



SPECTRUM

SYSTEMENTWICKLUNG MICROELECTRONIC GMBH

AN006 - Bus Transfer Speed Details/Setup/Figures

Revision	Date	Changes
V 1.0	6th of March 2007	First release of document
V 1.1	23th of July 2007	Added further test results for a RAID0 system
V 1.2	4th of August 2008	Added further test results and performance note on 64 bit OS.
V 1.3	24th of June 2009	Added continuous buffer mode

General Information

This application note should give you an overview on all details concerning the bus transfer speed of the Spectrum PCI and PCI-X cards. It also includes some bus speed figures measured in-house on different systems. This may give you an idea what transfer speed can be reached and which circumstances affect this transfer speed.

MBytes/s and MSamples/s and MHz

To avoid confusion we have to start with some general information on the units that are used throughout the world of PC based data acquisition.

On the one hand we have the sampling speed in MSamples/s (short form MS/s or MSa/s or MSPS). This figure stands for Million samples per second. A sampling speed of 10 MS/s is therefore 10,000,000 Samples/second. Besides this we have the unit of MHz. This is normally used for the analog bandwidth (or signal frequency on a generator card). Very occassionally it is also sometimes used for the sampling rate. On all Spectrum documents the sampling speed is always given in MS/s while any analog frequencies are given in MHz.

Taking the sampling rate into account is one part for the transfer speed. The second part is the width of the samples and the number of samples that are stored on one sampling clock edge. A 1 channel 8 bit card has 1 byte/sample. A 4 channel 14 bit card has 8 bytes/sample (4 channels x 2 bytes/sample). To calculate the transfer speed that will occur all values have to be multiplied:

$$[\text{Transfer Speed}] = [\text{Sampling Rate}] * [\text{Channels}] * [\text{Bytes/Sample}]$$

In the computer world all values are not given as Million Bytes but as Mega Bytes (MByte or MB). 1 Mega Byte is $1024 * 1024$ Bytes = 1048576 Bytes. It is these values based from the computer world (like bus transfer speed, on-board memory, block sizes) that are the basis of those used in the Spectrum documentation and test programs. As a result this means that one has to be a little careful when comparing measuring results and speeds. The following table shows a comparison of acquisition speed and resulting transfer throughput:

Channels	Resolution	Bytes/Sample	Sampling rate	Bytes/second	MByte/second
1	8 bit analog	1	100 MS/s	100,000,000	95.3 MB/s
4	8 bit analog	1	100 MS/s	400,000,000	381.5 MB/s
1	12 bit analog	2	100 MS/s	200,000,000	190.7 MB/s
32	16 bit analog	2	3 MS/s	192,000,000	183.1 MB/s
4	14 bit analog	2	40 MS/s	320,000,000	305.2 MB/s
16	digital	2	100 MS/s	200,000,000	190.7 MB/s
64	digital	8	10 MS/s	80,000,000	76.3 MB/s

Transfer speed between card and PC memory

For the M2i card series these figures can be obtained using the card control center. All tests have been done with a standard card of the current production. All test have been done under Windows XP 32 bit if not otherwise noted. The following driver versions have been used for the tests introduced with the different revisions of this application note:

AN006 Revision	V1.0 and V1.1	V1.2
Windows 32 bit kernel driver	V1.26 build 1410	V1.31 build 2109
Windows 32 bit library (dll)	V1.20 build 1414	V1.34 build 2126
Windows 64 bit kernel driver	not used	V1.31 build 2109
Windows 64 bit library (dll)	not used	V1.34 build 2126.

Test results on different systems

The card control center uses a special internal mode that allows testing of the maximum FIFO transfer speed even if the card isn't capable of running with this speed. Therefore only the PCI and the memory interface are running at full speed. The card function itself is not involved. Therefore the following results are maximum values that were reached on the system.

To have a reliable continuous transfer speed in real life one should calculate with not more than about 95% - 98% of these values. To get nearer to the maximum that is possible and to still have a reliable transfer, the on-board memory of the card can be extended to have a longer buffering time.

All debug outputs have been disabled. The operating system has been optimized for speed.

Motherboard	Chipset	CPU	Mem	M2i PCI/PCI-X PCI slot		M2i PCI/PCI-X 66 MHz PCI-X slot		M2i PCI Express PCI Express slot	
				Write	Read	Write	Read	Write	Read
Asus P5WDG2 WS Pro	Intel 975X	Pentium D 3.0 GHz	2 GByte	105 MB/s	108 MB/s	189 MB/s	194 MB/s	n.a.	n.a.
Asus K8V SE Deluxe	VIA K8T800	Athlon 64 3000+	1 GByte	76 MB/s	115 MB/s	n.a.	n.a.	n.a.	n.a.
Asus P5WDG2-WS	Intel 975X	Pentium 4 3.0 GHz	1 GByte	108 MB/s	110 MB/s	189 MB/s	194 MB/s	n.a.	n.a.
Asus P4GE-V	Intel 845GE	Pentium 4 2.0 GHz	1 GByte	115 MB/s	116 MB/s	n.a.	n.a.	n.a.	n.a.
Asus A7V133	VIA KT133A	Athlon 1.0 GHz	576 MByte	48 MB/s	92 MB/s	n.a.	n.a.	n.a.	n.a.
Supermicro P4SCT+II	Intel 875P	Pentium 4 3.0 GHz	1 GByte	117 MB/s	118 MB/s	180 MB/s	203 MB/s	n.a.	n.a.
Supermicro P4SCT+	Intel E7210	Pentium 4 3.0 GHz	1 GByte	117 MB/s	118 MB/s	180 MB/s	203 MB/s	n.a.	n.a.
Supermicro PDSGE	Intel 955X	Celeron 2.53 GHz	1 GByte	95 MB/s	98 MB/s	187 MB/s	192 MB/s	n.a.	n.a.
Asus P4PB-	Intel 865P	Pentium 4 2.6 GHz	1 GByte	118 MB/s	118 MB/s	n.a.	n.a.	n.a.	n.a.
Asus CUV4X-E	VIA Apollo Pro133A	Pentium 3 1.0 GHz	1 GByte	74 MB/s	92 MB/s	n.a.	n.a.	n.a.	n.a.
Gigabyte 6VTXE	VIA Apollo Pro133T	Pentium 3 1.0 GHz	256 MByte	70 MB/s	85 MB/s	n.a.	n.a.	n.a.	n.a.
Supermicro X5DPE-G2	Intel E7501	2x Xeon 2.4 GHz	1 GByte	n.a.	n.a.	222 MB/s	220 MB/s	n.a.	n.a.
Asus K8V SE Deluxe	VIA K8T800	Athlon 64 3000+	1 GByte	79 MB/s	117 MB/s	n.a.	n.a.	n.a.	n.a.
Asus M2N32 WS PRO	NVIDIA nForce 590 SLI	Athlon 64 3600+	1 GByte	108 MB/s	120 MB/s	40 MB/s	150 MB/s	n.a.	n.a.
Gigabyte GA-965P-DS3	Intel 965	Core2Duo E6600, 2.4 GHz	2 GByte	94 MB/s	99 MB/s	n.a.	n.a.	118 MB/s	130 MB/s
Asus P5K	Intel P35	Core2Duo E6750	1 GByte	104 MB/s	108 MB/s	n.a.	n.a.	127 MB/s	132 MB/s
Kontron 986LCD-M/ATXP	Intel 945GM	Core2Duo T7200, 2.0 GHz	2 GByte	103 MB/s	107 MB/s	n.a.	n.a.	n.a.	n.a.
Supermicro X7DBP-i	Intel 5000P	Xeon DualCore, 1.86 GHz	1 GByte	n.a.	n.a.	201 MB/s	208 MB/s	n.a.	n.a.
Supermicro X7DBE	Intel 5000P	2 x Xeon DualCore, 2.0 GHz	4 GByte	n.a.	n.a.	196 MB/s	203 MB/s	n.a.	n.a.
Protech PMB-B421	Intel 875P	Pentium 4, 2.8 GHz	1 GByte	105 MB/s	118 MB/s	142 MB/s	188 MB/s	n.a.	n.a.
Intel DQ35JO (32 Bit XP)	Intel Q35 Express	Core2Quad Q9450 2.66 GHz	8 GByte	99 MB/s	102 MB/s	n.a.	n.a.	126 MB/s	130 MB/s
Intel DQ35JO (64 Bit XP)	Intel Q35 Express	Core2Quad Q9450 2.66 GHz	8 GByte	94 MB/s	97 MB/s	n.a.	n.a.	120 MB/s	124 MB/s

These results did not take into account any online calculation or data display. As soon as one wants to do more than just transfer data, the CPU calculation power and the programming of the algorithm becomes the critical point. While simple monitoring and level check functions will be no problem on a current standard system floating point calculations or data reduction algorithms may not work at full speed.



Please note that due to the necessary dual address cycle on the bus for addressing the PC RAM above the 32 bit memory limit, the throughput on a 64 bit Windows compared to a 32 bit Windows is approximately 5% lower.



The above values have been measured without Continuous Memory enabled. For details how this mode can further increase the bus throughput, please see the according section later in this application note.

Available Memory for data storage on 32 Bit and 64 Bit systems

When streaming data to/from PC memory the limit is the available memory for data. Current 32 bit systems can access 4 GByte of data in total (if supported by the motherboard). But this memory has to be shared between the adapter area, the operating system, any background tasks and systems services and the program. As the 32 bit address range is in total 4 GByte a part of this address range is mapped to the devices like PCI cards, graphics adapter, USB ports and all the others. As especially PCI cards can request very large memory areas the system designers normally reserve about 512 MByte of address space for these adapters.

As long as the maximum memory is not installed this won't hurt as the adapter address space then is no longer located in the physical memory. But as soon as 4 GByte are installed on the system the physical memory and the adapter address space overlap. As a result there's only 3.5 GByte left for the operating system. A program itself will only see 2 GByte of the remaining address space as the operating system default gives each task not more than 2 GByte. This can be extended with some special boot options but still the operating system needs some memory for it's own handles and internal tables and will reserve at least another 512 MByte.

As a result our program won't get more than 3 GByte of room for data storage on a 32 bit system.

Going to 64 bit systems this limit is extended and need not be applied for future applications, where it is possible to access as much memory as the motherboard can handle (and one can afford). The operating system of course still reserves a part for its own use but all the rest is available for the program.



While Spectrum has drivers for 64 bit operating systems for all M2i cards most other manufacturers will not give you the choice as they don't support 64 bit systems. Be sure not to run into any memory limits if deciding for on another solution.

Hard disk streaming

Getting data into the PC memory allows online data calculation, data monitoring and if required extends the on-board memory by a few GByte. But as soon as the amount of data to be recorded or replayed exceeds the PC memory one has to take the hard disk(s) into account. Exceeding PC memory capacity can be a question of seconds as the amount of transferred data may grow rapidly. The following table gives a brief overview of the amount of data that a certain time of transfer will generate:

Transfer Speed	Time	Amount of Data	Transfer Speed	Time	Amount of Data
10 MByte/s	1 second	10 MByte	200 MByte/s	1 second	200 MByte
10 MByte/s	10 seconds	100 MByte	200 MByte/s	10 seconds	2 GByte
10 MByte/s	1 minute	600 MByte	200 MByte/s	1 minute	12 GByte
10 MByte/s	10 minutes	6 GByte	200 MByte/s	10 minutes	120 GByte
10 MByte/s	1 hour	36 GByte	200 MByte/s	1 hour	720 GByte

RAID Arrays

Redundant Arrays of Inexpensive Disks; (RAID). The word "redundant" might be a little misleading here, in fact RAID usefully combines multiple small, inexpensive disk drives into an array of disk drives that yields performance and data security benefits which can exceed that of a single large (more expensive) drive. This array of drives appears to the computer as a single logical storage unit or drive, but it must be noted that there is no gain in storage size in this arrangement, for example using two 200 GByte drives will yield 200 GByte! The key to increased performance under RAID is parallelism, where simultaneous access to multiple disks allows data to be written to or read from a RAID array faster than it would be possible with a single drive.

RAID is commonly available in configurations RAID 0, 1, 2, 3, 4, 5 or 10 (with more are being added over time). Here we will look closer at systems 0 and 1, both of which will work with just two drives, and represent the entry level system most applicable for a PC based instrumentation system.

RAID Level 0

At this level, data is split across drives by a process called "striping", resulting in higher data throughput onto disk. Since no redundant information is stored, performance is very good and can be expected to nearly double that of a single drive, but the failure of any disk in the array results in data loss.

RAID Level 1

Provides redundancy by writing all data to two (or more) drives in a "mirroring" process. As the data is identical on each drive, having a redundant drive has the advantage of always having a copy of the data safe. The performance of a level 1 array tends to be faster on reads and slower on writes compared to a single drive or Raid 0, but if either drive fails, no data is lost.

Nowadays there are two ways to undertake connection of two (or more) drives into a RAID system. One is the classical way by use of a controller card. There's a wide variety of controller cards on the market with prices ranging from about 50 Euros up to 1000 Euros. They differ in the supported RAID levels, the number and type of hard disks to connect, the cache memory and the intelligence of the controller. At the end best performance may only be reached by the more powerful (and expensive) controllers.

The other way is to use an on-board RAID controller that is already equipped on the motherboard. As we'll see when having a look at the measuring results this needn't to be a worse solution.

Both ways of connection suffer from one critical point and that's the driver support. Any operating system on the market needs a special driver to run these RAID arrays. While drivers for Windows 2000 and Windows XP are standard, any Linux system as well as Windows Vista or Windows 64 bit systems can be a big problem. When selecting a hard disk streaming system based on RAID be sure to have matching drivers for your desired operating system.

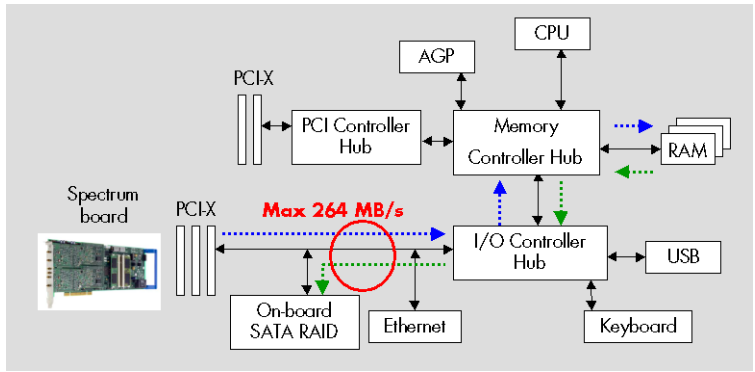
Software RAID

This is a speciality of the operating system. Instead of having a dedicated RAID controller the operating system itself uses two independent hard disks and forms a RAID system out of them. To get a RAID0 system one need to select „striping“. The software RAID feature is already build-in for Windows XP and Vista and also available for Linux systems. Setting up a software RAID under Windows is only a question of 2 minutes.

Connecting the hard disks

One thing that has to be kept in mind is the bus structure of the PCI and PCI-X buses. The maximum bus throughput is shared between all components that are located on this bus. Therefore a little care has to be taken when selecting the right motherboard and especially when using a dedicated RAID controller. A wrong system setup can greatly slow down the overall throughput. The following drawings will show some different system setups and explain what to check.

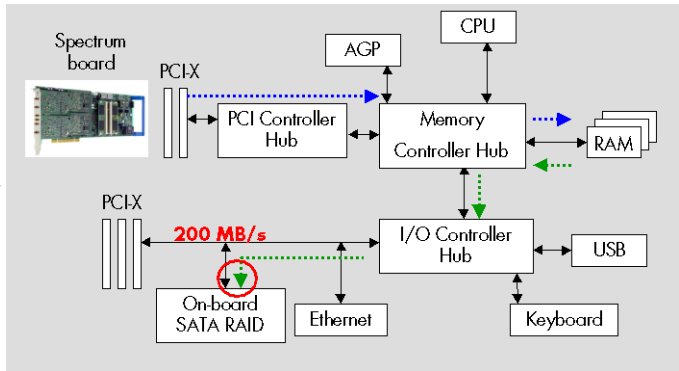
The right hand drawing shows a typical server motherboard layout. There are in total 5 PCI-X slots which are grouped in two bus segments. The upper Controller Hub that is directly connected to the Memory Controller Hub is intended for high performance cards and is able to run with up to 100 MHz while the second PCI-X bus is connected to the I/O controller and is intended for 66 MHz maximum.



The bottleneck here is the connection of the on-board SATA controller directly to the second PCI-X bus. The acquisition card and the SATA controller share the same bus segment and as data has to be transferred twice across this bus segment the overall throughput is only half of the maximum. In theory the PCI-X bus can transfer 264 MB/s when running with 32 bit and 66 MHz like the Spectrum cards do. In this setup there is only a maximum of 132 MB/s in each direction remaining. Even when using the newest components and the fastest hard disks on the market this setup won't stream with more than maybe 100 MB/s.

The worst scenario would be if the system hard disk is also connected through this controller and also the system hard disk access goes through this bottleneck.

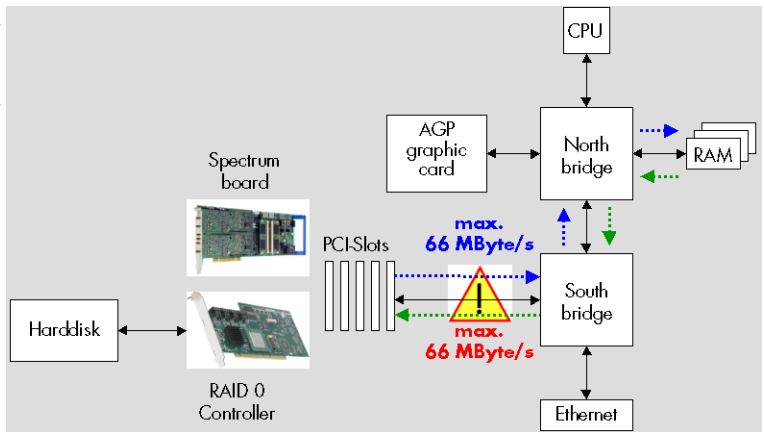
Simply changing the position of the Spectrum card as shown on the right hand drawing will enhance the system throughput. The SATA RAID controller can now run with full speed and none of the components have to share the bus segments.



Even the interconnection between the two motherboard controllers (north and south bridge) is now only occupied in one direction getting the best possible performance out of the system.

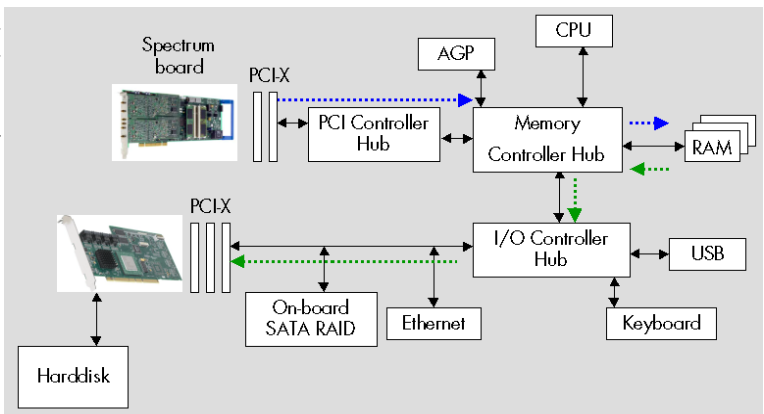
As long as the RAID controller is powerful and the hard disks fast enough this system setup can reach the 200 MB/s throughput with hard disk streaming.

This is an extreme example of wasting money. A dedicated RAID controller is plugged into a standard PCI bus where also the Spectrum card is located. As the legacy PCI bus can only transfer a maximum of 132 MB/s in total there is only 66 MByte/s for each direction left. In real life this setup will not reach more than 50 MB/s of hard disk streaming speed



As we will see later, a good single SATA hard disk can reach this streaming speed without any additional components. The RAID controller is completely useless here for increasing the streaming speed.

In here we again use our RAID controller from the last example but now it is plugged into a more efficient bus segment that it occupies without sharing.



In this setup the performance to reach is again only limited by the maximum throughput of the Spectrum card and the power of the combination of RAID controller and hard disks.

Hard disk streaming test

The test results differ depending on the card setup. The buffer sizes and the notify sizes define the maximum throughput that is possible to reach. When doing acquisition using the largest buffer size possible it results in no disadvantages. The software buffer is simply limited by the installed PC memory while the HW buffer is limited by the installed on-board memory of the card. When doing data generation using large buffer sizes a larger latency will be applied to the changing the outputs on-the-fly. Please see the hardware manual of the generator cards for more details.

The notify size should be programmed in the region of a few MBytes to reach maximum throughput. Defining a very small notify size generates a huge amount of interrupts and only has the advantage of having a very fine granularity. Normally this is not required for hard disk streaming. File operations are also most efficient when using larger blocks.

For the tests a M2i.7020 (32 bit digital i/o card with 125 MS/s) was used. This card was just taken from production series and not optimized in any way.

Reference system 1

The reference test system is based on a good workstation motherboard combined with an on-board RAID controller and 4 very fast SATA hard disks. The system is based on reliable state-of-the-art components and has a price of about 1,400 €.

Motherboard	Asus P5WDG2 WS Pro	RAID0 controller	on-board Intel ICH7R
CPU	Intel Pentium D 3 GHz	Chip set	Intel 975X
Hyperthreading	enabled	Operating System	Windows XP Professional SP2
Memory	DDR2 2 GByte	OS Setup	Optimized for speed
Hard Disks	4 x Seagate Barracuda 7200.10	Test card	M2i.7010 (16 Bit Digital I/O)

Testing the plain hard disk read and write speed with a self programmed C++ tool one can see the difference in speed depending on the selected block size that is written/read with one single system call to/from hard disk.

Block Size	4 disk RAID 0 strip size 16 kByte		2 disk RAID 0 strip size 16 kByte		2 disk software RAID (striped, same channel)		2 disk software RAID (striped, different channels)		single disk	
	Write	Read	Write	Read	Write	Read	Write	Read	Write	Read
64 kByte	132 MByte/s	85 MByte/s	55 MByte/s	85 MByte/s	62 MByte/s	85 MByte/s	80 MByte/s	105 MByte/s	67 MByte/s	70 MByte/s
128 kByte	160 MByte/s	195 MByte/s	100 MByte/s	135 MByte/s	85 MByte/s	95 MByte/s	120 MByte/s	135 MByte/s	67 MByte/s	70 MByte/s
256 kByte	180 MByte/s	195 MByte/s	130 MByte/s	135 MByte/s	90 MByte/s	102 MByte/s	125 MByte/s	138 MByte/s	67 MByte/s	69 MByte/s
512 kByte	195 MByte/s	225 MByte/s	132 MByte/s	137 MByte/s	95 MByte/s	105 MByte/s	126 MByte/s	138 MByte/s	67 MByte/s	69 MByte/s
1 MByte	195 MByte/s	230 MByte/s	132 MByte/s	137 MByte/s	95 MByte/s	107 MByte/s	134 MByte/s	138 MByte/s	67 MByte/s	69 MByte/s
2 MByte	205 MByte/s	230 MByte/s	131 MByte/s	138 MByte/s	96 MByte/s	107 MByte/s	134 MByte/s	139 MByte/s	67 MByte/s	69 MByte/s
4 MByte	205 MByte/s	230 MByte/s	133 MByte/s	138 MByte/s	97 MByte/s	108 MByte/s	135 MByte/s	139 MByte/s	67 MByte/s	69 MByte/s

It is obvious that there is no big difference between a software RAID and an on-board hardware RAID in this setup. The software RAID has the big advantage that no extra RAID driver has to be installed in the system. In theory there must be a little more CPU usage in software RAID but this can't be seen on the reference system. The CPU usage is every time less than 5%.

The connection of the hard disks heavily influences the software RAID. If both hard disks are connected to the same SATA channel the performance is much less than if connected to different SATA channels. An estimation would be that a 4 disk software RAID doesn't get the same performance as a 4 disk hardware RAID as there are only 2 SATA channels in the system present.

Test results reference system 1

The programmed notify size is also used as a block size for file write and read access. The transferred file was at least 100 GByte of size

Direction	SW Buffer Size	HW Buffer Size	Notify Size	Hard Disk set	Transfer Speed
Acquisition -> Hard Disk	512 MByte	64 MByte	8 MByte	4 hard disks RAID0	190 MByte/s
Acquisition -> Hard Disk	512 MByte	64 MByte	1 MByte	4 hard disks RAID0	95 MByte/s
Acquisition -> Hard Disk	64 MByte	64 MByte	8 MByte	4 hard disks RAID0	185 MByte/s
Hard Disk -> Output	512 MByte	64 MByte	1 MByte	4 hard disks RAID0	180 MByte/s
Hard Disk -> Output	16 MByte	16 MByte	64 kByte	4 hard disks RAID0	135 MByte/s
Hard Disk -> Output	16 MByte	16 MByte	4 kByte	4 hard disks RAID0	12 MByte/s
Acquisition -> Hard Disk	512 MByte	64 MByte	8 MByte	2 hard disks RAID0	125 MByte/s
Acquisition -> Hard Disk	512 MByte	64 MByte	1 MByte	2 hard disks RAID0	80 MByte/s
Hard Disk -> Output	512 MByte	64 MByte	1 MByte	2 hard disks RAID0	135 MByte/s
Hard Disk -> Output	16 MByte	16 MByte	64 kByte	2 hard disks RAID0	70 MByte/s
Acquisition -> Hard Disk	512 MByte	64 MByte	8 MByte	2 disk software RAID (same SATA channel)	85 MByte/s
Hard Disk -> Output	512 MByte	64 MByte	1 MByte	2 disk software RAID (same SATA channel)	101 MByte/s
Acquisition -> Hard Disk	512 MByte	64 MByte	8 MByte	2 disk software RAID (different SATA channels)	130 MByte/s
Hard Disk -> Output	512 MByte	64 MByte	1 MByte	2 disk software RAID (different SATA channels)	130 MByte/s
Acquisition -> Hard Disk	512 MByte	64 MByte	8 MByte	single data disk	60 MByte/s
Hard Disk -> Output	512 MByte	64 MByte	1 MByte	single data disk	63 MByte/s

Reference system 2

The second reference test system is based on an older server motherboard combined with a dedicated RAID controller and 4 fast SATA hard disks. All components are between 2 and 3 years old. To see the difference to modern hard disk this system was tested with two different sets of hard disks.

Motherboard	Supermicro X5DPE-G2	Operating System	Windows XP Professional SP2
CPU	Xeon Dual 2.4 GHz	OS Setup	Optimized for speed
Chip set	Intel E7501	RAID0 controller	3Ware Escalade 9000
Memory	1 GByte	Hard Disks (test set 1)	4 x Samsung SP1614C
Test card	M2i.7010 (16 Bit Digital I/O)	Hard Disks (test set 2)	4 x Seagate Barracuda 7200.10

Testing the plain hard disk read and write speed with a self programmed C++ tool one can see the difference in speed depending on the selected block size that is written/read with one single system call to/from hard disk and the programmed strip size of the RAID array.

Block Size	4 disk RAID 0 (Samsung) strip size 16 kByte		4 disk RAID 0 (Seagate) strip size 16 kByte		4 disk RAID 0 (Seagate) strip size 256 kByte	
	Write	Read	Write	Read	Write	Read
64 kByte	125 MByte/s	100 MByte/s	125 MByte/s	130 MByte/s	150 MByte/s	50 MByte/s
128 kByte	140 MByte/s	130 MByte/s	138 MByte/s	140 MByte/s	170 MByte/s	53 MByte/s
256 kByte	150 MByte/s	155 MByte/s	150 MByte/s	175 MByte/s	180 MByte/s	60 MByte/s
512 kByte	160 MByte/s	190 MByte/s	160 MByte/s	200 MByte/s	195 MByte/s	120 MByte/s
1 MByte	165 MByte/s	200 MByte/s	168 MByte/s	210 MByte/s	200 MByte/s	215 MByte/s
2 MByte	167 MByte/s	210 MByte/s	168 MByte/s	220 MByte/s	200 MByte/s	220 MByte/s
4 MByte	167 MByte/s	210 MByte/s	168 MByte/s	220 MByte/s	200 MByte/s	240 MByte/s

The reached transfer speed is more or less independent of the used hard disks although the Seagate hard disks are state-of-the-art and the Samsung ones are quite old. The selected strip size influences the maximum transfer to/from hard disk speed that can be reached. Especially when doing hard disk write operations the strip size need to be programmed quite large.

Test results reference system 2

The programmed notify size is also used as a block size for file write and read access.

Direction	SW Buffer Size	HW Buffer Size	Notify Size	Hard Disk Set	Strip Size	Transfer Speed
Acquisition -> Hard Disk	256 MByte	64 MByte	8 MByte	Set 1 (Samsung)	16 kByte	135 MByte/s
Acquisition -> Hard Disk	256 MByte	64 MByte	32 MByte	Set 1 (Samsung)	16 kByte	155 MByte/s
Hard Disk -> Output	256 MByte	64 MByte	64 kByte	Set 1 (Samsung)	16 kByte	80 MByte/s
Hard Disk -> Output	256 MByte	64 MByte	256 kByte	Set 1 (Samsung)	16 kByte	160 MByte/s
Hard Disk -> Output	256 MByte	64 MByte	1 MByte	Set 1 (Samsung)	16 kByte	200 MByte/s
Hard Disk -> Output	256 MByte	64 MByte	8 MByte	Set 1 (Samsung)	16 kByte	210 MByte/s
Acquisition -> Hard Disk	256 MByte	64 MByte	8 MByte	Set 2 (Seagate)	16 kByte	140 MByte/s
Acquisition -> Hard Disk	256 MByte	64 MByte	32 MByte	Set 2 (Seagate)	16 kByte	155 MByte/s
Hard Disk -> Output	256 MByte	64 MByte	1 MByte	Set 2 (Seagate)	16 kByte	210 MByte/s
Hard Disk -> Output	256 MByte	64 MByte	8 MByte	Set 2 (Seagate)	16 kByte	220 MByte/s
Acquisition -> Hard Disk	256 MByte	64 MByte	8 MByte	Set 2 (Seagate)	256 kByte	160 MByte/s
Acquisition -> Hard Disk	256 MByte	64 MByte	32 MByte	Set 2 (Seagate)	256 kByte	185 MByte/s
Hard Disk -> Output	256 MByte	64 MByte	1 MByte	Set 2 (Seagate)	256 kByte	200 MByte/s
Hard Disk -> Output	256 MByte	64 MByte	8 MByte	Set 2 (Seagate)	256 kByte	220 MByte/s

Reference system 3

The third reference test system is based on an state-of-the-art (mid 2007) Supermicro server system with lots of processign power and a very fast PCI Express RAID controller

Motherboard	Supermicro X7 DBE	Operating System	Windows XP Professional SP2
CPU	2 x Xeon DualCore 2.0 GHz	OS Setup	Optimized for speed
Chip set	Intel 5000P	RAID0 controller	PCI Express x8 Areca 1220
Memory	4 GByte	RAID controller on-board cache	245 MByte
Test card	M2i.7010 (16 Bit Digital I/O)	Hard Disks (test set 1)	6 x Seagate ST3500630AS

Testing the plain hard disk read and write speed with a self programmed C++ tool one can see the difference in speed depending on the selected block size that is written/read with one single system call to/from hard disk. In this setup the strip size has only a minor impact for smaller block sizes and is therefore not separately listed.

Block Size	2 disk RAID 0 strip size 64 kByte		4 disk RAID 0 strip size 64 kByte		6 disk RAID 0 strip size 64 kByte	
	Write	Read	Write	Read	Write	Read
64 kByte	139 MByte/s	128 MByte/s	225 MByte/s	259 MByte/s	247 MByte/s	380 MByte/s
128 kByte	138 MByte/s	124 MByte/s	278 MByte/s	269 MByte/s	277 MByte/s	380 MByte/s
256 kByte	138 MByte/s	133 MByte/s	280 MByte/s	278 MByte/s	366 MByte/s	375 MByte/s
512 kByte	133 MByte/s	138 MByte/s	295 MByte/s	283 MByte/s	381 MByte/s	367 MByte/s
1 MByte	132 MByte/s	139 MByte/s	290 MByte/s	289 MByte/s	391 MByte/s	359 MByte/s
2 MByte	132 MByte/s	139 MByte/s	281 MByte/s	290 MByte/s	391 MByte/s	366 MByte/s
4 MByte	132 MByte/s	139 MByte/s	281 MByte/s	290 MByte/s	391 MByte/s	366 MByte/s

Test results reference system 3

Direction	Cards	SW Buffer Size	HW Buffer Size	Notify Size	Strip Size	Transfer Speed
Acquisition -> Hard Disk	1 card at 16 bit, 105 MS/s	1024 MByte	64 MByte	8 MByte	64 kByte	200 MByte/s
Acquisition -> Hard Disk	2 cards at 16 bit, 90 MS/s	1024 MByte	64 MByte	8 MByte	64 kByte	345 MByte/s
Acquisition -> Hard Disk	3 cards at 16 bit, 52 MS/s	1024 MByte	64 MByte	8 MByte	64 kByte	300 MByte/s

Continuous memory for increased data transfer rate



The continuous memory buffer has been added to the driver version 1.36. The continuous buffer is not available in older driver versions. Please update to the latest driver if you wish to use this function.

Background

All modern operating systems use a very complex memory management strategy that strictly separates between physical memory, kernel memory and user memory. The memory management is based on memory pages (normally 4 kByte = 4096 Bytes). All software only sees virtual memory that is translated into physical memory addresses by a memory management unit based on the mentioned pages.

This will lead to the circumstance that although a user program allocated a larger memory block (as an example 1 MByte) and it sees the whole 1 MByte as a virtually continuous memory area this memory is physically located as spread 4 kByte pages all over the physical memory. No problem for the user program as the memory management unit will simply translate the virtual continuous addresses to the physically spread pages totally transparent for the user program.

When using this virtual memory for a DMA transfer things become more complicated. The DMA engine of any hardware can only access physical addresses. As a result the DMA engine has to access each 4 kByte page separately. This is done through the Scatter-Gather list. This list is simply a linked list of the physical page addresses which represent the user buffer. All translation and set-up of the Scatter-Gather list is done inside the driver without being seen by the user. Although the Scatter-Gather DMA transfer is an advanced and powerful technology it has one disadvantage: For each transferred memory page of data it is necessary to also load one Scatter-Gather entry (which is 16 bytes on 32 bit systems and 32 bytes on 64 bit systems). The little overhead to transfer (16/32 bytes in relation to 4096 bytes, being less than one percent) isn't critical but the fact that the continuous data transfer on the bus is broken up every 4096 bytes and some different addresses have to be accessed slow things down.

The solution is very simple: everything works faster if the user buffer is not only virtually continuous but also physically continuous. Unfortunately it is not possible to get a physical continuous buffer for a user program. Therefore the kernel driver has to do the job and the user program simply has to read out the address and the length of this continuous buffer. This is done with the function `spcm_dwGetContBuf` as already mentioned in the general driver description. The desired length of the continuous buffer has to be programmed to the kernel driver for load time and is done different on the different operating systems. Please see the following chapters for more details.

Next we'll see some measuring results of the data transfer rate with/without continuous buffer. You will find more results on different motherboards and systems in the application note number 6 „Bus Transfer Speed Details“

Bus Transfer Speed Details (example system)

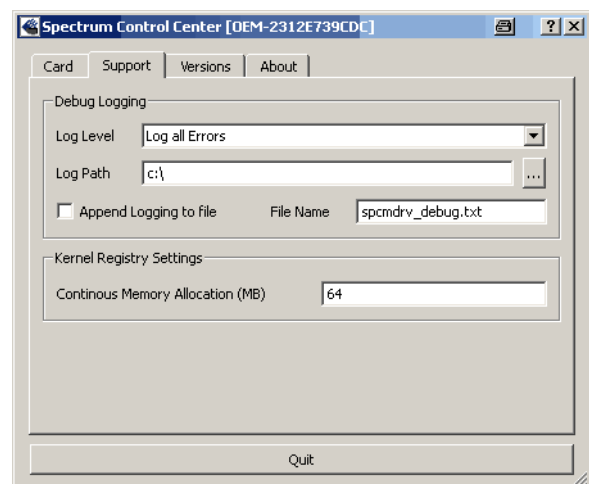
Mode	PCI 33 MHz slot		PCI-X 66 MHz slot		PCI Express x1 slot	
	read	write	read	write	read	write
User buffer	109 MB/s	107 MB/s	195 MB/s	190 MB/s	130 MB/s	138 MB/s
Continuous kernel buffer	125 MB/s	122 MB/s	248 MB/s	238 MB/s	160 MB/s	170 MB/s
Speed advantage	15%	14%	27%	25%	24%	23%

Setup on Windows systems

The continuous buffer settings is done with the Spectrum Control Center using a setup located on the „Support“ page. Please fill in the desired continuous buffer settings as MByte. After setting up the value the system needs to be restarted as the allocation of the buffer is done during system boot time.

If the system cannot allocate the amount of memory it will divide the desired memory by two and try again. This will continue until the system can allocate a continuous buffer. Please note that this try and error routine will need several seconds for each failed allocation try during boot up procedure. During these tries the system will look like being crashed. It is then recommended to change the buffer settings to a smaller value to avoid the long waiting time during boot up.

Continuous buffer settings should not exceed 1/4 of system memory. During tests the maximum amount that could be allocated was 384 MByte of continuous buffer on a system with 4 GByte memory installed.



Setup on Linux systems

On Linux systems the continuous buffer setting is done via the command line argument `contmem_mb` when loading the kernel driver module:

```
insmod spcm.ko contmem_mb=4
```

As memory allocation is organized completely different compared to Windows the amount of data that is available for a continuous DMA buffer is unfortunately limited to a maximum of 8 MByte. On most systems it will even be only 4 MBytes.

Usage of the buffer

The usage of the continuous memory is very simple. It is just necessary to read the start address of the continuous memory from the driver and use this address instead of a self allocated user buffer for data transfer.

Function `spcm_dwGetContBuf`

This function reads out the internal continuous memory buffer if one has been allocated. If no buffer has been allocated the function returns a size of zero and a NULL pointer.

```
uint32_stdcall spcm_dwGetContBuf_i64 ( // Return value is an error code
    drv_handle hDevice,           // handle to an already opened device
    uint32     dwBufType,         // type of the buffer to read as listed above under SPCM_BUF_XXXX
    void**     ppvDataBuffer,     // address of available data buffer
    uint64*    pqwContBufLen);    // length of available continuous buffer

uint32_stdcall spcm_dwGetContBuf_i64m ( // Return value is an error code
    drv_handle hDevice,           // handle to an already opened device
    uint32     dwBufType,         // type of the buffer to read as listed above under SPCM_BUF_XXXX
    void**     ppvDataBuffer,     // address of available data buffer
    uint32*    pdwContBufLenH,    // high part of length of available continuous buffer
    uint32*    pdwContBufLenL);    // low part of length of available continuous buffer
```

Please note that it is not possible to free the continuous memory for the user application.

Example

The following example shows a simple standard single mode data acquisition setup with the read out of data afterwards. To keep this example simple there is no error checking implemented.

```
int32 lMemsize = 16384; // recording length is set to 16 kSamples

spcm_dwSetParam_i32 (hDrv, SPC_CHENABLE, CHANNEL0); // only one channel activated
spcm_dwSetParam_i32 (hDrv, SPC_CARDMODE, SPC_REC_STD_SINGLE); // set the standard single recording mode
spcm_dwSetParam_i32 (hDrv, SPC_MEMSIZE, lMemsize); // recording length
spcm_dwSetParam_i32 (hDrv, SPC_POSTTRIGGER, 8192); // samples to acquire after trigger = 8k

// now we start the acquisition and wait for the interrupt that signalizes the end
spcm_dwSetParam_i32 (hDrv, SPC_M2CMD, M2CMD_CARD_START | M2CMD_CARD_ENABLETRIGGER | M2CMD_CARD_WAITREADY);

// we now try to use a continuous buffer for data transfer or allocate our own buffer in case there's none
spcm_dwGetContBuf_i64 (hDrv, SPCM_BUF_DATA, &pvData, &qwContBufLen);
if (qwContBufLen < (2 * lMemsize))
    pvData = new int16[lMemsize];

// read out the data
spcm_dwDefTransfer_i64 (hDrv, SPCM_BUF_DATA, SPCM_DIR_CARDTOPC, 0, pvData, 0, 2 * lMemsize);
spcm_dwSetParam_i32 (hDrv, SPC_M2CMD, M2CMD_DATA_STARTDMA | M2CMD_DATA_WAITDMA);

if (qwContBufLen < (2 * lMemsize))
    delete[] pvData;
```

Hints and Summary

Much is possible when using the correct components and a planned setup when doing system integration focused on high throughput speed. On the other hand one can see that when components are not that powerful (like the VIA chipset based motherboards) they will not allow the full performance that one expects.

As a summary we want to give some hints to be taken into account for high-speed system setup:

- On data acquisition systems the notify size should be selected quite large (a few MByte) to get a stable and fast throughput
- The stripe size for a hard disk RAID system needs to be selected depending on the intended main data throughput direction. When doing

- acquisition (card to hard disk) the stripe size should be large, when doing generation (hard disk to card) the stripe size can be small
- When setting up a RAID system the money is better spent on fast state-of-the-art hard disks than into a dedicated RAID controller
 - When doing software RAID it is absolutely necessary to put all disks on separate SATA channels to get best performance
 - When using a dedicated RAID controller the DAQ/generator card must be placed in a different bus segment than the RAID controller
 - Please be sure to use a driver for M2i card series that is at least version 1.18b1355 and for MI/MC/MX card series at least version 3.19b1340 to get full data transfer performance from PC to card. Older drivers have transfer problems on PCI-X slots and may not reach full throughput for output
 - For fast hard disk streaming it is recommended to turn off operating system caching for that device (for C++ FILE_FLAG_NO_BUFFERING for the CreateFile function). Caching only makes sense for random access
 - One thing to keep in mind when using RAID 0 systems is that the danger of failure and data loss will be n-times higher where n is the number of used hard disks. As RAID 0 is only striping data and does no mirroring any hardware failure will corrupt the complete data.
 - Enable continuous kernel buffer mode to get the maximum possible transfer speed on the card's bus segment.

All given figures in this document are just example of speeds which can be reached. As Windows is not a real-time operating system none of the above figures can be guaranteed for any system setup. Software, system installation and all other hardware components may also influence the transfer speed that can be reached.

