RAID

Definition, usage and performance considerations

Eurologic Systems Ltd
Maple House
South County Business Park
Leopardstown
Dublin 18
Ireland

This document is preliminary and subject to change without notice. Please contact Eurologic Documentation Control for Revision Verification.

Revision History

Version	Date	Author	Details	Reviewed By
1.0	17 Dec 2001	Marijke Keet	First document revision	C&TS and RAID team
1.1	19 Dec 2001	Marijke Keet	More performance/ stripe size details	Mylex

Table of Contents

1. Introduction

1.1 Scope

The purpose of this document is to provide the reader with a (brief) overview of hardware RAID levels, their performance and scenarios for use in real live systems. The conclusions list some shortcuts for advising "best" RAID levels
The intended reader is defined as Customer Engineer, Engineering Support and Systems Administrators/Engineers out in the field.

The conceptual difference between hardware and software RAID implementations is given in *Appendix A* as a reference and *Appendix B* lists best practices for the Qlogic HBA and SANbloc controller settings. Please refer to the reference documentation or contact Eurologic Systems Ltd for training or more detail on the topics covered in this document.

1.2 Reference documentation

- Berkeley RAID Standards
- Eurologic Systems – The *RAID*book, 5th edition
- Eurologic Systems – RAID Stratus2.ppt doc
- Mylex Encyclopaedia of Controller Fundamentals
- Mylex Performance Optimization 000312
- Mylex FFx-2 White Paper
- Compaq Computer Corporation training material – internal document
- A "Google-search" on RAID returns a virtually endless list of links to white papers, implementations and even more performance & usage considerations with all combinations of hardware and software RAID levels.

2. RAID levels

As a quick reminder, the (existing) RAID levels are briefly addressed. The RAID 0+1 and RAID 30 and 50 sections are likely of most interest.

2.1 RAID 0 – striping

RAID 0 is the "simplest" of all: it divides the data among the disks in equally sized chunks (see Table 1). This boosts the performance because of the reduced data retrieval time (more disks with data can be accessed at the same time), but does NOT provide you with any redundancy. If one disk fails, you lost your data.

Drive1	Drive2	Drive3	Drive4	Drive5
A1	A2	A3	A4	A5
B1	B2	B3	B4	B5
C1	C2	…	…	…
…	…	…	…	Z5

Table 1. RAID 0

The minimum amount of disks is 2 when the SANBloc 2500 is used.
(When only 1 disk is selected for an array, you will have the option to set it up as a JBOD, Just a Bunch Of Disks, if the SANbloc FC2500 is used.)

2.2 RAID 1 – mirroring

Here you have your maximal data redundancy. Every piece of information is written to disk and copied to a second disk, which then actually "mirrors" the data on the first disk.

Drive1	Drive2
A	A'
B	B'
C	C'
…	…

Table 2. RAID 1

As you can see in *Table 2*, you will loose one disk, which otherwise could have been used for extra data storage, to achieve this maximum redundancy.

2.3 RAID 2

Not commercially implemented.

2.4 RAID 3 – striping with parity

With this option you will have so-called parity, which is a nice algorithm doing calculations on your stored bits.
Taking the example of 5 disks in an array (one column each) as shown in *Table 3*, four "pieces" of data are written to disk, and the controller will calculate the extra number (the A'; that last drive is called the parity drive). If one of the first four disks fails, the controller will use the data bits stored in the other three drives, combines it with the related chunk of the parity drive and then calculates the data that used to be on the now defective drive. This "missing" information is then written on to a newly inserted drive.

Drive1	Drive2	Drive3	Drive4	Drive5
A1	A2	A3	A4	A'
B1	B2	B3	B4	B'
C1	C2	…	…	C'
…	…			

Table 3. RAID 3

This looks kind of nice: you have you striping and your redundancy.
See the Usage paragraph for reasons why some users would want to implement this RAID level.

2.5 RAID 4

Very rarely used nowadays, and not implemented with the SANbloc controller firmware.

2.6 RAID 5 – striping with parity

There were more people who acknowledged the problem mentioned in the RAID3 section, so they came up with something better: RAID 5. RAID5 addresses both the RAID 0 and the RAID 3 limitations and to a certain extend the RAID 1 one.

- It provides performance advantage of striping,
- Has the redundancy that RAID 0 doesn't have,
- Uses less disk space than RAID 1 to do that and

The table presents you the situation where 5 disks are in the array and the "'" indicates the parity information. Here it does not matter which drive fails, you will always be able to access your data, as long as there is only 1 failed disk, and revert back to the original healthy state when a new disk (or Hot Spare) is added.

Drive1	Drive2	Drive3	Drive4	Drive5
A1	A2	A3	A4	A’
B’	B1	B2	B3	B4
C1	C’	C2	C3	C4
D1	D2	D’	D3	D4
E1	E2	E3	E’	E4
…	…	…		

Table 4. RAID 5

2.7 RAID 0+1/10

RAID level 0+1 and/or level 10 can trigger a lovely and lively endless discussion among RAID experts about the exact naming, 0+1 or 10, and the exact implementation. There is a so-called Berkeley Standard, but for example Mylex has a different implementation of the “0+1”, adding up to the discussion. The following two paragraphs provide you the clarifications.

2.7.1 The SANbloc FC2500 controller 0+1

Though there are “standards” on how to implement the different RAID levels, there is no obligation to use that specific description of algorithms. The SANbloc FC2500 and FC2502 are equipped with the Mylex FFx and FFx2 controllers respectively, which do **not** use the default way of implementing RAID 0+1.
Memorizing the previous paragraphs about RAID 0, striping, and RAID 1, mirroring, RAID 0+1 is a combination of the two. The data chunks are first striped and then mirrored. Mirrored in a sense that each data chunk is written twice to two different disks (see *Table 5*).

Drive1	Drive2	Drive3	Drive4	Drive5
A1	A2	A3	A4	A5
B1	A1’	A2’	A3’	A4’
A5’	B2	B3	B4	B5
…	…	…	…	…

Table 5. RAID 1+0

The advantage of this implementation is that you can use an odd number of disks. E.g. you have a RAID 1 (and not your OS installed on it), and need to increase the size of the LUN (logical drive) with one drive and just want make more effective use of the physical disk space. Doing a MORE expansion operation adding one drive to a RAID1 os possible with the Mylex implementation of RAID 0+1, but not possible with the Berkeley standard.

2.7.2 The Berkeley standard 0+1

With the SANbloc controller RAID 0+1 (§2.7.1), the calculation is first to stripe, then to mirror the stripe. The Berkeley standard first mirrors, and then does the striping. The difference is shown in *Table 6*.

Drive1	Drive2	Drive3	Drive4
A1	A1'	A2	A2'
B1	B1'	B2	B2'
…	…	…	
			…

Table 6. RAID 10

Like everything, there are pros and cons. A pro is, that in this situation it is possible to loose 2 disks and still be able to retain your data. The negative aspect is that you always need an even number of disks.

2.8 RAID 30 and 50

RAID 30 and 50 are treated here as the same, because the idea is the same, ignoring the different location of the parity. The essence is, that an x amount of RAID 5 or RAID 3 arrays (also called disk groups) are clustered together to form one big entity. The data then is striped over the different arrays, which themselves stripe that information and calculate the parity.
Roughly, the disks will look like this in a RAID 50:

Drive1	Drive2	Drive3
A1	A2	A'
B'	B1	B2
C1	C'	C4

Drive1	Drive2	Drive3
A3	A4	A'
B'	B3	B4
C3	C'	C4

Drive1	Drive2	Drive3
A5	A6	A'
B'	B5	B6
C5	C'	C6

Seen as one very large drive by the OS

So, from a logical drive point of view, 3 drives (each from 1 array) can fail, and still you would retain all data.

Controllers from different manufacturers have found two ways for the user to select the use of a RAID 30 or 50.

1. When selecting the RAID level for the logical drive (LUN) there is an option that says “RAID 30”, or 50.
2. Select the check box “spanning” and create arrays of an equal amount of disks and size and the controller (firmware) will perform the RAID 30 or 50 automatically when RAID 5 or 3 is selected upon LUN creation.

---

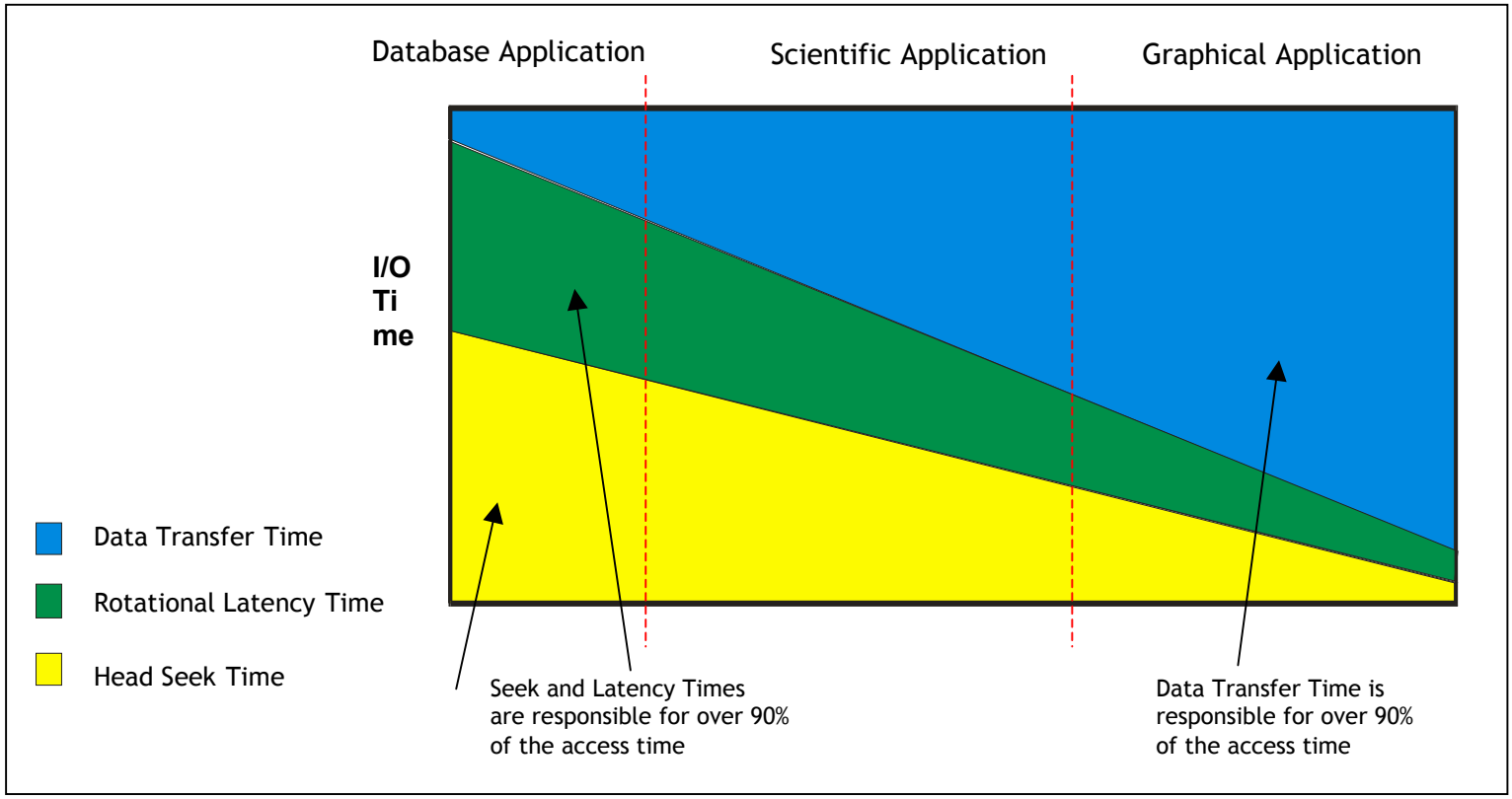
3. Performance considerations

3.1 Stripe size

Besides the RAID levels and the intended usage, the so-called stripe size is important for achieving optimal performance. The stripe size means the chunk of data that will be written to the disks in one go.

1. If your files or application were a database for example, you would want to define a small stripe size: In most situations, a cell/record of a database contains a small piece of information (like a few numbers) that will be written to/read from disk in one request. Reserving for example 64K disk space for each write (= saving a record) is way too much, because a cell normally does not contain that many data, but the remainder of the reserved 64K (50 to 60K) will not be used. In such situations it is advantageous to set the stripe size of the RAID to it's minimum of 8K to prevent loss of disk space. This generally will also improve performance, because only a small chunk has to be read by the controller.
2. The opposite is true for larger files (like on a file server) or video streaming. To optimise performance, you ideally want to retrieve (or write) as much data as possible within one send or request. In other words: maximize the stripe size, which is 64K for the SANbloc controller.

Translating the above text into a graph of I/O performance as a function of the used application:



Ref. Proceedings of the IEEE, Vol 77, No 12 December 1989

Note: Independent/third party results indicate that no *significant* difference in performance, relative to cache stripe size has been observed with the SANBloc FC2500 controller. However, this is left up for the user to verify as performance results are dependent on multiple parameters of a particular configuration.

3.2 RAID implementation differences

3.2.1 Software RAID

If you look ate the picture in Appendix A that outlines the data and communication paths needed for software RAID implementation, it will become clear that it is not highly effective. The process consumes bus bandwidth, claims memory space and requires extra CPU time, that otherwise could have been used for other processes. Therefore, per definition ***all software RAID implementations are slower than their hardware counterpart***.

3.2.2 Hardware RAID

Hardware RAID can be implemented internally, i.e. with an array controller residing in the server, or externally, where the controller is located in the storage enclosure.
Everybody has their own preferences, requirements and prior knowledge, hence might choose for internal RAID controllers.
However, external controllers provide more flexibility. Internal controllers you could use with direct attach storage (called DAS or SAS), but are not advisable for network attached storage (NAS) or Storage Attached Networks (SANs) due to configuration restrictions. Secondly, if the server goes down, the controller goes with it. Third, when you configure the storage, you must do that with the server *and* storage together.
External controllers overcome this situation. It is possible to configure the storage in a centralized location, and use it decentralized. You can configure the storage when it is connected to one server, and plug it in into another. Or if a server goes down, your storage is still there up and running. E.g. the system (read: storage enclosure) is busy doing a rebuild, MORE or consistency check and the server goes out for lunch, the storage enclosure will go on uninterrupted with with it was doing. Plugging in a hub or switch between the server(s) and enclosure(s) won't affect the storage enclosure set up and configuration. And in the occasion of a controller fault or feature upgrade, you can replace the component without any server downtime.
Though external controllers do not always have the exact same features, ***external controllers provide you with more flexibility***.

Presented here are a few performance details of the SANbloc FC2500 controllers on mentioned configurations. See *Appendix B* for the preferred settings to achieve these results.
Simplex: *(i.e. A Failed-Over Duplex setup or using simplex code)*

- Large block sustained sequential Read of 97 MB/s.
- Large block sustained sequential Write (Write-Back) of 97 MB/s.
- Small block cached sequential Read or Write (Write-Back) of 7900 IO/s.
- Small block random Read or Write (Write-Back) of 4000 IO/s.

Duplex:

- Large block sustained sequential Read of 194 MB/s.
- Large block sustained sequential Write (Write-Through) of 186 MB/s.
- Small block cached sequential Read or Write (Write-Back) of 15500 IO/s.
- Small block random Read or Write (Write-Back) of 8000 IO/s.

---

4. Usage

Ok, now we're going to integrate the information presented in the previous chapters. There are the different RAID levels will their advantages and disadvantages and on top of that parameters like the stripe size and location where the redundancy, if applicable, is performed… and now the quirks of the data itself and the applications have to be added to that.

In real life, you can't please everybody in every situation; likewise with RAID implementations.
Before I go into detail, I'm going to rule out software RAID and PCI RAID controller because of the relatively bad performance and the fact that the SANbloc uses an external controller respectively.

Lets view it from different user scenarios and questions they may fire on you.

1. *"I want the fastest and don't care if I lose my data when one disk fails"*
   This obviously asks for a **RAID 0** striping.

2. *"I want my OS redundant"*
   Due to OS restrictions, only a **RAID 1** allows for this because it can't cope with core system files all shattered around all sorts of disks (aka: no striping). The SANbloc allows for so-called external boot, because it is dependent on the functionality of the HBA (e.g. the Qlogic HBAs) and the server BIOS allowing booting from a PCI card.
   Set up a RAID 1 – go into the Qlogic BIOS with ALT+Q – enable boot and select the desired LUN – install the OS by selecting the RAID 1 LUN as destination drive – reboot the PC – go into the setup section – make the HBA first in the boot order – reboot – OK.

3. *"I want good performance, but also redundancy"*
   Now it's getting more interesting: RAID 1, 3 and 5 can do that for you! You will need to know a bit more information from the customer. In case of the OS there is the RAID 1, on other situations with plain data you would not want to waste more physical disk space for the redundancy calculations than absolutely necessary. So when would you go for a 3 or a 5? Because RAID 3 has this nasty fragile single parity disk, in most occasions **RAID 5 is the best option**: of the x disks in the array, x-1 disks will be used for data and any one disk can fail without any problem. So why RAID 3? With a small amount of typical, video streaming, applications the performance could be marginally better on a RAID 3 because of the way the data is read from disk, only accessing the x-1 disks instead of the x disks that need to be accessed in the situation of a RAID 5.

4. *"I want an exceptionally huge logical drive"*
   At the moment of writing, the maximum disk size on the market is 73 GB. The controller allows you to create an array of 16 drives. Because there is some overhead required for the COD and truncation (the latter needed for matching drives form different manufacturers), the usable space per drive is 69424 MB. RAID 1 is not an option because you will loose half of your disk space for the mirror. Going for the RAID 5: (16-1) x 69424 = 1041360 MB, or about 1 TB. For some this is not enough, thus remains the option of spanning arrays. For example 3 arrays of 10 drives in a **spanned RAID 5** situation:
   - Per array: (10-1) x 69424 = 624816 MB
   - Combine the 3 arrays: 3 x 624816 = 1874448 MB, about 1.8 TB, which is almost the double of a single array of 16 drives.

5. *"So, tell me, what is the optimal amount of drives for an array?"*
   A difficult one to answer, assuming the widely used RAID5. Besides performance and the wish to have a logical drive of size x, the parameter Mean Time Between Failure (MBTF) comes into play. Performance wise, **7 is ideal**. The logical drive size requirement can be anything; hence the amount of drives is **variable**. MBTF is another story. It is a fact that drives will fail at one stage. Statistically, with more drives in an array there is a higher probability that more than 1 drive will fail at some stage, i.e. that the storage enclosure is performing a rebuild and another drive fails. It is highly unlikely, but theoretically not impossible. The statistical probability that this occurs depends on the drive size, how it is used in the system, and the average MBFT figures from the manufacturer.

6. *"I would like to install a database on the SANbloc"*
   Normally sys admins are not too preoccupied with fear and do have a tight budget, so the "standard" **RAID 5** will most likely be suitable. However, that's not it. Besides maximizing usable drive space and having the redundancy, s/he very likely would like to see some performance optimalization. Though there are more factors defining the overall performance, from a hardware RAID perspective the additional advice is to set the **stripe size** of the LUN to **8K**. See §3.1 for more details.

7. *"For me just plain data with a file server, and a mail server as well"*
   The same reasoning as mentioned under item 6 is valid here, but then in the opposite direction: generally, there are larger files (as opposed to the small records of a database) so the requests/writes for/of data should read/write as much as possible in one operation, meaning a large stripe size. Overall: a **RAID 5 with 64K stripe size** will be the best option.

8. *"What if my manager says 'mirror!', but I would like to see some performance as well?"*

In other words: the requirements are a bit conflicting. The only option that comes close to it is **RAID 0+1**, provided the LUN is not used for the OS. Though there won't be many systems out there using this RAID level. It is possible to configure a LUN as a RAID 0+1 from the start. It is possible to move to a RAID 0+1 when doing a MORE expansion operation of a RAID 1 to either an even or an odd number of disks. Because drives are at the moment of writing not one of the cheapest items, the RAID 0+1 option is certainly not highly recommendable.

9. *" Any suggestions for usage of the SANbloc in the graphics industry?"*
   See also Item 3. Video streaming applications need continuous and fast data transfer. In such situations, the performance could be marginally better on a **RAID 3** because of the way the data is read from disk, only accessing the x-1 disks instead of the x disks that need to be accessed in the situation of a RAID 5. However, it should be a user decision what is more important: the marginally better read performance and having the parity disk like a sword of Damocles hanging above your head, or virtually the same performance and the comfortable idea the it doesn't matter which disk of the array fails.

10. *"I must have it all!!"*
    Get a life.
    Or build a real SAN.

---

5. Conclusions

Summarizing this summary of RAID levels, usage and performance, there is **no best RAID level for all occasions**. It all depends on what you want to achieve with your (SAN) system, how you want to use it, how important performance is to you and the budget of course.

However, there are "quick-fix" guidelines:

- **RAID 0**: best performance, but no redundancy
- **RAID 1**: waste of space, but required for OS redundancy
- **RAID 2**: there is no controller that supports it anymore
- **RAID 3**: good for some graphics applications
- **RAID 4**: not supported either
- **RAID 5**: the standard for database applications and file system data, redundancy, relatively good performance, and least amount of "lost" free space.
- **RAID 0+1/10**: some performance improvement over RAID 1, but still waste of space
- **RAID 30/50**: use it when huge logical drives is the major requirement of the system

Chapter 4 is highly recommended reading for an explanation of these guidelines.

Appendix A

Software Versus Hardware RAID

Essentially, the difference between hardware RAID and software RAID implementations is the location where the parity calculations take place. In this perspective, it is irrelevant if there is a PCI-based RAID controller or a RAID controller in the storage enclosure. The SANbloc FC2500 and FC2502 are designed with an external RAID controller, therefore the situation presented in the diagram resembles that implementation. Both diagrams show a write to the disks.

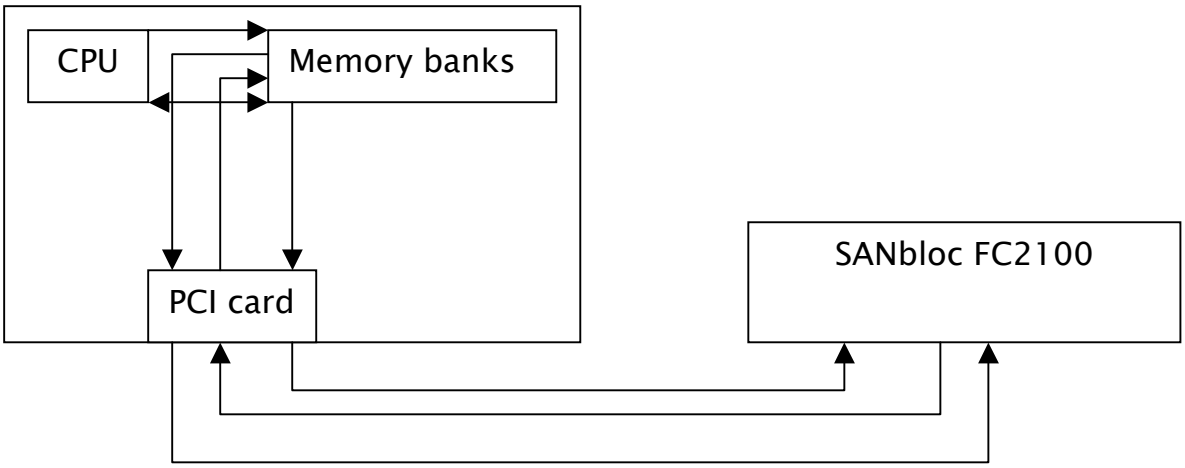


Figure 1. A software RAID write: data from CPU to memory, to storage, back to memory, involving the CPU, parity result back to memory, which then sends it to disk

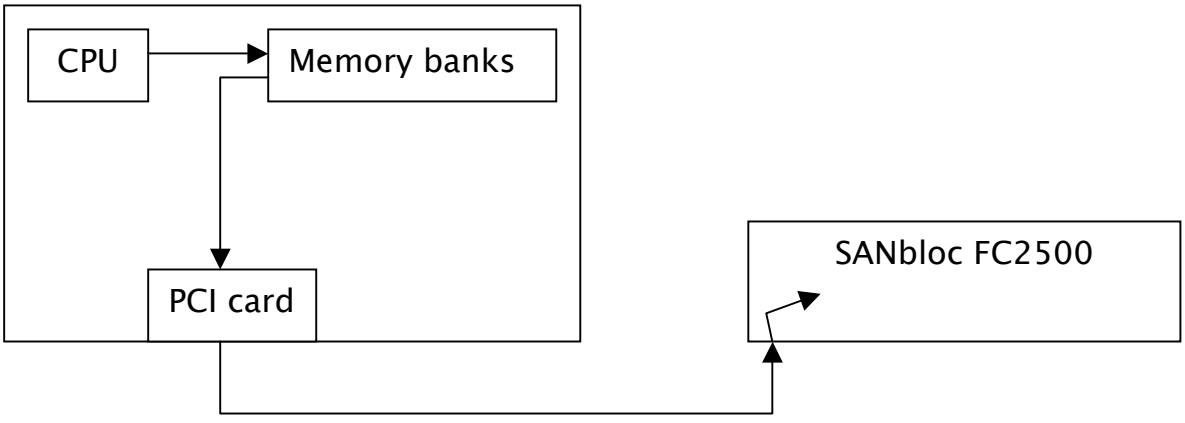


Figure 2. An external hardware RAID: data from CPU to memory and to storage, where the data is stored parity calculation is performed.

Appendix B

Settings For Optimal Performance SANBloc FC2500

Qlogic Host Bus Adapter

(Only the ones you can actually change are listed below)

Host Adapter Settings

Host Adapter BIOS	Disabled
Frame Size	2048
Loop Reset Delay	5
Adapter Hard Loop ID	Enabled
Hard Loop ID	0 or 1 (each host must be different)

Advanced Adapter Settings

Execution Throttle	255
Fast Command Posting	Enabled
>4GB Addressing	Disabled
LUNs Per Target	8 (normal operation either 8 or 32, depending on your set-up)
Enable LIP Reset	No
Enable LIP Full Login	Yes
Enable Target Reset	Yes
Login Retry Count	0
Port Down Retry Count	30
Drivers Load RISC Code	Enabled
Enable Database Updates	No
Disable Database Loads	No
IOCB Allocation	512
Extended Error Logging	Disabled

Contorller Options

The controller's advanced configuration parameters have to be optimized for Large Block Sequential transfers as follows:

Hardware

Automatic Rebuild Management	Enabled
Operational Fault Management	Enabled
Disconnect on First Command	Disabled
SAF-TE use of UPS	Enabled

Physical Default

Rebuild rate	50
Read ahead	Enabled
Super read ahead	Disabled

Mode select SRA enable	Disabled
True Verification of Data	Disabled
Reassign Restricted to 1 blk	Disabled
Write Through Verify	Disabled
Stripe Size(K bytes)	64
RAID 5 Algorithm Rt Asym	
Disk Side	
Coalescing	Enabled
Queue Limit	32
Spin up option	Automatic / 2 / 6 / 0
Host Side	
Disable Wide Operation	Disabled
Vendor Unique TUR	Disabled
Disable CC for Invalid LUN	Enabled
No Pause on Ctrlr not ready	Disabled
On Queue Full give Busy	Disabled
Smart Large Transfers	Enabled
Max IOPs Limit	244
Reset Propagation	Disabled
Serial port 0	
Type	Debug
Baud rate	19200
Data bits	8
Stop bits	2.0
Parity	EVEN
Active - Active	
Conservative Cache	Disabled
Auto Failback	Enabled
Force Simplex	Disabled
Host Bus Reset Delay	0
Ctrlr Pres/Flt Signals	Disabled
Simplex no RSTCOM	Disabled
Disable Busy on Failback	Disabled
Fibre	
PCI Latency Control	Long
Frame Size	2KB (This parameter must be set to 2KB in the Qlogic HBA Bios “Frame Size” parameter in the “Host Adapter Settings” menu)
Hard Loop IDs:	
Ctlr 0 Port 0 Enabled	2
Ctlr 0 Port 1 Disabled	
Ctlr 1 Port 0 Enabled	4
Ctlr 1 Port 1 Disabled	
Topology	MTID
Node Name Retention	Enabled

The controller's advanced configuration parameters can be optimized for Small Block Random transfers as follows:

Hardware

Automatic Rebuild Management	Enabled
Operational Fault Management	Enabled
Disconnect on First Command	Disabled
SAF-TE use of UPS	Enabled

Physical Default

Rebuild rate	50
Read ahead	Enabled
Super read ahead	Disabled
Mode select SRA enable	Disabled
True Verification of Data	Disabled
Reassign Restricted to 1 blk	Disabled
Write Through Verify	Disabled
Stripe Size(K bytes)	8
RAID 5 Algorithm Rt Asym	

Disk Side

Coalescing	Enabled
Queue Limit	8
Spin up option	Automatic / 2 / 6 / 0

Host Side

Disable Wide Operation	Disabled
Vendor Unique TUR	Disabled
Disable CC for Invalid LUN	Enabled
No Pause on Ctrlr not ready	Disabled
On Queue Full give Busy	Disabled
Smart Large Transfers	Enabled
Max IOPs Limit	244
Reset Propagation	Disabled

Serial port 0

Type	Debug
Baud rate	19200
Data bits	8
Stop bits	2.0
Parity	EVEN

Active - Active

Conservative Cache	Disabled
Auto Failback	Enabled
Force Simplex	Disabled
Host Bus Reset Delay	0
Ctrlr Pres/Flt Signals	Disabled
Simplex no RSTCOM	Disabled
Disable Busy on Failback	Disabled

Fibre

PCI Latency Control	Long
Frame Size	512 (This parameter must be set to 512B in the Qlogic HBA Bios "Frame Size" parameter in the "Host Adapter Settings" menu.)

Hard Loop IDs:

Ctlr 0 Port 0 Enabled	2
Ctlr 0 Port 1 Disabled	
Ctlr 1 Port 0 Enabled	4
Ctlr 1 Port 1 Disabled	
Topology	MTID
Node Name Retention	Enabled

Operating System

A. Determine the optimal Scatter/Gather list size for a given configuration.

- The Strip size for large data transfers should be set to 64KB. The optimal large data transfer Raid5 group is 7+1, so the Row size that should be used is the number of active data drives of 7 multiplied by 64KB = 448KB.
- Then we divide the Row size by the maximum NT transfer size of 4KB which come out to be 448KB / 4KB = 112 Scatter/Gather elements.
- Add 1 to this number and you come up with the optimal Scatter/Gather number of 113 (0x71 in Hex) for a 7+1 Rad5 LUN.

B. Set the Scatter/Gather list in the NT registry.

- Click on "Start", select "Run" and launch the REGEDT32 program.
- Select HKEY_LOCAL_MACHINE and follow the tree structure down to the driver which you want to improve the performance for, which is the Qlogic driver:
  HKEY_LOCAL_MACHINE
  SYSTEM
  CurrentCotrolSet
  Services
  Ql2200
- Select the "Ql2200" key and use the Edit/Add key to create a new subkey named "parameters".
- Select the new "parameters" key and use the Edit/Add to create a new "Device" subkey. If multiple HBAs are installed you can specify separate subkeys for each HBA by using the name of the "DeviceN", where N = the SCSI HBA number (0, 1, 2, … N).
- The tree structure should now look as follows:
  HKEY_LOCAL_MACHINE
  SYSTEM
  CurrentCotrolSet
  Sevices

Ql2200

Parameters

Device (or DeviceN)

- Select the “Device key and use the Edit/Add Value key to add a new value name of “MaximumSGList”. Set the data type to “REG_DWORD” and enter the optimal Scatter/Gather list size calculated above (0x71).

C. Exit the registry editor, then shutdown and reboot the system.

Note: Significant performance improvements can be realized by controlling the NT/win2k number of Scatter/Gather elements; however, some HBA mini-port drivers can lower the NumberOfPhysicalBreaks if it has a limitation on the number that it can support. Please check with the specific HBA documentation for additional information if an HBA other than the QLA2200 family is used.