

Chapter 26

-

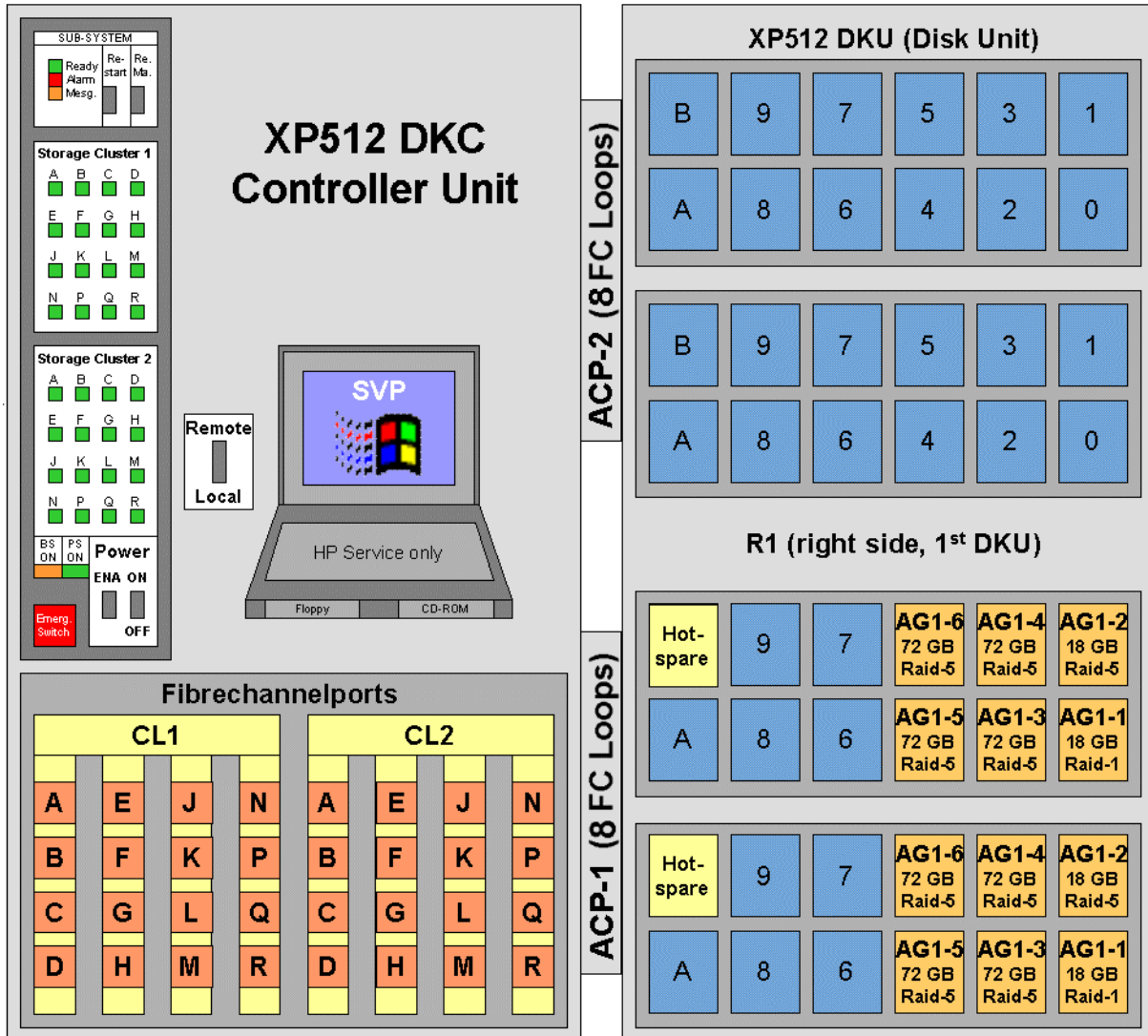
XP Disk Arrays

INDEX

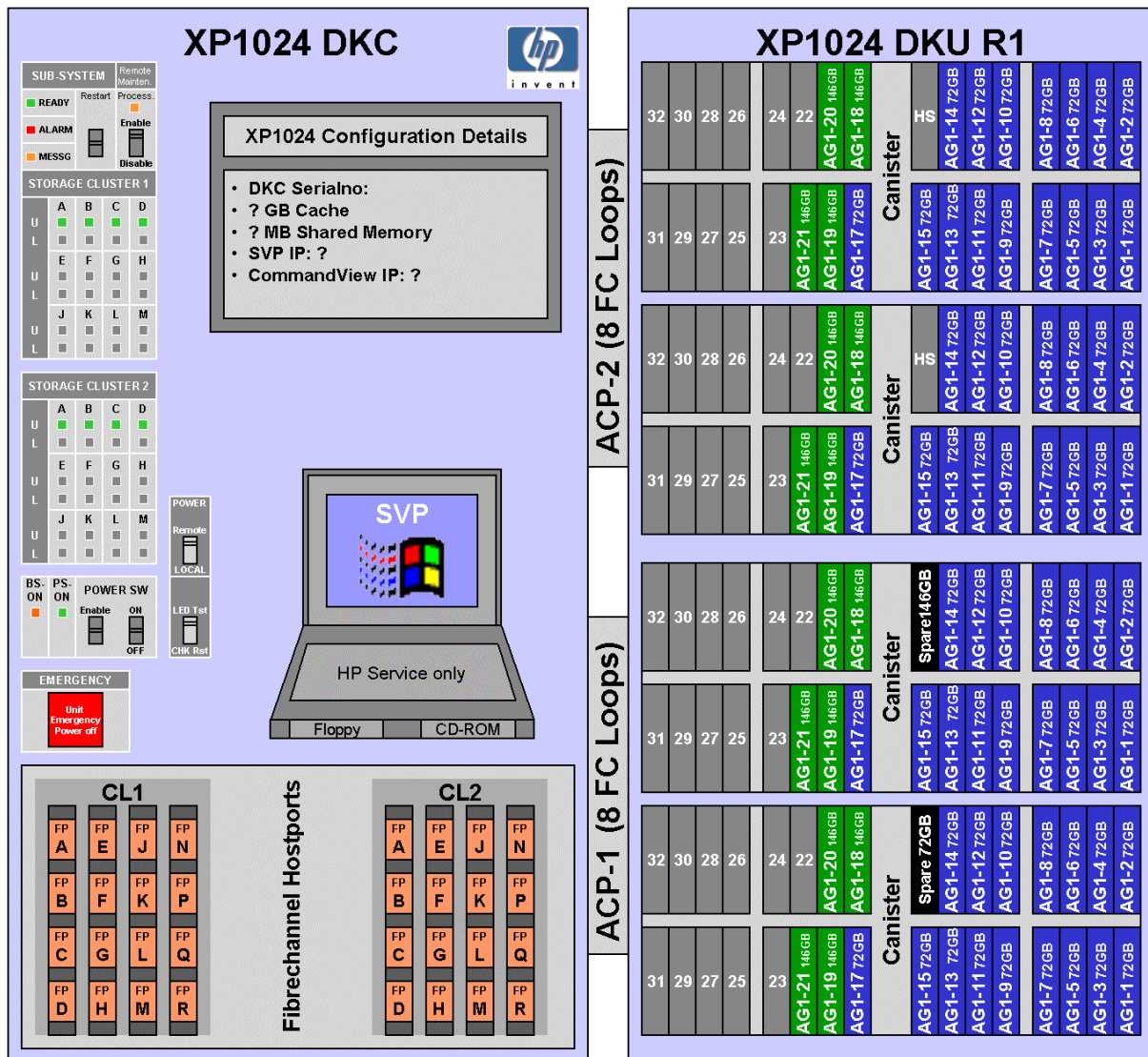
Introduction and Hardware Overview	3
Generation 400: XP512 Diskarray (smaller version: XP48).....	3
Generation 450: XP1024 Diskarray (smaller version: XP128).....	4
Startup/Shutdown Procedure	6
Basic Configuration: LDEVs, LUNs, LUSE, CVS	7
Portmapping and LUN Addressing	10
Additional XP Diskarray Functionality	12
Secure Manager.....	12
Cache LUN.....	12
AutoLUN.....	12
Business Copy.....	12
Continuous Access.....	13
Command View and Remote Control.....	13
Performance Manager and Performance Advisor.....	13
Autopath.....	13
Raidmanager - Host Control of XP Functionality	13
Performance Aspects for XP Diskarrays	18
Frontend Performance: XP48/XP512/XP256 fibre channel ports.....	18
Backend Performance: Extent-Based Striping.....	21
Additional Information	24

Introduction and Hardware Overview

Generation 400: XP512 Diskarray (smaller version: XP48)



Generation 450: XP1024 Diskarray (smaller version: XP128)



This is a logical view of a sample XP1024 diskarray. The basic components of the XP diskarrays are:

- DKC: the controller unit which holds cache, shared memory, frontpanel, hostports, SVP (Laptop) and ACP.
- DKU: disk Control Unit. This unit holds the disks, half of the disks at the front side, the other half at the back side. Disks are always grouped in array groups. Depending of the XP diskarray type there can be a maximum number DKUs: 4 for XP256 (2 left, 2 right side), 6 for XP512 (3 left, 3 right side) and 4 for XP1024 (2 left, 2 right side). The XP48 and XP128 doesn't have a separate DKU, instead all 48 / 128 disks are located at the rear side of the DKC.
 A XP512 DKU can hold up to 23 Arraygroups (11 in the lower part, 12 in the upper part), a XP256 DKU can hold up to 15 Arraygroups (7 in the lower part, 8 in the upper part), a XP1024 DKU can hold up to 63 Arraygroups (31 in the lower part, 32 in the upper part).
- Arraygroup: each AG consists of 4 physical disks. Each arraygroup forms a raidgroup, either raid-5 or raid-1/0. For the XP48/XP512 the raidlevel can be chosen for each

arraygroup individually. For the XP256 the raidlevel must be chosen for the complete ACP, so all arraygroups connected to this ACP are running the same raidlevel. Logical Devices (LDEV) within an Arraygroup are always striped evenly across all 4 physical disks within the Arraygroup. An exception to this is the XP256: while the raid-5 implementation is similar to the XP48/XP512, there is no raid-1/0 implementation, but instead a raid-1 implementation, that divides an arraygroup consisting of 4 disks in two raid-1 groups (no striping, only mirroring) each consisting of two disks. With the introduction of the XP128 and XP1024 there is the possibility to have 8 instead of 4 disks in one Arraygroup, but only if these 8 disks can be spread across 2 ACPs. If using Raid1/0 this implementation is then called 4D+4D. If using Raid5 this implementation is then called 7D+1P and has the advantage that the capacity can be used more efficiently (87.5% data efficiency) compared to the “classic” 3D+1P implementation (75% data efficiency).

- ACP: the Access-Control-Pair connects the DKC with the DKU. Each DKC consists of 8 FC-copper busses (XP1024+XP128 +XP512+XP48) or 8 FW/SCSI busses (XP256). There can be a maximum of 4 ACPs per XP512 and XP256: 2 at the left and two at the right side of the DKC.
- SVP: the Service Processor is a Windows 95 (XP256), Windows 98 (XP512+XP48) or Windows2000 (XP1204+XP128) laptop. Microcode updates and array configurations can be done with the SVP. The SVP contains an additional LAN interface that allows a connection to the a PC running “Remote Control XP” (XP256, XP512, XP48) or “Command View” (all XPs).
- Hostports: The hostport interface cards always come in pairs. One card is inserted in CL1, the other one in CL2. CL1 and CL2 are powered by separate power supplies (if working with primary and alternate links in LVM it is a good idea to chose one link to CL1 and the other link to CL2). The ports themselves are alphabetically ordered. For each hostport there is a respective green LED. If the LED is blinking, then there is no fibre channel link established. An established fibre channel link is displayed with a steady green light. There are different hostport interfaces available, starting with 4 ports per pair (XP256, XP512, XP48), 8 ports per pair (XP512, XP48, XP1024, XP128) up to 16 ports per pair (XP1024, XP128).
- Disks and Hotspares: All disks are dual ported, connecting to two fibre channel (or fwscsi for XP256) cables from the ACP. Hotspare disks are global so they can replace any other disk of the same or lower capacity.

Startup/Shutdown Procedure

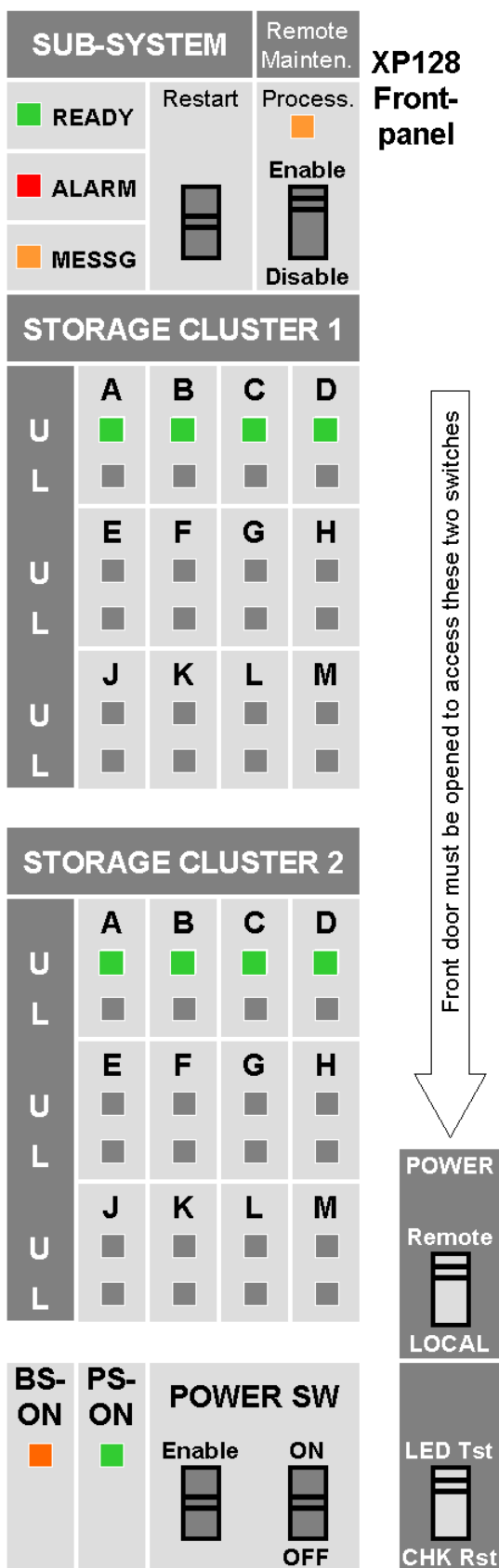
The power up/down behaviour of an XP diskarray differs from other HP diskarrays. One of the differences is that the XP diskarray must be manually started up after a power failure. Shutdown and startup of the diskarray should always be done by an HP customer engineer, because it is crucial to exactly follow the procedure. The steps are as follows:

Shutdown of an XP diskarray

- Make sure that the XP is not in a warning state (“Message LED” must be off)
- Turn the “Remote/Local” switch to “Local” (need to open the front door of the DKC)
- At the same time press “Enable” and “Off”
- The XP now powers down. This process may take up to 15 minutes and is finished as soon as both LEDs “Ready” and “PS-ON” are off.
- Shutdown the SVP (Laptop).
- Turn the “Remote/Local” switch to “Remote” (in this “Remote” position, all other switches are disabled for safety).
- If the XP is without power for more than a couple of days a customer engineer should disable the batteries to prevent discharging.

Startup of an XP diskarray

- Power on the SVP (Laptop).
- Turn the “Remote/Local” switch to “Local”
- At the same time press “Enable” and “ON”
- The XP now powers up. This process may take up to 15 minutes and is finished as soon as the LED “Ready” is on.
- Turn the “Remote/Local” switch to “Remote” (in this “Remote” position, all other switches are disabled for safety).



Basic Configuration: LDEVs, LUNs, LUSE, CVS

XP devices seen in `ioscan` never refer to real physical disks in the array, but instead to logical LUNs. When installing an arraygroup (= a bundle of 4 physical disks) it must be decided on how this arraygroup is configured:

- choose a raid level: raid-5 or raid-1/0 (where raid-5 is usually the preferred one, because of higher storage efficiency with only a small impact in write performance)
- choose an emulation type for the array group: The emulation type decides, how many logical devices (LDEVs) here are and how big these LDEVs are.

The following emulation types can be used for the configuration of XP array groups:

Emulation type	Space/GB	Space/MB
OPEN-K	1,75	1788
OPEN-3	2,29	2347
OPEN-8	6,84	7007
OPEN-9	6,88	7043
OPEN-E	13,58	13906
OPEN-L	33,94	34761
OPEN-M	43,94	45000
OPEN-V	variable	variable

Dependent on the disk type, raid level and emulation type you will get a certain number of LDEVs per array group. If there is a rest capacity in the array group, then this capacity can be used in the XP48/XP512/XP1024/XP128 for creating “small” logical devices, CVS devices (mentioned later). XP256 diskarrays can not use this rest capacity.

Starting from a certain microcode version 21.04 of the XP1024/128 there is a variable Emulation type called “OPEN-V” which allows to specify any size from 46MB up to 2TB. Internally this “OPEN-V” is a mixture between CVS Volumes and LUSE.

The following table shows the usable space, number of LDEVs and the “rest capacity” (=left over) per array group for different raid levels and emulation types:

XP256							
Drive type	Emulation	Raid-1/0			Raid-5		
		Usable [GB]	LDEV Number	CVS rest [MB]	Usable [GB]	LDEV Number	CVS rest [MB]
15GB disks XP256 only	OPEN-3	27	12	-	41	18	-
	OPEN-8	27	4	-	41	6	-
	OPEN-9	27	4	-	41	6	-
47GB disks XP256 only	OPEN-3	87	38	-	131	57	-
	OPEN-8	82	12	-	130	19	-
	OPEN-9	83	12	-	131	19	-
72GB disks XP256 only	OPEN-3	135	59	-	202	88	-
	OPEN-8	130	19	-	198	29	-
	OPEN-9	131	19	-	199	29	-
	OPEN-E	136	10	-	204	15	-

XP48, XP512							
Drive type	Emulation	Raid-1/0			Raid-5		
		Usable [GB]	LDEV Number	CVS rest [MB]	Usable [GB]	LDEV Number	CVS rest [MB]
18GB/10k disk DKR2B/2C-18 XP48 + XP512	OPEN-3	32	14	2264	50	22	1044
	OPEN-8	34	5	80	48	7	3633
	OPEN-9	28	4	6962	48	7	3382
	OPEN-E	27	2	7398	41	3	11099
18GB/15k disk DKS2A-18 XP48 + XP512	OPEN-3	32	14	1142	48	21	1713
	OPEN-8	28	4	5984	48	7	1949
	OPEN-9	28	4	5840	48	7	1698
	OPEN-E	27	2	6276	41	3	9415
47GB/10k disk DKR1B/C-J47FC XP48 + XP512	OPEN-3	87	38	575	131	57	864
	OPEN-8	82	12	5687	130	19	1505
	OPEN-9	83	12	5255	131	19	821
	OPEN-E	81	6	6565	122	9	9848
	OPEN-M	87	2	0	131	3	0
72GB/10k disk DKR1C/D-J72FC XP48 + XP512	OPEN-3	135	59	215	202	88	1500
	OPEN-8	130	19	5553	198	29	4832
	OPEN-9	131	19	4879	199	29	3788
	OPEN-E	136	10	0	204	15	0
	OPEN-L	135	4	0	203	6	0
180GB/7k disk DKS1A-H180FC XP48 + XP512	OPEN-3	318	139	1977	478	209	1789
	OPEN-8	314	46	5883	478	70	1798
	OPEN-9	317	46	4227	475	69	6340
	OPEN-E	312	23	9246	476	35	6917
	OPEN-L	305	9	16210	475	14	6932
	OPEN-M	307	7	14000	439	10	43626

XP1204 / XP128 (Careful, CVS Rest is just an approximate value, maybe different!)													
		Raid5 3D+1P			Raid1 2D+2D			Raid5 7D+1P			Raid1 4D+4D		
		Cap /GB	#LD-EVs	CVS Rest	Cap /GB	#LD-EVs	CVS Rest	Cap /GB	#LD-EVs	CVS Rest	Cap /GB	#LD-EVs	CVS Rest
36 GB 15K	Open3	98	43	1200	64	28	2200	231	101	1300	128	56	4400
	Open8	96	14	3800	62	9	4600	226	33	6700	123	18	9300
	Open9	96	14	3300	62	9	4300	227	33	5600	124	18	8700
	OpenE	95	7	4500	54	4	11600	230	17	2100	108	8	23200
73 GB 10K	Open3	201	88	1800	135	59	400	471	206	2700	270	118	900
	Open8	198	29	4800	130	19	5400	472	69	2700	260	38	10800
	Open9	199	29	3900	130	19	4700	467	68	7000	261	38	9500
	OpenE	203	15	0	135	10	0	474	35	200	271	20	100
	OpenL	203	6	0	136	4	0	474	14	0	271	8	0
72 GB 15K	Open3	197	86	2400	130	57	2300	462	202	2700	261	114	4600
	Open8	198	29	1100	130	19	2900	458	67	6900	260	38	5800
	Open9	192	28	6700	130	19	2300	460	67	4800	261	38	4600
	OpenE	190	14	9100	122	9	10300	461	34	4300	244	18	20700
	OpenL	169	5	28300	102	3	29500	441	13	23300	203	6	59100
144GB 10K	Open3	398	174	2500	265	116	1600	932	407	3600	531	232	3300
	Open8	396	58	4300	260	38	7100	929	136	5700	519	76	14300
	Open9	398	58	2400	261	38	5900	927	135	7800	522	76	11900
	OpenE	393	29	7500	257	19	9300	935	69	500	515	38	18600
	OpenL	373	11	26600	237	7	28400	915	27	19400	474	14	56800

During the installation of an array group, each LDEV in this array group gets a number, the so called LDEV number. Because this LDEV number can not exceed 255 and you might have more LDEVs in the XP diskarray, this number is preceded by the “cu Number” (control-unit). The combination of cu-number and LDEV number is unique within an XP diskarray. Example: “1:0A” is cu number 1 and LDEV number 0A.

In XP256 diskarrays, a maximum number of 4 cu’s can be configured, XP512 diskarrays can hold 16 cu’s. Very often the cu number of the LDEVs is configured to represent the physical location of the LDEV. For example: The lower part of the first diskrack (DKU R1) uses cu number 0 for the LDEVs, the upper part of DKU R1 uses cu number 1.

You can directly use the LDEVs in order to map them to the hosts (example 1 below), but you also may want to create LUNs that have more or less capacity than the basic emulation type. There are two ways to do this:

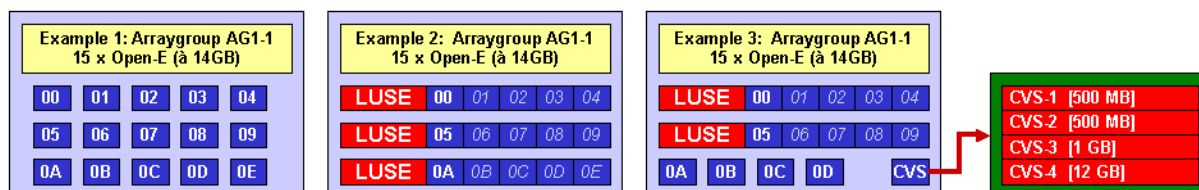
- **LUSE: Logical-Unit-Size-Extension:**

A LUSE combines several basic LDEVs in a “bigger” LDEV. In example 2 below, 3 LUSES have been created, each consisting of 5 LDEVs. The LDEV number for these LUSES correspond to the LDEV number of the first LDEV in the LUSE. It is possible to create LUSES that contain LDEVs of different array groups. This is called “dispersed LUSE”. Although with dispersed LUSES, the data is spread among several arraygroups, this can not be compared to a real striping, because the “stripe size” in dispersed LUSES is the size very big, it matches the capacity of an LDEV, for example 13.5GB / OPEN-E. There are two limitations for the creation of LUSES: all LDEVs must be in the same cu and the total number of LDEVs per LUSE must not exceed 36.

- **CVS: Custom Volume Sizing:**

It may be desirable to create very small LUNs. XP diskarrays provide a method of fragmenting LDEVs in smaller pieces, called CVS. Example 3 shows the creation of 4

CVS with different size out of one LDEV. LDEV 0E in this example is “deleted”, the freed space is used to create 4 smaller LDEVs. CVS is also a method of using the left-over capacity of the array groups (see previous table).



Portmapping and LUN Addressing

Once you have finished the basic configuration and created LUSEs and CVSs as needed, you can assign the LDEVs to the XP ports. These ports may connect directly to a host or use fibre channel switches or hubs as interconnect devices in order to connect several hosts to one XP port. Once an LDEV is “mapped” to a port (or multiple ports), it is now called a LUN. But there is more than just to define which LDEV connects to which XP port: in addition an address must be assigned to each LUN. Unfortunately the XP internal addressing scheme uses a hexadecimal scheme (00-FF), where HPUX uses an octal scheme (t0d0 - t15d7). The following table “translates” the XP LUN (hex) to HP-UX devicefiles (example: XP LUN 2D is translated to an HP-UX devicefile c*t5d5):

XP LUN to HPUX-Devicefile Mapping							
00= t0d0	01= t0d1	02= t0d2	03= t0d3	04= t0d4	05= t0d5	06= t0d6	07= t0d7
08= t1d0	09= t1d1	0A= t1d2	0B= t1d3	0C= t1d4	0D= t1d5	0E= t1d6	0F= t1d7
10= t2d0	11= t2d1	12= t2d2	13= t2d3	14= t2d4	15= t2d5	16= t2d6	17= t2d7
18= t3d0	19= t3d1	1A= t3d2	1B= t3d3	1C= t3d4	1D= t3d5	1E= t3d6	1F= t3d7
20= t4d0	21= t4d1	22= t4d2	23= t4d3	24= t4d4	25= t4d5	26= t4d6	27= t4d7
28= t5d0	29= t5d1	2A= t5d2	2B= t5d3	2C= t5d4	2D= t5d5	2E= t5d6	2F= t5d7
30= t6d0	31= t6d1	32= t6d2	33= t6d3	34= t6d4	35= t6d5	36= t6d6	37= t6d7
38= t7d0	39= t7d1	3A= t7d2	3B= t7d3	3C= t7d4	3D= t7d5	3E= t7d6	3F= t7d7
40= t8d0	41= t8d1	42= t8d2	43= t8d3	44= t8d4	45= t8d5	46= t8d6	47= t8d7
48= t9d0	49= t9d1	4A= t9d2	4B= t9d3	4C= t9d4	4D= t9d5	4E= t9d6	4F= t9d7
50=t10d0	51=t10d1	52=t10d2	53=t10d3	54=t10d4	55=t10d5	56=t10d6	57=t10d7
58=t11d0	59=t11d1	5A=t11d2	5B=t11d3	5C=t11d4	5D=t11d5	5E=t11d6	5F=t11d7
60=t12d0	61=t12d1	62=t12d2	63=t12d3	64=t12d4	65=t12d5	66=t12d6	67=t12d7
68=t13d0	69=t13d1	6A=t13d2	6B=t13d3	6C=t13d4	6D=t13d5	6E=t13d6	6F=t13d7
70=t14d0	71=t14d1	72=t14d2	73=t14d3	74=t14d4	75=t14d5	76=t14d6	77=t14d7
78=t15d0	79=t15d1	7A=t15d2	7B=t15d3	7C=t15d4	7D=t15d5	7E=t15d6	7F=t15d7
80= t0d0	81= t0d1	82= t0d2	83= t0d3	84= t0d4	85= t0d5	86= t0d6	87= t0d7
88= t1d0	89= t1d1	8A= t1d2	8B= t1d3	8C= t1d4	8D= t1d5	8E= t1d6	8F= t1d7
90= t2d0	91= t2d1	92= t2d2	93= t2d3	94= t2d4	95= t2d5	96= t2d6	97= t2d7
98= t3d0	99= t3d1	9A= t3d2	9B= t3d3	9C= t3d4	9D= t3d5	9E= t3d6	9F= t3d7
A0= t4d0	A1= t4d1	A2= t4d2	A3= t4d3	A4= t4d4	A5= t4d5	A6= t4d6	A7= t4d7
A8= t5d0	A9= t5d1	AA= t5d2	AB= t5d3	AC= t5d4	AD= t5d5	AE= t5d6	AF= t5d7
B0= t6d0	B1= t6d1	B2= t6d2	B3= t6d3	B4= t6d4	B5= t6d5	B6= t6d6	B7= t6d7
B8= t7d0	B9= t7d1	BA= t7d2	BB= t7d3	BC= t7d4	BD= t7d5	BE= t7d6	BF= t7d7
C0= t8d0	C1= t8d1	C2= t8d2	C3= t8d3	C4= t8d4	C5= t8d5	C6= t8d6	C7= t8d7
C8= t9d0	C9= t9d1	CA= t9d2	CB= t9d3	CC= t9d4	CD= t9d5	CE= t9d6	CF= t9d7
D0=t10d0	D1=t10d1	D2=t10d2	D3=t10d3	D4=t10d4	D5=t10d5	D6=t10d6	D7=t10d7
D8=t11d0	D9=t11d1	DA=t11d2	DB=t11d3	DC=t11d4	DD=t11d5	DE=t11d6	DF=t11d7
E0=t12d0	E1=t12d1	E2=t12d2	E3=t12d3	E4=t12d4	E5=t12d5	E6=t12d6	E7=t12d7
E8=t13d0	E9=t13d1	EA=t13d2	EB=t13d3	EC=t13d4	ED=t13d5	EE=t13d6	EF=t13d7
F0=t14d0	F1=t14d1	F2=t14d2	F3=t14d3	F4=t14d4	F5=t14d5	F6=t14d6	F7=t14d7
F8=t15d0	F9=t15d1	FA=t15d2	FB=t15d3	FC=t15d4	FD=t15d5	FE=t15d6	FF=t15d7

Because HP-UX uses SCSI-2 addressing with only 128 addresses the only way for HP-UX to represent addresses exceeding XP LUN 7F is to create an additional ext_bus device for another 128 addresses. Example: XP LUN 7F refers to HP-UX devicefile c50t15d7. XP LUN 80 refers to HP-UX devicefile c51t0d0.

Important to notice is that the LUN number is the address of an XP device. It is NOT the LDEV number.

Example:

LDEV 0:00 can be mapped to XP Port CL1-A with LUN number 2D. Using the very helpful tool “xpinfo” (refer to <http://xpnow.rose.hp.com/support/>), the output for this particular example would be:

```
# xpinfo -i
Device File                ALPA Tgt Lun Port CU:LDev Type                Serial#
=====
/dev/rdisk/c50t5d5        b2  00  2D CL1A 00:00  OPEN-E*2                00031157
```

Additional XP Diskarray Functionality

Secure Manager

Secure Manager is used in environment where multiple hosts connect via a fc switch or fc hub to a single XP port. Secure Manager allows LUNs to be made visible only to certain hosts. This can be achieved because each host bus adapter has a unique “World-Wide-Name”, similar to the MAC address of LAN interfaces. If a host bus adapter is replaced, the WWN changes also and the Secure Manager configuration must be modified accordingly. To find the WWN of a host bus adapter the following method can be used:

Method for finding the WWN of “old” Tachyon based host bus adapters

```
# ioscan -fknH 10/8
Class      I  H/W Path  Driver      S/W State H/W Type  Description
=====
fc         0  10/8     fcT1        CLAIMED   INTERFACE HP Fibre Channel ..
lan        3  10/8.5   fcT1_cntl   CLAIMED   INTERFACE HP Fibre Channel ..
                                     /dev/fcms3

# fcmsutil /dev/fcms3 | grep "Port World"
N_Port Port World Wide Name = 0x10000060B03ED5CD <= WWN of HBA
```

Method for finding the WWN of “new” Tachylite based Hostbusadapters

```
# ioscan -fknH8/4
Class      I  H/W Path  Driver      S/W State H/W Type  Description
=====
ba         1  8/4      GSCToPCI    CLAIMED   BUS_NEXUS GSCToPCI Bridge
fc         0  8/4/1/0  td          CLAIMED   INTERFACE HP Tachyon TL/TS ..
                                     /dev/td0

# fcmsutil /dev/td0 | grep "Port World"
N_Port Port World Wide Name = 0x50060b00000903f8 <= WWN of HBA
```

Cache LUN

Cache LUN locks a complete LUN in cache. The space in cache then is completely reserved for this LUN. This makes the particular LUN very special: it is very fast (because completely in cache), but also extremely expensive. Cache LUN is rarely used.

AutoLUN

Autolun is used to move LDEVs within the XP diskarray from one physical location to another. The purpose behind it is to optimize the XP performance dynamically: There maybe areas in the XP that hold “fast” LDEVs (e.g. 18GB arraygroup in raid-1/0) and other areas with “slow” LDEVs (e.g. 180GB arraygroup in raid-5). With Autolun you can move LDEVs between these different locations online.

Business Copy

With Business Copy it is possible to create copies of LUNs within the XP. These copies maybe used to “clone” data or use it as snapshots for split-mirror backups. The Business Copy functionality can be operated from the XP directly or through host commands using Raidmanager.

Continuous Access

Continuous Access is similar to Business Copy, the difference is that the LUNs are copied not within one XP, but between two XP diskarrays. CA is usually used to implement HA solutions like MetroCluster, where data is mirrored between two datacenters without using host-based mirroring. Special hardware requirements must be met in order to connect two XP diskarrays (need special FC or ESCON connection interfaces in the XPs).

Command View and Remote Control

The XP configuration is usually done through a software called “Remote Control Software” that runs on a separate PC that is connected over LAN to the XP. In addition HP developed “Command View”. Command View runs as webserver on a separate PC and allows the XP configuration through a web interface. Although CV is nicely designed, it is rarely used for the XP configurations because it is much slower than the “old” program “Remote Control Software”.

Performance Manager and Performance Advisor

Performance Manager has been developed by HDS and has been replaced by an HP development: Performance Advisor. While the old product only collected performance data from the XP itself, the new product “Performance Advisor” runs as hostagent and collects both XP and host performance data.

Autopath

Autopath is a host-based software mainly used for Windows hosts. Without Autopath, Windows is not capable to handle LUNs with more than one path to the server. Autopath can handle multiple paths to single LUNs and provides path failover plus dynamic load balancing. Meanwhile Autopath is available also for HP-UX but rarely used because LVM already provides similar functionality (alternate pathing).

Raidmanager - Host Control of XP Functionality

Raidmanager is a host-based software (available for all supported operating systems) that allows the host to control XP internal operations. Raidmanager is used to control Business Copy and Continuous Access Volumes.

Prerequisite for Raidmanager is that there is a physical connection from the host to the XP (FC for XP48/XP512/XP1024/XP128 plus SCSI for XP256). In order to assign commands or read the status from the host, there must be at least one LUN that can exclusively be used for this communication. No data must reside on this LUN. This LUN is called “Command Device”. Usually a very small CVS device is used as command device in order to avoid wasting capacity. A LUN that is marked as command device can be recognized in ioscan through the description ending with “-CM” (for example: OPEN-E-CVS-CM).

After the installation of Raidmanager (/HORCM), a Raidmanager instance must be created for each part of the LDEV copy (which can be either a copy within the same XP = Business Copy

or a copy in another XP = Continuous Access). The primary copy is called P-Vol, the copy of this P-Vol is called S-Vol. Each instance must have an entry in /etc/services.

Example:

```
horcm0 5000/udp          # XP512 RAID Mgr instance 0
horcm1 5001/udp          # XP512 RAID Mgr instance 1
```

Copies are only allowed between LDEVs of the same size. Through the Raidmanager configuration files (/etc/horcmx.conf, where x is the instance number) the definition of the pairs can be made. In the following example it is assumed that you want to create a Business Copy pair for 2 LUNs:

LUN 01 (=t0d1) will be the P-Vol and will be mirrored to S-Vol LUN 02 (=t0d2)

LUN 03 (=t0d3) will be the P-Vol and will be mirrored to S-Vol LUN 04 (=t0d4)

The Command Device is LUN 00 (=t0d0).

Careful! If using an XP1024/XP128 with LUN Security enabled, the “target ID” and “LU#” fields in the configuration files may not correspond to the SCSI Target and SCSI LUN of the respective Devicefiles! This is discussed later in this chapter.

configuration file for the P-Vols /etc/horcm0.conf:

```
HORCM_MON
#ip_address      service      poll(10ms)    timeout(10ms)
bbx0417          horcm0       1000          3000

HORCM_CMD
#dev_name        dev_name      dev_name
/dev/rdisk/c8t0d0

HORCM_DEV
#dev_group       dev_name      port#         TargetID      LU#
GROUP1          dev1          CL1-A         0             1
GROUP1          dev2          CL1-A         0             3

HORCM_INST
#dev_group       ip_address    service
GROUP1          bbx0417      horcm1
```

configuration file for the S-Vols /etc/horcm1.conf:

```
HORCM_MON
#ip_address      service      poll(10ms)    timeout(10ms)
bbx0417          horcm1       1000          3000

HORCM_CMD
#dev_name        dev_name      dev_name
/dev/rdisk/c8t0d0

HORCM_DEV
#dev_group       dev_name      port#         TargetID      LU#
GROUP1          dev1          CL1-A         0             2
GROUP1          dev2          CL1-A         0             4
```

```
HORCM_INST
#dev_group      ip_address      service
GROUP1          bbx0417         horcm0
```

In order to issue Raidmanager commands, one raidmanager process per instance must be started:

```
# horcmstart.sh 0 1
  starting HORCM inst 0
  HORCM inst 0 starts successfully.
  starting HORCM inst 1
  HORCM inst 1 starts successfully.

# ps -ef | grep -i horc
  root  2365      1  0 09:48:34 ?          0:00 horcmd_01
  root  2360      1  4 09:48:31 ?          0:00 horcmd_00
```

Every time the configuration files of Raidmanager change, all Raidmanager processes must be restarted through shutting down existing processes and starting them up again:

```
# horcmshutdown.sh 0 1
  inst 0:
  HORCM Shutdown inst 0 !!!
  inst 1:
  HORCM Shutdown inst 1 !!!
```

The following example shows how to create a Business copy pair, check the pair status and split the pair.

```
# export HORCC_MRCF=1           # for CA Volumes: unset HORCC_MRCF
# export HORCMINST=0           # P-Vol is instance 0
# pairdisplay -g GROUP1 -fc -CLI
Group   PairVol L/R   Port# TID LU-M Seq# LDEV# P/S Status %
GROUP1 dev1    L    CL1-A 0  1 0 60069  2 P-Vol COPY  24
GROUP1 dev1    R    CL1-A 0  2 0 60069  3 S-Vol COPY  24
GROUP1 dev2    L    CL1-A 0  3 0 60069  2 P-Vol COPY  22
GROUP1 dev2    R    CL1-A 0  4 0 60069  3 S-Vol COPY  22

# pairsplit -g GROUP1
# pairdisplay -g GROUP1 -fc -CLI
Group   PairVol L/R   Port# TID LU-M Seq# LDEV# P/S Status %
GROUP1 dev1    L    CL1-A 0  1 0 60069  2 P-Vol P-SUS 100
GROUP1 dev1    R    CL1-A 0  2 0 60069  3 S-Vol S-SUS 100
GROUP1 dev2    L    CL1-A 0  3 0 60069  2 P-Vol P-SUS 100
GROUP1 dev2    R    CL1-A 0  4 0 60069  3 S-Vol S-SUS 100
```

The creation of Continuous Access Volumes works similar to Business Copy volumes, however there are some additional parameter (like the parameter “fence-level” that defines the characteristics of the mirror in case of link failures between the XP diskarrays).

The following table explains some basic Raidmanager commands. Careful! Raidmanager commands can be very powerful! Any mistake can be penalized with data loss!

<i>Raidmanager command</i>	Description
raidqry -l	displays Raidmanager version, XP serialnumber, microcode revision and cache capacity

<code>raidscan -p CL1-B</code>	displays all XP LUNs mapped to XP Port CL1-B
<code>paircreate -g GROUP1 -vl</code>	creates the pair GROUP1 as defined in the configuration files
<code>pairdisplay -g GROUP1 -CLI -fc</code>	Displays pairstate of GROUP1 (-l only shows P-Vols)
<code>pairsplit -g GROUP1</code>	Suspend GROUP1: mirroring is stopped and S-Vol is in read/write
<code>pairsplit -S -g GROUP1</code>	Delete GROUP1: pair relation between P-Vol and S-Vol is deleted.
<code>pairresync -g GROUP1</code>	Resynchronize P-Vol to S-Vol
<code>pairresync -g GROUP1 -restore</code>	Careful! Reverse resync: S-Vol to P-Vol ! Only for BC, not CA. System Mode 80 of the XP determines the characteristic of this resync.
<code>pairevtwait -g CA1 -s pair -t\ 3600</code>	wait at most 3600 sec. for status "pair"
<code>horctakeover -g CA1</code>	CA only, runs on S-Vol side. Swaps P-Vol and S-Vol. Works only, if P-Vol and S-Vol are in "pair" state.
<code>horctakeover -g pkga -S</code>	CA only, runs on S-Vol side. Brings S-Vol in read/write, even without having pair state before.
<code>pairresync -swaps -g CA1</code>	CA only, runs on S-Vol side. Swaps P-Vol and S-Vol
<code>pairresync -swapp -g CA1</code>	CA only, runs on P-Vol side. Swaps P-Vol and S-Vol

Configuration file definition for XP1024/XP128 with LUN Security enabled

The XP1024/XP128 allows multiple "Hostgroups" per FC Port if LUN Security is enabled. LUNs can be assigned independently in each hostgroup. That is, for example on one FC port there might be a LUN0 in hostgroup-a and a LUN0 on hostgroup-b. For this reason, the configuration file (which only contains the fields port, target and lun and NOT hostgroup) cannot be handled as for XP256/XP512/XP48 where those field distinctly refer to a certain LDEV. Instead the relation between the configuration file fields "target" and "LU" must be derived using a certain method described below:

1. Create "empty" raidmanager configuration files (only with fields HORCM_MON and HORCM_CMD, no entry for HORCM_DEV and HORCM_INST)
2. Start Raidmanager with these empty configuration files. You then can use the raidscan command which is needed in order to gather the Target and LU fields for the "real" configuration file.
3. You the run xpinfo and raidscan as shown below. Note that raidscan returns different values for "TARG" and "LUN" compared to xpinfo fields "Tgt" and "Lun". In order to create the "real" raidmanager configuration file, the values from raidscan must be used:

```
# xpinfo -i | grep CL1B
Device File          ALPA Tgt Lun Port  CU:LDev Type          Serial#
=====
/dev/rdisk/c13t0d0  27  00  00  CL1B  00:ff  OPEN-9-CVS  00060060
/dev/rdisk/c13t0d1  27  00  01  CL1B  00:5d  OPEN-9*3    00060060
/dev/rdisk/c13t0d2  27  00  02  CL1B  00:28  OPEN-9*5    00060060

# ls /dev/rdisk/* | raidscan -find | grep CL1-B
DEVICE_FILEUID      S/F    PORT    TARG LUN  SERIAL  LDEV  PRODUCT_ID
/dev/rdisk/c13t0d0  0      F      CL1-B  0    1    60060  255  OPEN-9-CVS
/dev/rdisk/c13t0d1  0      F      CL1-B  0    2    60060  93   OPEN-9*3
/dev/rdisk/c13t0d2  0      F      CL1-B  0    4    60060  40   OPEN-9*5
```

Note: there is also a tool called “mkconf.sh” in /HORCM/usr/bin which is able to simplify the creation of the configuration file creation.

Performance Aspects for XP Diskarrays

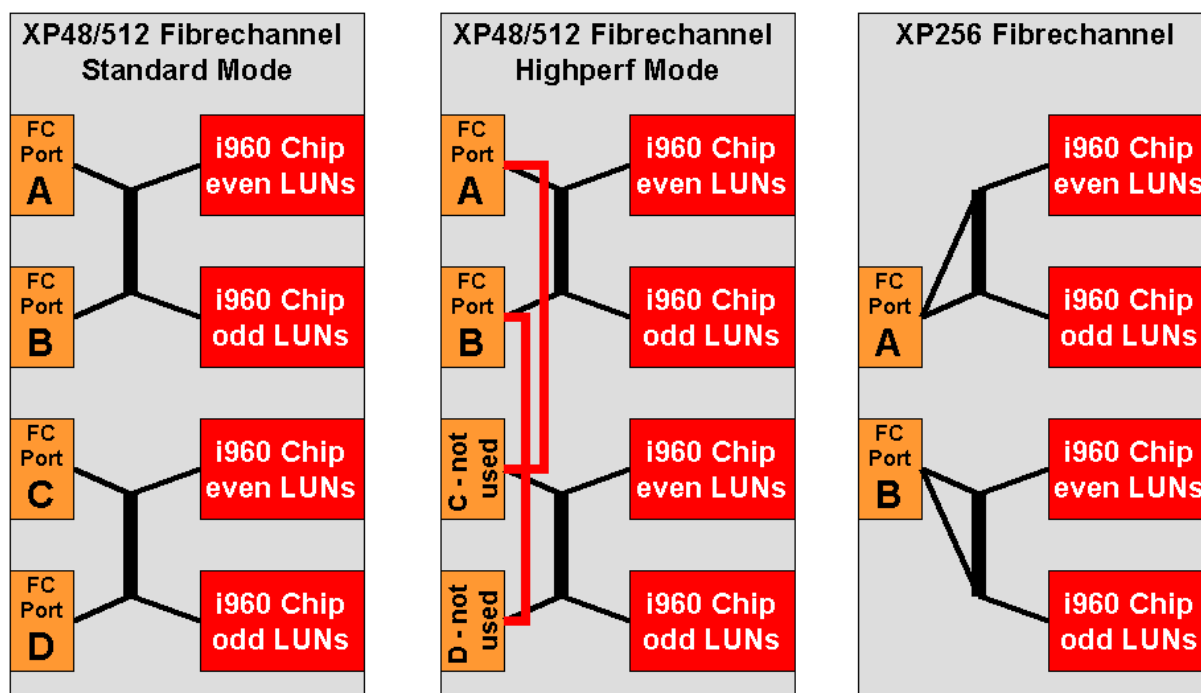
Many aspects influence the performance of an XP diskarray. There are some simple strategies to get the optimal performance out of an XP that all have the same idea: try to avoid bottlenecks in the dataflow between host and physical disk. Possible factors in this chain might be:

- the host itself: it is always advisable to have current fibre channel/scsi patches and a current version of the fibre channel driver installed. Besides that it is advisable to have not too many and not too less active LUNs per hostbusadapter. The reason is that HPUX creates a SCSI queue for each physical device. Performance maybe better, if more physical devices are used, because there is more “queue space” available. However, the amount of active LUNs (alternate paths do not count) should not exceed a certain limit. With the default q-depth used in HPUX, the following rule of thumb can be used: the number of active physical devices per hostbusadapter should be between 4 and 32.
- the fibre channel ports of the XP. The microprocessors used in the XP fc ports are working in a certain way explained later. Bad configurations of portmapping and setting primary and alternate paths in a certain way may lead to unbalanced workload of the microprocessors of the XP. Read below for detailed information.
- the cache size and amount and type of disks in the XP: increasing cache is always good for performance. Using more but smaller and faster disks also is a method of increasing the performance, unfortunately it may not always be an option because of financial reasons.
- avoiding hotspots is always the easiest and most effective way of getting the optimal performance. The idea is to have the same load to all disks in the XP. Avoiding heavy loads on single arraygroups allows a faster destaging from cache to disk thus leading to more free cache capacity which again leads to a better performance. The best way of reaching that goal is to stripe all data from the host across as many arraygroups as possible. Extent based striping is the favoured method for HPUX and described below.

Frontend Performance: XP48/XP512/XP256 fibre channel ports

This chapter mainly refers to the fibrechannelports of the “old” XP256/XP48/XP512 diskarrays. The new products XP128 and XP1024 have a dedicated processor for each fibrechannelport. However, the highperformance mode also exists for the XP1024/XP128 and works similar as in the “old” products.

A main factor for good IO performance of an XP Array are the fibre channel ports of the XP. Not knowing about the characteristics of the XP fibre channel ports may lead to configurations that use only “half the power” of these ports. Always two ports together use two i960 chips (for example: FC port CL1-A and CL1-B are sharing two i960 chips together). All “even” LUNs (e.g. c5t0d0) are using the first i960 chip, all “odd” LUNs (c5t0d1) are using the second i960 chip. The following pictures shows this XP fibre channel interface architecture. Depending on the slot that is used for the fibre channel interface, the ports maybe numbered differently (e.g. E,F,G,H for the second slot). “Low cost” XP512 fibre channel interfaces and all XP256 fibre channel interfaces only have 2 ports and 2 chips implemented.



The crucial point is that you only get full performance, if all i960 chips are used in a balanced manner. You can achieve this goal through an intelligent portmapping and LVM primary path distribution as shown in the “Even / odd LUN example” below.

Which ports should be used?

If you only need to connect two out of the four FC ports on one fibre channel board, you should make sure to use two ports that access different i960 chips (for example: use A and C, leave B and D). There is actually not much difference in IO performance if you connect only ports A and C versus all ports A,B,C,D to a host, because in both scenarios it is possible to access all four i960 chips.

“Highperf Mode”

If you want to use only one out of the 4 ports you might consider using a very special (and rarely used) configuration called “Highperf Mode” of the XP48/512. Enabling this mode will do the following:

- disable ports C and D.
- connect the i960 chips via a fc loop to ports A and B (careful! the port now doesn’t allow a “fabric point-to-point” configuration!)

Now it is possible to use all four i960 chips via only one physical connection from the host: connect either A or B to the host. Map even and odd LUNs to port A (upper two i960 chips are used) AND in addition map even and odd LUNs to port C (lower two i960 chips are used). Although port C is not connected to the host directly, these LUNs are visible because the XP internally creates a fc loop between port A and C.

Even / odd LUN example

A performance critical server is directly connected via two fibre channelcables to FC port CL1-A and FC port CL2-A of an XP diskarray. 4 LUNs are mapped for this server: LUN 00,

01, 02, 03 (devicefiles: CL1-A: c5t0d0, c5t0d1, c5t0d2, c5t0d3 and CL2-A: c6t0d0, c6t0d1, c6t0d2, c6t0d3).

LVM Layout 1 (non-optimal performance)

```
# vgdisplay -v /dev/vgdata | grep "PV Name"
PV Name          /dev/dsk/c5t0d0  => even, CL1-A
PV Name          /dev/dsk/c6t0d0  Alternate Link
PV Name          /dev/dsk/c6t0d1  => odd, CL2-A
PV Name          /dev/dsk/c5t0d1  Alternate Link
PV Name          /dev/dsk/c5t0d2  => even, CL1-A
PV Name          /dev/dsk/c6t0d2  Alternate Link
PV Name          /dev/dsk/c6t0d3  => odd, CL2-A
PV Name          /dev/dsk/c5t0d3  Alternate Link
```

i960 chip usage for LVM Layout 1:

- chip for even LUNs of port CL1-A: c5t0d0, c5t0d2
- chip for odd LUNs of port CL1-A: no active odd LUNs (alternate Links do not count!)
- chip for even LUNs of port CL2-A: no active even LUNs (alternate Links do not count!)
- chip for odd LUNs of port CL2-A: c6t0d1, c6t0d3

Conclusion for LVM Layout 1: two out of four i960 chips are not used at all in this non-optimal configuration.

It is very easy to correct the situation by exchanging primary and alternate links as shown in LVM Layout 2. The primary link is always the link that has been added first to the Volumegroup. Simply taking away this link (vgreduce) makes the formerly alternate link to the primary link. If you then add the formerly primary link (vgextend) it is now the alternate link. Here is the procedure on how LVM Layout 1 can be corrected to LVM Layout 2:

```
# vgreduce /dev/vgdata /dev/dsk/c6t0d1
# vgextend /dev/vgdata /dev/dsk/c6t0d1  => now alternate link!
# vgreduce /dev/vgdata /dev/dsk/c5t0d2
# vgextend /dev/vgdata /dev/dsk/c5t0d2  => now alternate link!
```

LVM Layout 2 (optimal performance)

```
# vgdisplay -v /dev/vgdata | grep "PV Name"
PV Name          /dev/dsk/c5t0d0  => even, CL1-A
PV Name          /dev/dsk/c6t0d0  Alternate Link
PV Name          /dev/dsk/c5t0d1  => odd, CL1-A
PV Name          /dev/dsk/c6t0d1  Alternate Link
PV Name          /dev/dsk/c6t0d2  => even, CL2-A
PV Name          /dev/dsk/c5t0d2  Alternate Link
PV Name          /dev/dsk/c6t0d3  => odd, CL2-A
PV Name          /dev/dsk/c5t0d3  Alternate Link
```

i960 chip usage for LVM Layout 2:

- chip for even LUNs of port CL1-A: c5t0d0
- chip for odd LUNs of port CL1-A: c5t0d1
- chip for even LUNs of port CL2-A: c6t0d2
- chip for odd LUNs of port CL2-A: c6t0d3

Conclusion for LVM Layout 2: All four i960 chips are used thus leading to an optimal

configuration for performance.

Bottom line:

Make sure to always use all available i960 chips of the XP fibre channel interface. Achieve this goal by using a balanced number of even and odd LUNs as LVM primary path for each XP fibre channel port.

Backend Performance: Extent-Based Striping

Host-based Striping can help to avoid hotspots in the XP diskarray. The idea behind striping in general is to spread the data as evenly as possible across the physical disks. The XP diskarray always groups 4 physical disks into an arraygroup where data is always striped across these 4 disks, so the aim for XP diskarrays is to spread the data across different *arraygroups*.

HP-UX allows two different ways of striping through LVM (refer to the [LVM chapter](#)):

“Normal striping” (`lvcreate -i ...`) can create a striping with small stripe size (4K - 32MB). The disadvantage of this method of striping is that if you started with a certain number of physical devices to stripe to, you always have to provide this number of physical devices in order to extend the lvol. Because of this disadvantage the preferred method of striping for the XP diskarray is *extent-based-striping* aka *pseudo striping*.

Extent-based-striping simply stripes each physical extent (PE) of an lvol to a different physical device. So the “stripe size” of extent-based-striping is always the size of a physical extent (usually 4MB). The big advantages of this method are:

- Extent-based-striping simply takes all available physical volumes (that are part of the VG plus are part of a physical volume group = are mentioned in `/etc/lvmpvg`) and stripes the data to it. So if you start with 5 physical volumes, the stripe will be created on these PVs. If these 5 PVs finally are filled up, you could add any number of PVs (>1) and continue the stripe on these volumes. The order LVM will use the PVs is the order of the PVs in `/etc/lvmtab` (see `strings /etc/lvmtab`).
- If you have only one physical volume to extend the lvol, extent-based striping cannot be continued for this lvol. In this case you can disable extent-based-striping on the particular lvol (using `lvchange -D n`) and extend this lvol with single physical devices. From that point on, extent-based-striping can only be re-enabled for the lvol using the force option “-D f”. The force option can also be used in order to migrate from a non-extent-based-striped-LVOL to an extent-based-striped LVOL.
- Extent-based-striping allows the use of **mirroring** (“normal” striping doesn’t). This is a very nice feature. One example for extent-based-striping and mirroring would be a data migration from a XP256 with 3 arraygroups (lets assume 1 PV per arraygroup) to a XP512 with 5 arraygroups (lets assume 5 PVs). If extent-based-striping was used for the 3 XP256 PVs, then simply add the 5 PVs of the XP512 (`vgextend ...`), enter the 5 new devices in a separate PVG (`/etc/lvmpvg`) and start mirroring the lvol (`lvextend -m 1 /dev/vgxx/lvolxx <xp512 pvs>`). After the mirror has been established you can remove the “old” XP256 devices (`lvreduce -m 0 /dev/vgxx/lvolxx <xp256 pvs>; vgreduce ...`) and the data migration is finished - the previously created stripeset of 3 PVs has been changed to a stripeset of 5 PVs - pretty cool!

Follow the steps below to create an extent-based striped lvol:

1) Choose devices located in different arraygroups

Of course striping only makes sense if the physical devices that are used for the stripeset are really located on different physical disk, so for the XP diskarray: different arraygroups. Lets assume that we have 4 arraygroups available and have mapped one LUN out of each arraygroup: c0t5d1 (AG1), c0t5d2 (AG2), c0t5d3 (AG3), c0t5d4 (AG4).

2) Create a physical volume groups (PVG)

After creating a VG as usual (`pvcreate`, `vgcreate...`) the file `/etc/lvmmpvg` must be created and must contain all PVs to stripe to:

```
# vi /etc/lvmmpvg
VG      /dev/vgtest
PVG     pvg_dummy
/dev/dsk/c5t0d1
/dev/dsk/c5t0d2
/dev/dsk/c5t0d3
/dev/dsk/c5t0d4
```

3) Create the Logical Volume:

The options for extent-based-striping are `-D y -s g`. Option `-r N` sets the correct mode for LVM bad-block-relocation and should always be used for diskarrays.

```
# lvcreate -D y -s g -r N -L 8000 /dev/vgdata

# lvsdisplay -v /dev/vgdata/lvol1 | more
--- Logical volumes ---
LV Name                /dev/vgdata/lvol1
...
Bad block              NONE
Allocation             PVG-strict/distributed
--- Distribution of logical volume ---
PV Name                LE on PV  PE on PV
/dev/dsk/c5t0d1        500      500
/dev/dsk/c5t0d2        500      500
/dev/dsk/c5t0d3        500      500
/dev/dsk/c5t0d4        500      500
--- Logical extents ---
LE    PV1                PE1    Status 1
00000 /dev/dsk/c5t0d1    00000  current
00001 /dev/dsk/c5t0d2    00000  current
00002 /dev/dsk/c5t0d3    00000  current
00003 /dev/dsk/c5t0d4    00000  current
00004 /dev/dsk/c5t0d1    00001  current
00005 /dev/dsk/c5t0d2    00001  current
00006 /dev/dsk/c5t0d3    00001  current
00007 /dev/dsk/c5t0d4    00001  current
...
```

4) Extension of the Logical Volume with other PVs

If it should be necessary to extend this logical volume with other physical devices, then all you have to do is to extend the VG with the new devices plus add them to the PVG in `/etc/lvmmpvg`.

For the following example lets assume that the four previously used PVs are “full” and 2

additional disks (c5t0d5, c5t0d6) should be used to extend the lvol with another 2000 MB:

```
# vgextend /dev/vgdata /dev/dsk/c5t0d5 /dev/dsk/c5t0d6

# vi /etc/lvmpvg
VG      /dev/vgtest
PVG     pvg_dummy
/dev/dsk/c5t0d1
/dev/dsk/c5t0d2
/dev/dsk/c5t0d3
/dev/dsk/c5t0d4
/dev/dsk/c5t0d5
/dev/dsk/c5t0d6

# lvextend -L 10000 /dev/vgdata/lvol1 /dev/dsk/c5t0d5 /dev/dsk/c5t0d6
```

NOTE: if you don't specify the PVs at the end of lvextend then simply all available PVs are used.

```
# lvsdisplay -v /dev/vgdata/lvol1 | more
--- Distribution of logical volume ---
PV Name          LE on PV  PE on PV
/dev/dsk/c5t0d1  500      500
/dev/dsk/c5t0d2  500      500
/dev/dsk/c5t0d3  500      500
/dev/dsk/c5t0d4  500      500
/dev/dsk/c5t0d5  250      250    <= new PV
/dev/dsk/c5t0d6  250      250    <= new PV
--- Logical extents ---
LE    PV1          PE1    Status 1
00000 /dev/dsk/c5t0d1 00000 current  <= Striping with 4 PVs
00001 /dev/dsk/c5t0d2 00000 current  starts here
00002 /dev/dsk/c5t0d3 00000 current
00003 /dev/dsk/c5t0d4 00000 current
....
01996 /dev/dsk/c5t0d1 00499 current
01997 /dev/dsk/c5t0d2 00499 current
01998 /dev/dsk/c5t0d3 00499 current
01999 /dev/dsk/c5t0d4 00499 current
02000 /dev/dsk/c5t0d5 00000 current  <= Striping with 2 new PVs
02001 /dev/dsk/c5t0d6 00000 current  starts here
02002 /dev/dsk/c5t0d5 00001 current
02003 /dev/dsk/c5t0d6 00001 current
....
```

5) Extension of the Logical Volume with a single physical device

If it should be necessary to extend this logical volume with only a single physical disk, then obviously extent-based-striping can not be continued. With “lvchange” you can switch to the non-striped mode. Note: once back in non-striped mode it is possible to get back to striped mode again using the option “-D f”.

For the following example lets assume that the previously used PVs are “full” and a single additional disk (c5t0d7) should be used to extend the LVOL with another 2000 MB:

```
# lvchange -D n /dev/vgdata/lvol1

# lvextend -L 12000 /dev/vgdata/lvol1 /dev/dsk/c5t0d7

# lvsdisplay -v /dev/vgdata/lvol1 | more
Allocation          PVG-strict
```

```

--- Distribution of logical volume ---
PV Name          LE on PV  PE on PV
/dev/dsk/c5t0d1  500      500
/dev/dsk/c5t0d2  500      500
/dev/dsk/c5t0d3  500      500
/dev/dsk/c5t0d4  500      500
/dev/dsk/c5t0d5  250      250
/dev/dsk/c5t0d6  250      250
/dev/dsk/c5t0d7  500      500      <= new PV
--- Logical extents ---
LE    PV1          PE1    Status 1
00000 /dev/dsk/c5t0d1  00000 current <= Striping starts here
00001 /dev/dsk/c5t0d2  00000 current
00002 /dev/dsk/c5t0d3  00000 current
00003 /dev/dsk/c5t0d4  00000 current
....
02496 /dev/dsk/c5t0d5  00248 current
02497 /dev/dsk/c5t0d6  00248 current
02498 /dev/dsk/c5t0d5  00249 current
02499 /dev/dsk/c5t0d6  00249 current <= Striping ends here
02500 /dev/dsk/c5t0d7  00000 current <= new PV starts here
02501 /dev/dsk/c5t0d7  00001 current
02502 /dev/dsk/c5t0d7  00002 current
02503 /dev/dsk/c5t0d7  00003 current
....

```

Additional Information

XP diskarray family:

<http://h18006.www1.hp.com/storage/array/systems.html>

XP Storage Software:

http://www.hp.com/products1/storage/products/disk_arrays/xpstorage/sw

Network Array Solutions Lab - Documents, Configurations:

<http://xpstorage.corp.hp.com> (HP internal)

XP Support Web Site - Documents, Whitepapers:

<http://tce-web.boi.hp.com/xpsupport> (HP internal)

XP Performance Estimator Tools:

<http://nss-xpperf-tools.corp.hp.com> (HP internal)

Stress Free Central USA - XP Support:

<http://quantum.atl.hp.com> (HP internal)

Stress Free Central Europe - XP Support:

<http://mcscwww.brussels.hp.com/XPsfc> (HP internal)

Configuration Guide "Spock" - Single Point of Configuration Knowledge:

<http://turbo.rose.hp.com/spock> (HP internal)