) Das Installationsprogramm installiert den Kernel Mode Treiber und die 32 Bit Windows DLL, sowie einige Hilfsprogramme im Verzeichnis 'Spectrum GmbH'. Die Registry wird ebenfalls angepaßt.
- (4) Starten Sie den Computer neu.
- (5) Die PCI Karten werden automatisch vom Kernel erkannt und eingetragen.
- (6) Falls der Geräte Treiber nicht korrekt startet (Eine Nachricht im Event Log von der Datei SPCDRV.SYS), ist der Treiber nicht korrekt konfiguriert. Bitte überprüfen Sie, ob mit dem Programm DRVCONFIG.EXE, ob die Standard Karte „PCI Board“ eingetragen ist.

Falls Sie Visual C++ benutzen, so ist es möglich, die Library Datei SPECTRUM.LIB mit in ein Projekt zu integrieren, um die Funktionen des Treibers auf einfache Weise in das Programm einzubinden. Die Library Datei arbeitet nicht mit Borland Compilern zusammen.

Es werden die beiden DLL's SPECTRUM.DLL und SPCSTDNT.DLL installiert. Die beiden DLL's unterscheiden sich nur im Aufruf der Funktionen. Die Datei SPECTRUM.DLL exportiert die Funktionen als `_cdecl` (für C, C++, Delphi), die Datei SPCSTDNT.DLL als `_stdcall` (für Visual Basic). Je nach benutztem Compiler kann eine der beiden DLL's benutzt werden.

C++, Delphi), the file SPCSTD95.DLL uses `_stdcall` definition (for Visual Basic). Depending on the used programming language, one of the two DLL's may be used.

Windows NT (Legacy)

The driver consists of a kernel mode driver for Windows NT and a 32 bit windows DLL which uses the functions of the kernel mode driver. Examples for Microsoft Visual C++, Borland Delphi and Microsoft Visual Basic are included.

Windows NT

- (1) Login as ADMINISTRATOR or with another account having the right to install drivers and to change the registry.
- (2) Start the setup program on the driver CD. The installation program is found in the directory `\Install\WinNTDrv`.
- (3) The installation routine will install the kernel mode driver, the 32 bit windows DLL and some utilities in the program folder 'Spectrum GmbH'. It will also update the registry.
- (4) Restart the computer
- (5) The PCI boards are automatically detected by the kernel driver.
- (6) If the service does not start correct (A message in the event log from the service SpcDrv.SYS), the driver is not setup correctly. Please run DRVCONFIG.EXE and check whether the standard board is correctly set to "PCI Board"

If you are using Microsoft Visual C++, you may use the delivered library file SPECTRUM.LIB to access the driver functions easily. The library file will not work with Borland compilers. The both DLL's SPECTRUM.DLL and SPCSTDNT.DLL are installed. The only difference between the both DLL's is the calling convention. The file SPECTRUM.DLL uses `_cdecl` definition (for C, C++, Delphi), the file SPCSTD95.DLL uses `_stdcall` definition (for Visual Basic). Depending on the used programming language, one of the two DLL's may be used.

Installation für Linux

Der Treiber besteht aus einem ladbaren Kernel Modul für alle Karten. Beispiele für Gnu C werden ebenfalls mitgeliefert.

Login

Loggen Sie sich als root ein oder als Benutzer mit dem Recht Module zu laden und Devices anzulegen.

Auswahl des richtigen Treibers

Die Verwendung von Linux-Kernel-Modulen hängt stark von der Kernelversion sowie der verwendeten Distribution ab. Diesem Umstand Rechnung tragend werden die Spectrum Treiber in verschiedenen Versionen ausgeliefert. Bitte wählen Sie das am besten passende Archiv für Ihre Installation aus.

Treiber laden

Der Linux Treiber wird als ladbares Kernel Modul spc.o ausgeliefert. Der Treiber enthält alle Spectrum PCI, CompactPCI und ISA Karten. Die PCI und CompactPCI Karten werden automatisch erkannt.

Laden Sie das Modul mit „insmod -f spc.o“.

Der insmod Befehl kann die Warnung generieren, daß das Kernel Modul für eine andere Kernel Version kompiliert wurde. Dies Meldung können Sie ignorieren.

Wenn das Kernel-Modul nicht in Ihre Linux Installation geladen werden kann, so ist es notwendig den Treiber auf Ihrem System neu zu kompilieren. Bitte setzen Sie sich mit Spectrum in Verbindung, um die benötigten Sourcedateien zu bekommen.

Major Number

Für den Zugriff auf den Treiber benötigen Sie die zugeteilte Major number. Sie finden diese Zahl in /proc/devices. Der Treiber trägt den Namen „spec“. Normalerweise ist diese Nummer 254, kann aber auch je nach vorher installierten Treibern davon abweichen.

Device anlegen

Als letzten Schritt muß ein Device mit dem Treiber verknüpft werden. Dieses geschieht über den Befehl mknod. Als Major number wird die in /proc/devices gefundene Zahl eingetragen. Als Minor Number der Index der Karte die angesprochen wird. Die Indexzählung beginnt bei 0.

„mknod /dev/spc0 c 254 0“ für die erste Karte
 „mknod /dev/spc1 c 254 1“ für die zweite Karte

Stellen Sie sicher, daß alle Benutzer, die mit dem Treiber arbeiten müssen Schreibrechte für das neu angelegte Device haben. Dafür können Sie allen Personen Schreibrechte für das Device erteilen: `chmod a+w /dev/spc0`.

Ende

Die Karte kann jetzt über das angelegte Device angesprochen werden. Das genaue Vorgehen kann aus den Beispielen entnommen werden.

Nach einem Neustart von Linux ist es nur nötig das Treiber Modul zu laden. Das Device muß nur geändert werden, falls die Major Number nicht mehr stimmt.

Installation for Linux

The driver consists of a loadable kernel module for all boards. Examples for Gnu C are also delivered.

Login

Login as root or login as a user who has the right to load modules and to install devices.

Select the right driver

Linux kernel modules are heavily depending on the kernel version and distribution. Therefore the kernel driver for the Spectrum boards is shipped in different versions. Please select the archive that is best matching your installed version.

Load Driver

The linux driver is shipped as the loadable kernel module spc.o. The driver includes all Spectrum PCI, CompactPCI and ISA boards. All PCI and CompactPCI boards are recognised automatically.

Load the module with “insmod -f spc.o”

The insmod command could generate a warning that the driver module was compiled for an other kernel version. You could ignore this warning.

It is not possible to use the driver module with linux versions prior to kernel version 2.0.

If the kernel module could not be loaded in your linux installation it is necessary to compile the driver directly on your system. Please contact Spectrum to get the needed source files.

Major Number

For accessing the device driver it is necessary to know the major number of the driver. This number is listed in /proc/devices. The device driver is called “spec” in this list. Normally this number is 254 but this depends on the already installed device drivers.

Installing the Device

You connect a device to the driver with the mknod command. The major number is the number found in /proc/devices. The minor number is the index of the board starting with 0.

“mknod /dev/spc0 c 254 0” for the first board
 “mknod /dev/spc1 c 254 1” for the second board

Make sure that the users that should work with the driver has write rights to access the device. Therefore you should give all persons all rights to the device: `chmod a+w /dev/spc0`

End

The board could now be accessed using the device. See the example files for more information.

After restarting linux it is only necessary to load the driver again. The device must only be changed if the major number has changed.

Der Zugriff auf das Linux Device erfolgt mit Read und Write Befehlen sowie ioctl Befehlen. Eine Umsetzung dieser Befehle in die Standard Treiber Schnittstelle von Spectrum kann über die Datei „spciocctl.inc“ realisiert werden. Das genaue Vorgehen ist aus den Beispielen ersichtlich.

Info

Informationen über die installierte Spectrum Karten können unter /proc/spectrum abgefragt werden. Für ISA Karten ist hier der Typ und die Basisadresse sichtbar. Für PCI Karten sind alle grundlegenden Informationen aus dem onboard EEPROM aufgelistet.

Hilfsprogramme

SBench 5.x

Auf der CD wird eine Vollversion von SBench 5.x mitgeliefert. Das Programm unterstützt alle aktuellen Erfassungs-, Ausgabe- und Digital I/O Karten von Spectrum. Je nach verwendeter Karte und nach Konfiguration des Programms kann SBench als Digitales Speicheroszilloskop, als Spectrumanalyser, als Logikanalyser oder einfach als Datenerfassungssystem benutzt werden. Verschiedenen Import- und Exportfunktionen erlauben die einfache Nutzung von SBench mit diversen anderen Programmen.

Eine Installationsversion ist im Verzeichnis /Install/SBench5 auf der CD zu finden. Im Verzeichnis /Manuals auf der CD ist eine kurze Anleitung zur Bedienung von SBench in Deutsch und Englisch zu finden. Eine aktuelle Version ist jederzeit aus dem Internet unter www.spec.de zu bekommen.

DRVCONFIG.EXE

Automatisch installiert im Ordner ‚Spectrum GmbH‘ bei der Installation des Windows NT Treibers. Dieses Programm erlaubt die Änderung der Treiber Konfiguration der Spectrum ISA Karten unter Windows NT. Für PCI Karten braucht das Programm nicht benutzt werden. Das Programm ändert die Eintragungen in der Registry. Die neue Konfiguration wird beim nächsten Start des Systems benutzt.

PCITEST.EXE

Zu finden auf der Treiber CD im Verzeichnis \UTILS. Dieses Hilfsprogramm sammelt alle verfügbaren Informationen über alle im System installierten Spectrum PCI Karten. Die Informationen werden aus dem on-board EEPROM ausgelesen und angezeigt. Das Programm läuft nur unter DOS oder in der DOS-Box von Windows 3.11 oder Windows 9x/ME. Das Programm läuft nicht unter Windows NT/2000/XP.

Accessing the linux device is done with read and write commands and ioctl commands. These commands could be converted to the standard Spectrum driver interface with the file “spciocctl.inc”. See the examples for this.

Info

Information about the installed boards could be found in the /proc/spectrum file. For ISA boards the board type and the base address are listed. For PCI boards the basic information from the onboard EEPROM is listed.

Utilities

SBench 5.x

A full version of SBench 5.x is delivered with the board on CD. The program supports all actual acquisition, generator and digital I/O boards from Spectrum. Depending on the used board and the software setup, one could use SBench as a digital storage oscilloscope, a spectrum analyser, a logic analyser or simply as a data recording front end. Different export and import formats allow the use of SBench together with a variety of other programs.

An install version of the program is found in the directory /Install/SBench5 on CD. There is also a short program description in german and english in the /Manuals directory.

A current version could be downloaded from the internet at www.spec.de at any time.

DRVCONFIG.EXE

Installed in the folder ‚Spectrum GmbH‘ when installing the Windows NT driver. This utility manages the driver configuration of the Spectrum ISA boards for Windows NT. The program need not to be used for PCI boards. The utility changes the registry. The new configuration will only be used after the next reboot of the system.

PCITEST.EXE

Found on the driver CD in the directory \UTILS. This utility will collect some information about all installed Spectrum PCI boards. The information of the onboard EEPROM will be read out and shown. The utility will only work with DOS, Windows 3.1x, Windows 9x and Windows ME. It will not work with Windows NT/2000/XP.

Hardware Description

Trigger Informationen

Nach dem Start der PCI.DIO32 werden die Eingänge abgetastet und die Daten im Speicher abgelegt, bzw. ausgegeben (Der Speicher arbeitet als Ringbuffer, die Daten werden kontinuierlich in den Speicher geschrieben / ausgelesen). Die Triggerereignisse werden ignoriert bis der programmierte Speicher einmal komplett mit Daten gefüllt ist. Danach wird die Triggerverarbeitung freigeschaltet.

Wenn der Software Trigger ausgewählt wurde, wird sofort ein Triggerereignis erkannt. Wird der TTL Trigger benutzt, so wird ein Triggerereignis erkannt, wenn der Trigger Eingang von LOW Pegel zu HIGH Pegel wechselt (steigende Flanke) oder von HIGH Pegel zu LOW Pegel wechselt (fallende Flanke).

Eine weitere Triggerart ist die Patterntriggerung. Hierbei wird ein Muster vorgegeben, bei dem der Trigger ausgelöst werden soll. Außerdem kann noch angegeben werden, wie viele Takte dieses Muster anstehen muß, bevor getriggert wird. Zusätzlich ist es noch möglich, für ein Eingangssignal eine Flanke anzugeben, die den trigger auslösen soll.

Der Status ändert sich zu ‚Trigger found‘ und der Postcounter fängt an den programmierten Posttrigger Wert herunter zu zählen. Wenn dieser Wert Null erreicht, stoppt die PCI.DIO32 und das Status ändert sich auf ‚Ready‘.

Trigger Information

After the PCI. DIO32 has been started it samples the input signals and stores the converted data to the memory. (The memory operates as a circular buffer, so data are written continuously to the RAM). No trigger events are processed until the programmed memory is filled one time completely with data. Afterwards the trigger sequencer will be enabled.

If Software trigger is used a trigger event is detected immediately. Using the TTL trigger will cause a trigger event if the external TTL input will go from low to high (rising edge) or from high to low (falling edge).

An other trigger mode is the pattern trigger. Additional it is possible to specify how long this pattern must be valid before the trigger event is detected. It is also possible to select an edge for one channel for generating a trigger event.

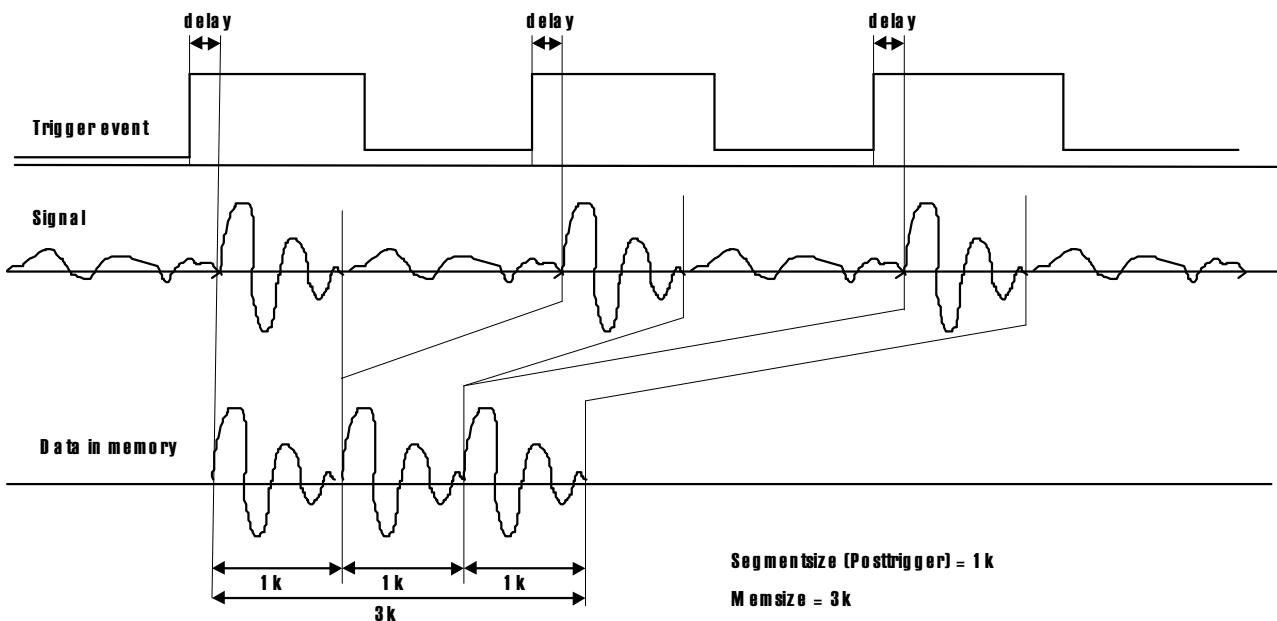
The status will be change to ‚trigger found‘ and the postcounter starts counting down the posttrigger value. After the postcounter reaches zero the PCI.DIO32 stops and signals ‚ready‘ in the status register.

Option Multiple Recording

Die Option Multiple Recording erlaubt die Aufnahme/Ausgabe mehrerer Triggerereignisse, ohne die Hardware dazwischen neu zu starten. Der Speicher der Karte wird in mehrere gleich große Segmente unterteilt. Jedes Segment wird bei Auftreten eines Triggerereignisses mit Daten gefüllt. Im Multiple Recording Modus ist kein Pretrigger möglich.

Option Multiple Recording

The option Multiple Recording allows the recording/replay of several trigger events without restarting the hardware. The memory of the board will be divided into several segments of the same size. Each segment will be filled with data when a trigger event occurs. Pretrigger is not available when using Multiple Recording

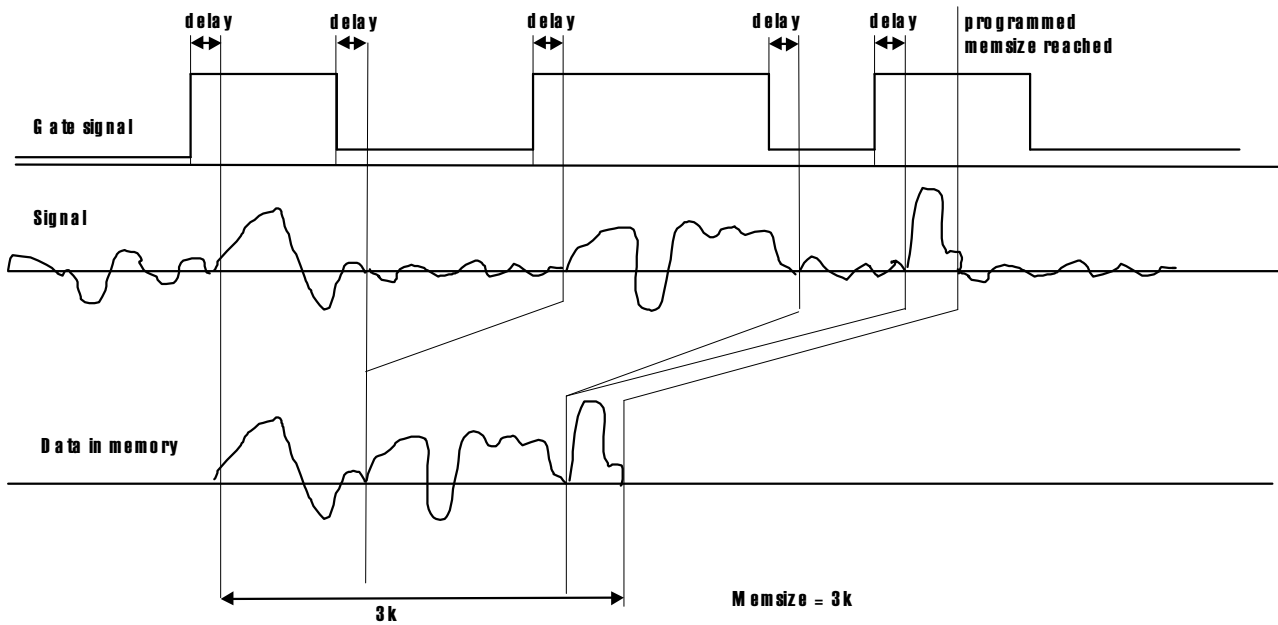


Option Gated Sampling

Die Option Gated Sampling erlaubt die Aufnahme/Wiedergabe eines Signals gesteuert über ein Gatesignal. Es werden nur Daten aufgenommen, wenn das Gatesignal einem programmierten Pegel (TTL HIGH oder TTL LOW) entspricht.

Option Gated Sampling

The option Gated Sampling allows recording/replay of a signal controlled by an gate signal. Data is only recorded if the gate signal is equal to a programmed level (TTL HIGH or TTL LOW).

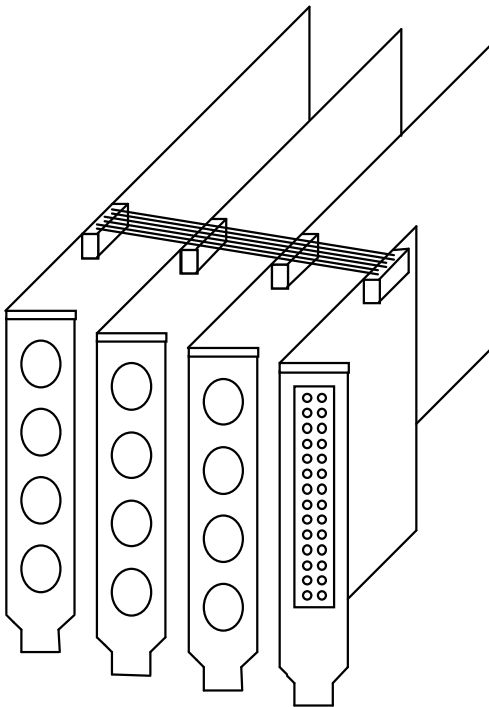


Option Synchronisierung

Diese Option erlaubt es, mehrere Karten von Spectrum miteinander intern zu synchronisieren, um auf einfache Art und Weise Mehrkanalsystem aufzubauen. Es ist ebenso möglich, synchrone Systeme aufzubauen, die in einem Rechner sowohl analoge Kanäle als auch digitale Kanäle zusammen aufzeichnen.

Die Karten können innerhalb eines Kartentyps synchronisiert werden. Genauso ist es möglich verschiedene Spectrum Karten, die alle den *SPC100-SyncBus* unterstützen, miteinander frei zu kombinieren.

Eine Karte wird als Master konfiguriert und generiert den Takt und die Triggerinformation für die anderen (Slave) Karten. Alle Karten laufen synchron mit exakt dem gleichen Abtasttakt.



Option Synchronisation

This option allows it to connect several boards from Spectrum to generate a multi-channel system. It is also possible to build up synchronously mixed mode systems to record analogue channels and digital channels together.

The boards may be connected in one type of board with each other. Also it is possible to connect Spectrum boards of different types that uses the *SPC100-SyncBus* together.

One board is dedicated as the master board and generates clock and trigger signals for the other (slave) boards. All boards are running synchronously.

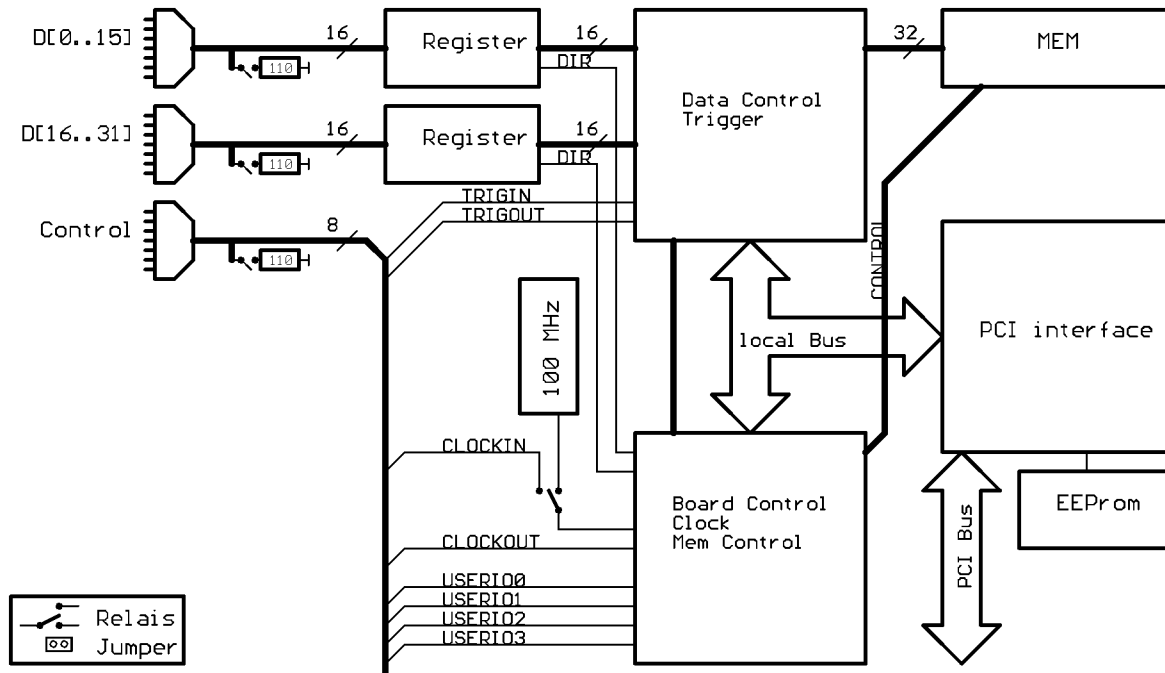
Example configuration for Multi-Channel Systems:

- 3 boards PCI.248 32MSample memory
- 1 board PCI.DIO32 32MSample memory
- Synchronisation cable

forming a synchronously system with

- 6 analogue channels 8 bit resolution
- 32 digital channels
- samplerate 1.5 MHz to 400 MHz
- 96 MByte on-board memory
- 16 MSample memory per channel

Block Diagram



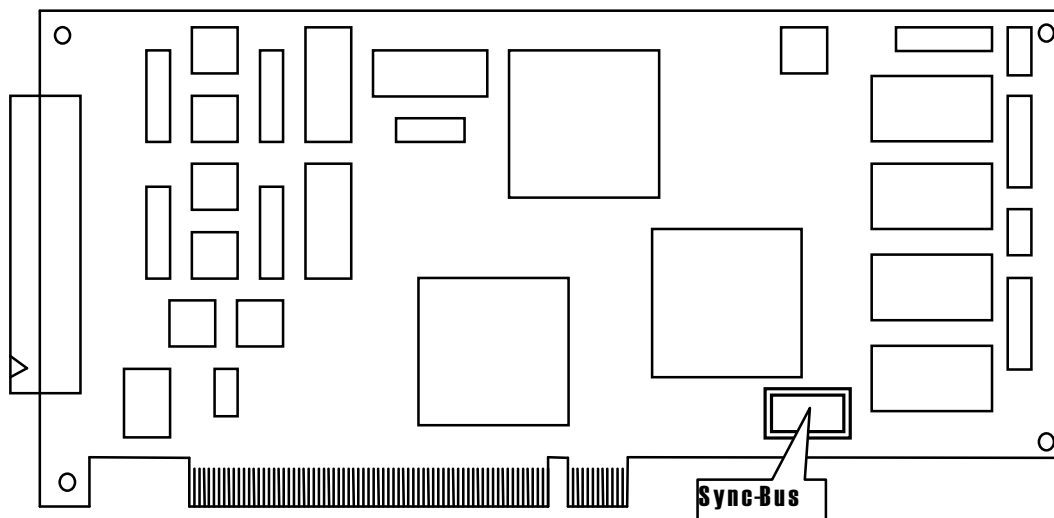
Technical data

Samplerate 3.33 MHz up to 100 MHz
 Input impedance 110 Ohm or 50 kOhm || 15 pF
 Signal level TTL
 Multi: Trigger to 1st sample delay 26 samples
 Multi: Recovery time 21 samples
 Trigger output delay 5 samples
 Trigger accurathy 1 sample
 Ext. clock: output delay 12.5 ns
 Ext. clock: delay to internal clock 9.0 ns
 Sync: board to board trigger jitter 0 samples
 Sync: board to board clock delay ≤ 1 ns

Dimension 203 mm x 109 mm
 Connector 80 pin (2 * 40 pin flat ribbon cable 2.54 mm)
 Operating temperature 0°C - 50°C
 Storage temperature -10°C - 70°C
 Humidity 10% to 90% non condensing

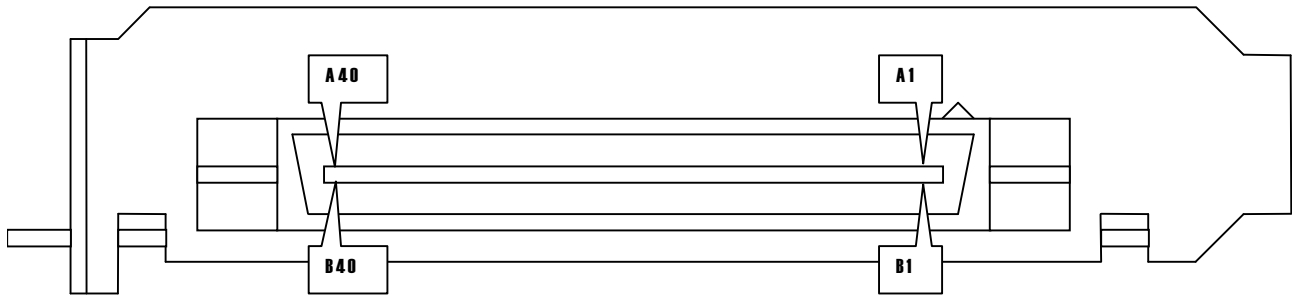
	+3.3 V	+5 V	+12 V	-12 V
Power consumption (A)	0 mA	2500 mA	0 mA	0 mA
Power consumption (W)	0.0 W	12.5 W	0.0 W	0.0 W

Placement



Connector

The PCI.DIO32 has an 80 pin connector for two 40 pole 2.54 mm flat ribbon cable.



signal	pin	pin	signal
GND	A1	B1	GND
ExtClkIn	A2	B2	ClkOut
GND	A3	B3	GND
User0In	A4	B4	User0Out
GND	A5	B5	GND
User1In	A6	B6	User1Out
GND	A7	B7	GND
ExtTrigIn	A8	B8	TrigOut
GND	A9	B9	GND
D16	A10	B10	D0
GND	A11	B11	GND
D17	A12	B12	D1
GND	A13	B13	GND
D18	A14	B14	D2
GND	A15	B15	GND
D19	A16	B16	D3
GND	A17	B17	GND
D20	A18	B18	D4
GND	A19	B19	GND
D21	A20	B20	D5
GND	A21	B21	GND
D22	A22	B22	D6
GND	A23	B23	GND
D23	A24	B24	D7
GND	A25	B25	GND
D24	A26	B26	D8
GND	A27	B27	GND
D25	A28	B28	D9
GND	A29	B29	GND
D26	A30	B30	D10
GND	A31	B31	GND
D27	A32	B32	D11
GND	A33	B33	GND
D28	A34	B34	D12
GND	A35	B35	GND
D29	A36	B36	D13
GND	A37	B37	GND
D30	A38	B38	D14
GND	A39	B39	GND
D31	A40	B40	D15

Sync Bus

Carries the signals for synchronisation of multiple PCI.DIO32

- Pin 1 Sync Clock
- Pin 3 Sync Trigger
- Pin 5, 7, 9 not used
- Pin 2, 4, 6, 8 GND

Software Description

Allgemeine Information

Der Spectrum Treiber besteht aus einem Satz Funktionen zur Manipulation der Register auf der Karte und zum Daten Transfer in beide Richtungen. Es gibt nur einen Treiber für alle Karten von Spectrum. Abhängig von der Funktionalität der Karte und dem benutzten Bus werden nicht alle Funktionen des Treibers von allen Karten unterstützt. Die unterschiedliche Funktionalität der Karten ist mit Hilfe von kartenspezifischen Registern realisiert. Der Treiber ist für verschiedene Betriebssysteme erhältlich und wird unter allen Betriebssystemen auf die gleiche Art und Weise programmiert.

Header Dateien auf CD

DLLTYP.H

Enthält alle Plattform spezifischen Definitionen der Datentypen und der Funktionsdeklarationen. Alle Datentypen basieren auf diesen Definitionen.

SPECTRUM.H

Definiert die sechs Funktionen des Treibers. Alle Definitionen sind aus der Datei DLLTYP.H entnommen. Die Funktionen selbst werden weiter unten beschrieben.

REGS.H

Definiert alle Register und Kommandos, die im Spectrum Treiber für die verschiedenen Karten benutzt werden. Die Register, die von einer Karte benutzt werden sind weiter unten im kartenspezifischen Teil beschrieben.

ERRORS.H

Listet alle möglichen Errorcodes der Funktionen auf.

Funktionen des Treibers

Der Spectrum Treiber besteht aus den folgenden sechs Funktionen. Die Funktionen sind in der Header-Datei SPECTRUM.H definiert. Abhängig von dem Funktionsumfang der Karte und dem verwendeten Bussystem sind nur einige der Funktionen für die spezielle Karte notwendig. Bei einigen Karten werden nicht alle Parameter der Funktion unterstützt.

General Information

The SPECTRUM driver consists of a set of functions to manipulate registers on the board and to transfer data from or to the board. There is only one driver for all the SPECTRUM boards. Depending on the functionality of the board and the used bus not all functions will be implemented for all boards. The different functionality of the boards is implemented with the help of board specific registers. The driver is available for different operating systems but will be programmed the same way on all operating systems.

Header files on CD

DLLTYP.H

Includes the platform specific definitions for data types and function declarations. All data types are based on this definitions.

SPECTRUM.H

Defines the six functions of the driver. All definitions are taken from the file DLLTYP.H. The functions itself are described below.

REGS.H

Defines all registers and commands which are used in the SPECTRUM driver for the different boards. The registers a board uses are described in the board specific part of the documentation.

ERRORS.H

Lists all possible error codes of the functions.

Driver functions

The SPECTRUM driver consists of the following six functions. The functions are declared in the header file SPECTRUM.H. Depending on the functionality of the board and the used bus only some of the functions are used for the specific board. Not all board specific drivers will interpret all parameters of a function.

	PAD52	PAD82a/b	PAD242	PCI.412	PCI.212	PCI.208	CPCI.208	PCI.248	PCI.258	PCI.DIO32	PAD1232	PAD1616	PAD164	DAPI16	PCK400	TRS582	PADCO06	MI.30xx	MI.31xx	MI.40xx	MI.45xx	MI.60xx	MI.70xx
SpcInitPCIBoards	-	-	-	+	+	+	+	+	+	+	-	-	-	-	-	-	-	+	+	+	+	+	+
SpcInitBoard	+	+	+	-	-	-	-	-	-	-	+	+	+	+	+	+	+	-	-	-	-	-	-
SpcSetParam	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+
SpcGetParam	+	+	+	+	+	+	+	+	+	+	+	+	+	-	+	+	+	+	+	+	+	+	+
SpcSetData	-	-	-	-	-	-	-	-	-	+	-	-	-	+	-	+	-	-	-	-	-	+	+
SpcGetData	+	+	+	+	+	+	+	+	+	+	+	+	+	-	-	-	-	+	+	+	+	-	+

int16 SpcInitPCIBoards (int16* count, int16* PCIVersion)

<i>count</i>	adr of 16 bit integer	number of found PCI boards
<i>PCIVersion</i>	adr of 16 bit integer	found PCI version
<i>return</i>	16 bit integer	error code of function like listed below

Initialises all installed PCI boards. The board numbers will start with zero. The number of PCI boards will be given back in the value *Count*. All installation parameters will be read from the hardware.

Using Windows NT the boards are already installed in the registry. This function just gives back the values of the kernel driver.

Linux initialises the boards while loading the kernel module. This function is not available under Linux.

int16 SpcInitBoard (int16 nr, int16 typ)

<i>nr</i>	16 bit integer	number of the board to be defined in range 0-15
<i>typ</i>	16 bit integer	type of the defined board listed in REGS.H
<i>return</i>	16 bit integer	error code of function like listed below

Defines a board for the driver. The driver supports up to 16 boards at the same time. For all ISA boards the type of installed board must be defined before using the driver the first time. All other functions just use the board number to access the board. After initialising the board all parameters will be set to default values.

Using Windows NT the board is already installed in the registry. This function will then just compare the board type with the already installed one.

Linux initialises the boards while loading the kernel module. This function is not available under Linux.

int16 SpcSetParam (int16 nr, int32 reg, int32 value)

<i>nr</i>	16 bit integer	number of the board as defined by SpcInit...
<i>reg</i>	32 bit integer	register to be changed
<i>value</i>	32 bit integer	value for the register
<i>return</i>	16 bit integer	error code of function like listed below

Sets a register to a defined value or executes a command. The board must be initialised before. When using ISA boards, all installation parameters must be set before (address, installed memory, ...). The allowed registers for the driver are listed in the board specific part of the documentation.

When using Windows NT the installation parameters may not be changed, they are set in the registry using the driver configuration utility.

int16 SpcGetParam (int16 nr, int32 reg, int32* value)

<i>nr</i>	16 bit integer	number of the board as defined by SpcInit...
<i>reg</i>	32 bit integer	register to be read
<i>value</i>	adr of 32 bit integer	value from the register
<i>return</i>	16 bit integer	error code of function like listed below

Reads a register or a status information of the board. The board must be initialised before. When using ISA boards, the installation address must be set before. The allowed registers for the driver are listed in the board specific part of the documentation.

int16 SpcSetData (int16 nr, int16 ch, int32 start, int32 len, dataptr data)

<i>nr</i>	16 bit integer	number of the board as defined by SpcInit...
<i>ch</i>	16 bit integer	channel to be written to
<i>start</i>	32 bit integer	startvalue to be written
<i>len</i>	32 bit integer	number of values to be written
<i>data</i>	huge ptr to data	data to be written
<i>return</i>	16 bit integer	error code of function like listed below

Writes data to the board for a specific channel. The board must be initialised before. When using ISA boards, all installation parameters must be set before (address, installed memory, ...). The Start and Len parameter are implemented on all PCI boards. On ISA boards the whole data will be written in one turn. The data must be in two's complement format (standard integer format).

int16 SpcGetData (int16 nr, int16 ch, int32 start, int32 len, dataptr data)

<i>nr</i>	16 bit integer	number of the board as defined by SpcInit...
<i>ch</i>	16 bit integer	channel to be read
<i>start</i>	32 bit integer	startvalue to be read
<i>len</i>	32 bit integer	number of values to be read
<i>data</i>	huge ptr to data	data space for read values
<i>return</i>	16 bit integer	error code of function like listed below

Reads data from the board from a specific channel. The board must be initialised before. When using ISA boards, all installation parameters must be set before (address, installed memory, ...). The Start and Len parameter are implemented on all PCI boards. On ISA boards the whole data will be read in one turn. The read out data is in the two's complement format and could be directly used for data processing as standard integer values.

Error Codes

error name	value (hex)	value (dec.)	description
ERR_OK	0	0	Execution OK, no error.
ERR_INIT	1	1	The board number is not in the range of 0 to 15. When initialisation is executed: the board number is yet initialised, the old definition will be used.
ERR_NR	2	2	The board is not initialised yet. Use the function <i>SpcInitBoard</i> or <i>SpcInitPCIBoards</i> first.-
ERR_TYP	3	3	Initialisation only: The type of board is unknown.
ERR_FNCNOTSUPPORTED	4	4	This function is not supported by the hardware version.
ERR_LASTERR	10	16	Old Error waiting to be read.
ERR_ABORT	20	32	Abort of wait function
ERR_BOARDLOCKED	30	48	Access to the driver already locked by another program. Stop the other program before starting this one.
ERR_REG	100	256	The register is not valid for this type of board.
ERR_VALUE	101	257	The value for this register is not in a valid range, the allowed values and ranges are listed in the board specific documentation.
ERR_FEATURE	102	258	Feature is not installed on this board
ERR_SEQUENCE	103	259	Channel sequence is not allowed.
ERR_READABORT	104	260	Data read is not allowed after aborting the data acquisition.
ERR_NOACCESS	105	261	Access to this register denied. No access for user allowed.
ERR_POWERDOWN	106	262	Not allowed if powerdown mode is activated.
ERR_CHANNEL	110	272	The channel number may not be accessed on the board: Either it is not a valid channel number or the channel is not accessible due to the actual setup (e.g. Only channel 0 is accessible in interlace mode)
ERR_RUNNING	120	288	The board is still running, this function is not available now or this register is not accessible now.
ERR_ADJUST	130	304	Automatic adjustment has reported an error. Please check the boards inputs.
ERR_NOPCI	200	512	No PCI BIOS is found on the system.
ERR_PCIVERSION	201	513	The PCI bus has the wrong version. SPECTRUM PCI boards require PCI revision 2.1 or higher.
ERR_PCINOBORDS	202	514	No SPECTRUM PCI boards found.
ERR_PCICHECKSUM	203	515	The checksum of the board information has failed.
ERR_DMALOCKED	204	516	DMA buffer not available now.
ERR_MEMALLOC	205	517	Internal memory allocation failed.
ERR_FIFOBUFOVERRUN	300	768	Driver buffer overrun in FIFO mode.
ERR_FIFOHWOVERRUN	301	769	Hardware buffer overrun in FIFO mode.
ERR_FIFOFINISHED	302	770	FIFO transfer has been finished, programmed number of buffers has been transferred.
ERR_FIFOSSETUP	309	777	FIFO setup not possible, transfer rate to high (max 250 MB/s)
ERR_TIMESTAMP_SYNC	310	784	Synchronisation to external reference clock failed.

Valid Board Types

board	type(hex)	type (dec)
PAD52	600	1536
PAD82	200	512
PAD82a	210	528
PAD82b	220	544
PAD242	700	1792
PAD1232-10	400	1024
PAD1232-30	410	1040
PAD1232-40	420	1056

board	type(hex)	type (dec)
PAD1616a	500	1280
PAD1616b	510	1296
PAD164/2	900	2304
PAD164/5	910	2320
PADCO-06	1400	5120
PCK400	800	2048
DAP116	100	256
TRS582	1500	5376

board	type(hex)	type (dec)
PCI.212	300	384
PCI.208	1000	4096
PCI.412	1100	4352
PCI.DIO32	1200	4608
PCI.248	1300	4864
PCI.258	1600	5632
MI.3010	3010	12304
...

Hints for programming the boards

Programming an ISA board is done in the following steps:

- * initialise and define boards with function *SpcInitBoard* (Windows NT: utility DRVCONFIG.EXE)
- * set installation parameters like address, installed memory, version with function *SpcSetParam*
- * set user specific parameters and start board (loop)

Programming an PCI board is done by the following steps:

- * initialise PCI boards automatically with function *SpclnitPCIBoards*
- * read out installation parameters for all found PCI boards like version, installed memory
- * set user specific parameters and start board (loop)

If you are using ISA and PCI boards in one system at the same time, use the function *SpclnitPCIBoards* first and initialise the ISA boards after this. The function *SpclnitPCIBoards* uses the first board numbers and will overwrite other definitions.

It is only necessary to define the boards once for the driver with the functions *SpclnitPCIBoards* and *SpclnitBoard*. If you are defining the boards again, you will get an error code from the function and the old definition is still used. You may ignore this error.

Software - Register of PCI.DIO32

These software register are to be used for the functions *SpcSetParam* and *SpcGetParam* of the software driver. All constants are found in the header file REGS.H.

PCI register

These Registers are set by the driver after PCI initialisation. The information is found in the on-board ROM. The program PCITEST.EXE on the driver disk will give the same information's.

register name	reg no.	r/w	
SPC_PCITYP	2000	r	type of board as listed above
SPC_PCIVERSION	2010	r	board revision: high part in bit 8..15, Low part in bit 0..7
SPC_PCIDATE	2020	r	production date: month in bit 0..7, year in bit 16..31
SPC_PCISERIALNR	2030	r	serial number of the PCI.DIO32
SPC_PCISAMPLERATE	2100	r	max. samplerate as 32 bit integer value
SPC_PCIMEMSIZE	2110	r	installed memory in bytes as 32 bit integer value
SPC_PCIFEATURES	2120	r	installed features as a bitfield. See description beneath.

PCI Features register

- Bit 31 not used
- ...
- Bit 8 not used
- Bit 7 option synchronisation master: board is clock-master for synchronisation
- Bit 6 option synchronisation slave : board is clock-slave for synchronisation.
- Bit 5 option gated sampling installed
- Bit 4 not used
- Bit 3 not used
- Bit 2 not used
- Bit 1 not used
- Bit 0 option multiple recording installed

Error registers

If one action caused an error in the driver this error and the register and value where it occurs will be saved. The driver is then locked until the error is read out using the *SPC_LASTERRORCODE* function. All other functions will lead to the same errorcode unless the error is cleared by reading *SPC_LASTERRORCODE*.

Name	value (dec)	r/w	
SPC_LASTERRORCODE	999999	r	errorcode of the last error as defined in errors.h
SPC_LASTERRORREG	999998	r	software register which causes the error
SPC_LASTERRORVALUE	999997	r	value which causes the error
SPC_LASTERRORTEXT	999996	r	Copies a short explanation of the error to a string. The argument value must be a pointer to a string with at least ERRORTXTLEN characters.

Status register

Status information can be read at any time. The other parameters can only be written and the data can be read if the board is stopped.

register name	reg no.	r/w	
SPC_STATUS	10	r	status register, values listed below.

status code	value	
SPC_RUN	0	board is running.
SPC_TRIGGER	10	trigger has been found.
SPC_READY	20	recording has stopped.

Command register

The command register executes commands like start and stop or synchronises the board with other boards.

register name	reg no.	r/w	
SPC_COMMAND	0	r/w	command register, allowed values listed below.

status code	value	
SPC_RESET	0	The board hardware and logic is cleared. The driver settings are cleared too.
SPC_START	10	starts the board with the current register settings.
SPC_STOP	20	stops the board, data in memory is undefined.
SPC_SYNCMASTER	100	synchronisation with internal SPC100 sync bus, this board works as master
SPC_SYNCSLAVE	110	synchronisation with internal SPC100 sync bus, this board works as slave
SPC_NOSYNC	120	no synchronisation
SPC_SYNCTRIGGERMASTER	101	synchronisation: board generates trigger for all boards
SPC_SYNCTRIGGERSLAVE	111	synchronisation: board receives trigger from trigger master board.

Synchronisation (Option)

See Hardware Part of the manual for further details.

This option allows it to connect several boards from Spectrum to generate a multi-channel system. It is possible to connect several PCI.DIO32 with each other as well as to connect the PCI.DIO32 with other Spectrum boards using the *Spc100 Sync-Bus*. One board is the clock master and generates the clock for the other (slave) boards. The clock master is defined in hardware and has the synchronisation master bit set in the PCI features register(see above). Only one board may be the clock master. At runtime any of the synchronised boards may be defined as a trigger master and generates trigger information for the other boards.

If the boards are synchronised, they must be programmed in the following steps:

- (1) Set all parameters for all boards except the sync information
- (2) Set the sync information for the clock-master board.
- (3) Set the sync information for all clock-slave boards.
- (2) Start all trigger-slave boards.
- (3) Start the trigger-master board.

All boards will run with the clock generated by the clock-master board. Only the trigger-master board may generate a trigger. The trigger settings for the trigger-slave boards will be ignored.

Memory register

This register holds the number of samples, not the number of bytes.

register name	Reg no.	r/w	
SPC_MEMSIZE	10000	r/w	memory size for recording: 16 samples up to <i>installed mem</i> /4 samples with steps of 16 samples.

Posttrigger register

Sets the number of samples to be recorded AFTER the triggerevent has been found. The corresponding pretrigger is calculated by the formula: $pretrigger = memsize - posttrigger$

If the posttrigger value is higher than the programmed memsize, the triggerevent is not visible.

If the option Multiple Recording is used, this register holds the segmentsize.

register name	Reg no.	r/w	
SPC_POSTTRIGGER	10100	r/w	posttrigger value in the range 8 samples up to 32 MSamples with steps of 8.

UserxIn/Out register

This register holds the level of the UserxIn/Out signals. A write to the UserxOut Signals is only possible, when the SPC_USEROUT feature bit is set and SPC_EXTERNOUT is enabled.

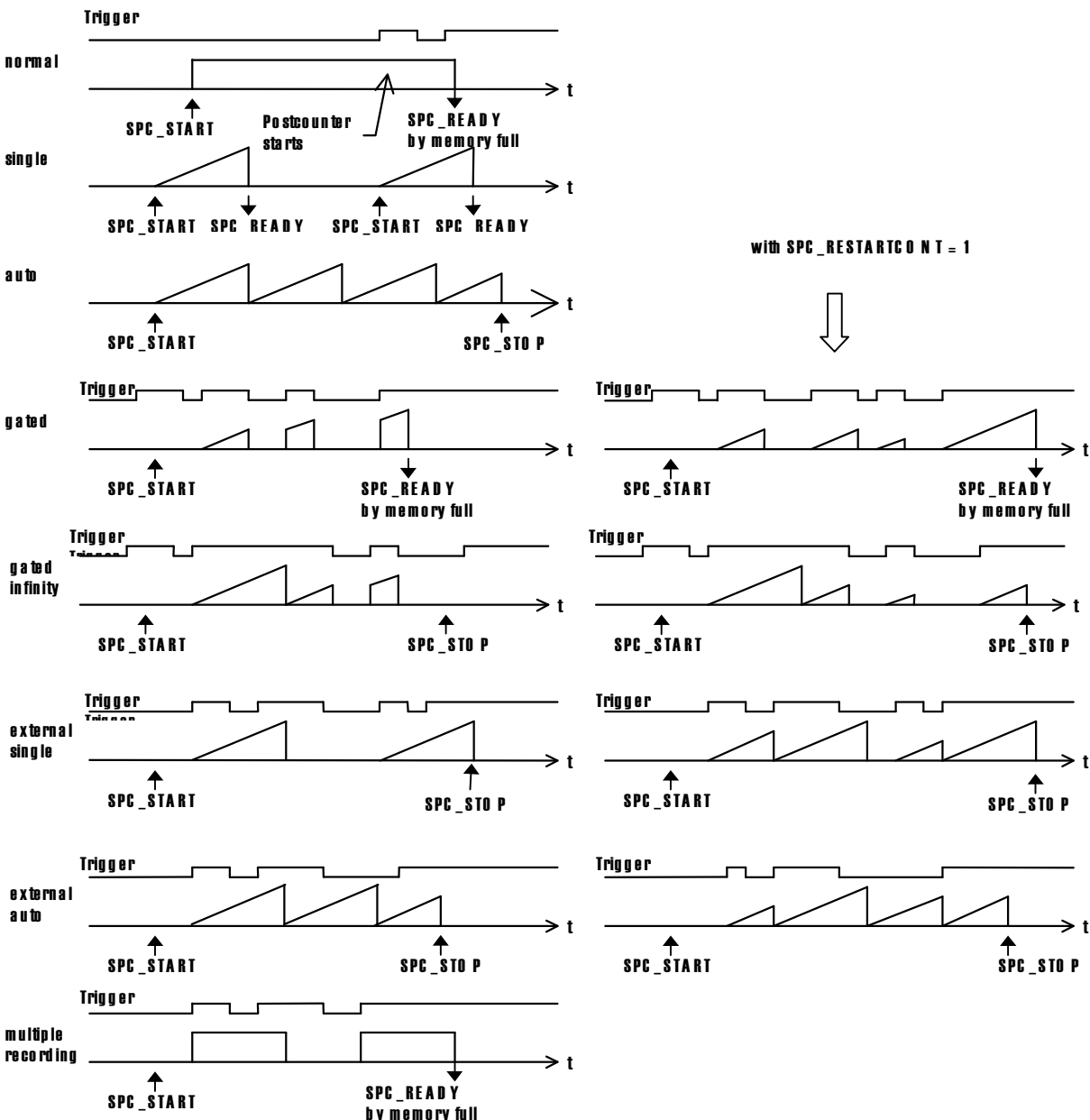
register name	Reg no.	r/w	
SPC_WRITEUSER0	230100	w	Sets the level of the User0Out signal to low (0) or high (1).
SPC_WRITEUSER1	230110	w	Sets the level of the User1Out signal to low (0) or high (1).
SPC_READUSER0	230200	r	Get the level of the User0In signal.
SPC_READUSER1	230210	r	Get the level of the User1In signal.

Features register

All of this features registers may be set by writing a 1 or cleared by writing a 0. Some features may only be used if this features is installed on the board (see PCI Features above).

register name	Reg no.	r/w	
SPC_EXTERNALCLOCK	20100	r/w	the external clock will be used for recording. The external clock is allowed in the range 3.33 MHz to 100 MHz. Between the SPC_START command and the board signals SPC_READY or the SPC_STOP command is send the external clock must not stop or pause.
SPC_EXTERNOUT	20110	r/w	enables the clock, user out 0, user out 1 and trigger output of the board
SPC_DIRECTIO	260000	r/w	The data is direct read/write to the connector. Every access to the function SpcSetData and SpcGetData will directly write or read one data sample to the connector.
SPC_USEROUT	230010	r/w	Enables the both UserxOut Signal to set with software commands. Otherwise the User0Out Signal generates a high to low edge, after the board has started and is ready to accept a trigger event. At the recording has stopped, the opposite edge occurs. The signal User1Out carries out the inverted level from User0Out.
SPC_SINGLESHOT	41000	r/w	The memory is filled/sending out once without the use of posttrigger.
SPC_RESTARTCONT	41200	r/w	(from version 2.32 on) Starts the patternoutput at the beginning of the memory after a new trigger event is detected.
SPC_OUTONTRIGGER	41100	r/w	(from version 2.32 on) Starts the patternoutput after a trigger event is detected.

Modes of operation (from version 2.32 on)



The following table shows which flags and register values are necessary to enable the different modes.

mode	SPC_TRIGGERMODE	SPC_RESTARTCONT	SPC_SINGLESHOT	SPC_OUTONTRIGGER	SPC_MULTI	special for
normal	TM_SOFTWARE, TM_PATTERN, TM_PATTERNANDEDGE, TM_TTL	0	0	0	0	input
single	TM_SOFTWARE	0	1	0	0	output
auto		1	0	0	0	
gated	TM_GATEDLOW,	0/1	1	0	0	
gated infinity	TM_GATEDHIGH, TM_GATEDPATTERN	0/1	0	0	0	
external single	TM_PATTERN,	0/1	1	1	0	
external auto	TM_PATTERNANDEDGE, TM_TTL	0/1	0	1	0	
multiple recording	TM_PATTERN, TM_PATTERNANDEDGE, TM_TTL	0	0	0	1	input

Attention is necessary for the SPC_RESTARTCONT flag. When the board send out data and (periodic) trigger events occurs just before all memory patterns are send out, the top of the memory is not refreshed. So the data is corrupted, when after a time the hole memory is send out. To prevent the memory to lost data it is necessary to send out the hole memory once every 64 ms.

Triggermode register

register name	Reg no.	r/w	
SPC_TRIGGERMODE	40000	r/w	triggermode for recording.

triggerrmodes	Value	
TM_SOFTWARE	0	recording will start immediately
TM_PATTERN	21000	wait for the programmed pattern
TM_PATTERNANDEDGE	22000	wait for the programmed pattern and then for the programmed edge on one input
TM_TTL	20020	wait for external TTL trigger
TM_GATELOW	30000	(Option from version 2.32 on) wait for external TTL low level
TM_GATEHIGH	30010	(Option from version 2.32 on) wait for external TTL high level
TM_GATEPATTERN	30020	(Option from version 2.32 on) wait for programmed pattern

Triggeredge register

register name	Reg no.	r/w	
SPC_TRIGGEREDGE	46000	r/w	triggeredge for the triggermodes TM_TTL / PATTERNANDEDGE

triggeredge	Value	
TE_POS	10000	wait for rising edge
TE_NEG	10010	wait for falling edge
TE_BOTH	10020	wait for rising and falling edge. The board will trigger on both edges on the selected bit.

Triggermask register

register name	Reg no.	r/w	
SPC_TRIGGERMASK	43100	r/w	triggermask for the triggermodes TM_PATTERN / PATTERNANDEDGE

For a detailed explanation look at the Triggerpattern register below.

Triggerpattern register

register name	Reg no.	r/w	
SPC_TRIGGERPATTERN	43000	r/w	triggerpattern for the triggermodes TM_PATTERN / PATTERNANDEDGE

The Triggerpattern and Triggermask registers set together the Triggerevent. Each DIO bit on the connector corresponds with one bit in both registers. The following bits are only used, if the channel is configured as an input.

Connectorbits	Channel 1							Channel 0						
	D31	D30	D29	...	D18	D17	D16	D15	D14	D13	...	D2	D1	D0
Triggerpattern	31	30	29	...	18	17	16	15	14	13	...	2	1	0
Triggermask														

Triggermask	Triggerpattern	
0	0	0 pattern
0	1	1 pattern
1	0	edge: Only for one bit is it valid to specify a edge. See also the triggeredge register.
1	1	This bit is switched off and not used for the trigger.

Pulsewidth register

Sets the count of valid samples with identical pattern to the programmed one, before the trigger occurs.

register name	Reg no.	r/w	
SPC_PULSEWIDTH	44000	r/w	count of valid samples between 1 and 256

Examples for trigger definition

The board should trigger if bit d0 to d3 are high for at least 10 samples.

```

SPC_TRIGGERMASK      0xFFFFFFFF0
SPC_TRIGGERPATTERN  0xFFFFFFFF
SPC_PULSEWIDTH       10
SPC_TRIGGERMODE     TM_PATTERN
SPC_TRIGGEREDGE     not used
    
```

Bit d0 to d7 should be zero for at least 100 samples. The board should then trigger if an positive edge occurs at bit d0. The board will not trigger if any positive edge occurs on bit d1 to d7.

```

SPC_TRIGGERMASK      0xFFFFF01
SPC_TRIGGERPATTERN  0xFFFFF00
SPC_PULSEWIDTH       100
SPC_TRIGGERMODE     TM_PATTERNANDEDGE
SPC_TRIGGEREDGE     SPC_TRIGGEREDGE_GETE_POS
    
```

Multiple Recording (option)

See the hardware description part of the manual for basic information about multiple recording.

register name	reg no.	r/w	
SPC_MULTI	220000	r/w	enables Multiple Recording for the board

The register memsize holds the total amount of memory to be recorded. The register posttrigger will hold the size of one segment. Recording is started with a fixed delay after the trigger event is found. There is no pretrigger possible in Multiple Recording mode.

The delay between the external trigger event or the programmed pattern and the first sampled data is fixed for each recording. The delay of 26 samples (datainput) is necessary for this board because it works with dynamic RAM and needs refresh cycles to let the data stay in memory when the board is not recording.

Gated Sampling (option from version 2.32 on)

See the hardware description part of the manual for basic information about gated sampling.

register name	reg no.	r/w	
SPC_TRIGGERMODE	40000	r/w	trigger mode set to TM_GATELOW , TM_GHATEHIGH or TM_GATEPATTERN

The gated sampling mode is enabled by settings the triggermode to TM_GATELOW, TM_GATEHIGH or TM_GATEPATTERN. The register memsize holds the total amount of memory to be recorded. The register posttrigger has no function at gated sampling.

The sampling of data starts with the first edge of the external gate signal or with the dedicated Pattern. At trigger mode TM_GATELOW this is the falling edge. At trigger mode TM_GATEHIGH this is the rising edge. For TM_GATEPATTERN it is necessary to use one channel as input. Data is not recorded before the first occurrence of the correct edge even if the programmed gate level is present at the input connector at start time.

The delay between the external trigger event and the first sampled is fix for each recording. The delay of 28 (dataoutput) / 26 (datainput) samples is necessary for this board because it works with dynamic RAM and needs refresh cycles to let the data stay in memory when the board is not recording.

Recording will pause at the end of a gate interval (rising edge on trigger mode TM_GATELOW, falling edge on trigger mode TM_GATEHIGH or the pattern is not valid). Due to the structure of the on board memory, recording may only stop at a 8 alignment. So there will be a delay of 18 samples plus 1 to 8 additional samples recorded after the end of the gate interval.

Example (trigger mode TM_GATEHIGH):

sample	input data	gate signal	data	comments
...	recording is running
-	-	1	-	
0	y0	1	y0	
1	y1	1	y1	
2	y2	1	y2	
...	
60	y60	1	y60	
61	y61	0	y61	end of gate interval
62	y62	0	y62	
63	y63	0	y63	end of recording at 8 byte boundary.
-	-	0	-	
...	
-	-	0	-	
-	-	1	-	start of next gate interval
...	
64	y64	1	y64	first recorded sample after 26 samples delay
65	y65	1	y65	
...	

Gated sampling may not be used together with the option Multiple Recording.

Samplerate register

Sets the samplerate for recording.

register name	Reg no.	r/w	
SPC_SAMPLERATE	20000	r/w	samplerate between 3.33 MHz and 100 MHz

The value is a 32 bit integer in the range from 100 MHz down to 3.33 MHz using an 4 bit divider. Possible values are: 100 MHz, 50 MHz, 50 MHz/2 = 25 MHz, 50 MHz/3 = 16.67 MHz, ..., 50 MHz/15 = 3.33 MHz

Input direction register

A value of 1 switch the channel to an output. A value of 0 switch the channel to an input.
Write data to the board with the function SpcSetData() is only possible, when the specific channel is as output configured.

register name	Reg no.	r/w	
SPC_INOUT0	30070	r/w	input direction of chanel 0 (D15...D0)
SPC_INOUT1	30170	r/w	input direction of chanel 1 (D31...D16)

Input impedance register

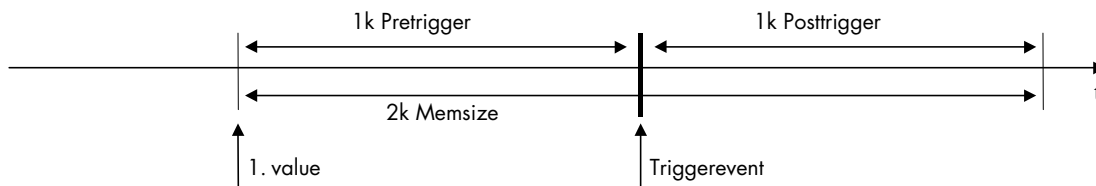
An input impedance of 110 Ohm is set by writing a 1. By writing a 0 the inputs have a high impedance.
When channel 0 or 1 is an output, dont set the impedance to 110 Ohm.

register name	Reg no.	r/w	
SPC_110OHM0	30060	r/w	input impedance of channel 0
SPC_110OHM1	30160	r/w	input impedance of channel 1
SPC_110OHMTRIGGER	30400	r/w	input impedance of the TTL-trigger input
SPC_110OHMCLOCK	30410	r/w	input impedance of the external clock input

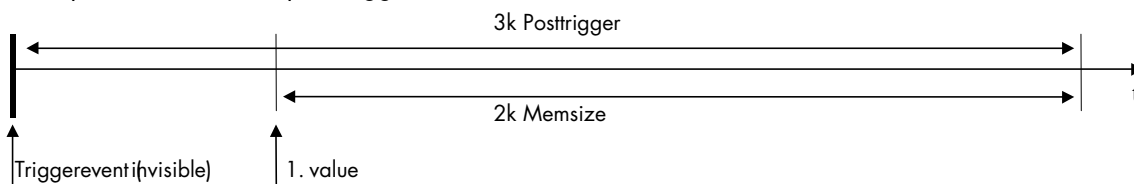
Data (Read/Write)

Data can be read after the board has stopped. The data can be random accessed in blocks of variable length. The trigger event is found at the position memsize - posttrigger.

Example: 2k memsize, 1k posttrigger



Example: 2k memsize, 3k posttrigger



The data may be programmed as 32 bit output, 32 bit input or 16 bit input synchronous to 16 bit output. For this reason, the channel parameter of the functions SpcGetData and SpcSetData has a special meaning. A "0" accesses channel 0 (bit d0 to d15). A "1" accesses channel 1 (bit d16 to d31) and a "2" accesses both channels (bit d0 to d31). Write data to the board with the function SpcSetData() is only possible, when the specific channel is as output configured.

Output	Input		Driver commands	Data format
D0 to D31	—	32 bit pattern output	SpcSetData (Brd, 2, Start, Len, OutData)	32 bit integer
—	D0 to D31	32 bit pattern recording	SpcGetData (Brd, 2, Start, Len, InData)	32 bit integer
D0 to D15	D16 to D31	16 bit pattern output and 16 bit pattern recording	SpcSetData (Brd, 0, Start, Len, OutData) SpcGetData (Brd, 1, Start, Len, InData)	16 bit integer 16 bit integer
D16 to D31	D0 to D15	16 bit pattern output and 16 bit pattern recording	SpcSetData (Brd, 1, Start, Len, OutData) SpcGetData (Brd, 0, Start, Len, InData)	16 bit integer 16 bit integer

Example of Driver use

This example is written for a C or C++ compiler. It reads out the information of the PCI.DIO32, sets the parameters and starts the board once. This file is also found in the example directory on the driver disk.

```
#include <stdio.h>

#include "DLLTYP.H"
#include "SPECTRUM.H"
#include "ERRORS.H"
#include "REGS.H"

main()
{
    int16 Count, PCIVersion;
    int32 Typ, Version, Date, Status;
    int32 data0[10241], data1[10241];

    // ----- Initialisation of PCI Bus -----
    if (SpcInitPCIBoards (&Count, &PCIVersion) != ERR_OK) return 0;
    if (Count == 0) return 0;

    // ----- Test for PCI.DIO32: 1.st board is board number 0 -----
    SpcGetParam (0, SPC_PCITYP, &Typ);
    if (Typ != TYP_PCIDIO32) return 0;

    // ----- Get some PCI Parameters from driver and print them -----
    SpcGetParam (0, SPC_PCIVERSION, &Version);
    SpcGetParam (0, SPC_PCIDATE, &Date);
    printf ("PCI.DIO32 V %x.%x produced %02d.%04d\n", (int16) ((Version>>8)&0x000000FF),
           (int16) (Version&0x000000FF), (int16) (Date&0x000000FF), (int16) (Date>>16));

    // ----- Set Parameters for Recording -----
    SpcSetParam (0, SPC_SAMPLERATE, 1000000001); // samplerate 100 MHz
    SpcSetParam (0, SPC_MEMSIZE, 10241); // memsize 1 kSample
    SpcSetParam (0, SPC_SINGLESHOT, 0); // normal operation
    SpcSetParam (0, SPC_POSTTRIGGER, 5121); // posttrigger 512 Sample
    SpcSetParam (0, SPC_EXTERNALCLOCK, 0); // no external clock
    SpcSetParam (0, SPC_EXTERNOOUT, 0); // no clock and trigger output
    SpcSetParam (0, SPC_TRIGGERMODE, TM_SOFTWARE); // software trigger
    SpcSetParam (0, SPC_TRIGGERPATTERN, 0xffffffff1); // no triggerpattern
    SpcSetParam (0, SPC_TRIGGERMASK, 0xffffffff1); // no triggerpattern
    SpcSetParam (0, SPC_PULSEWIDTH, 11); // triggerwithd one sample
    SpcSetParam (0, SPC_TRIGGEREDGE, TE_POS); // raising triggerededge
    SpcSetParam (0, SPC_MULTTI, 01); // no multiple recording
    SpcSetParam (0, SPC_INOUT0, 01); // channel 0 input
    SpcSetParam (0, SPC_INOUT1, 11); // channel 1 output
    SpcSetParam (0, SPC_110OHM0, 11); // channel 0 110 Ohm input impedance
    SpcSetParam (0, SPC_110OHM1, 01); // channel 1 high input impedance

    // ----- calculate and set output data -----
    for (int i = 0; i < 1024; i++)
        data1[i] = i;
    SpcSetData (0, 2, 0, 10241, (dataptr) data1); // write data after the channel direction is set

    // ----- start the board -----
    SpcSetParam (0, SPC_COMMAND, SPC_START);

    // ----- wait for status ready -----
    do
    {
        SpcGetParam (0, SPC_STATUS, &Status);
    }
    while (Status != SPC_READY);

    // ----- read data -----
    SpcGetData (0, 2, 0, 10241, (dataptr) data0);

    return 0;
}
```