SASAKI informática
Paulo Sasaki & Consultores Associados
desde 1995

SUBSISTEMAS AVANÇADOS DE ARMAZENAMENTO

CÓDIGO DO DOCUMENTO : SSK0151 DE SETEMBRO DE 2004

Segunda Revisão - 11/2009

WEB: WWW.PAULOSASAKI.COM.BR

Índice

I - Introdução

As unidades de disco rígido fazem parte dos sistemas de processamento desde os idos de 1957, quando a IBM anunciou seu primeiro RAMAC ("*Random Access Method of Accounting and Control")*, modelos 305 e 605, que comportavam espantosos *5 MegaBytes* de dados em seus 50 discos de mais de meio metro de diâmetro ("two-feet disks").

Crescendo em capacidade, velocidade e confiabilidade, os modernos subsistemas de armazenamento compõem o coração de qualquer centro de processamento de dados atual. Podendo atingir capacidades acima de 80 TeraBytes por subsistema, e conectando-se a dezenas de sistemas simultaneamente, estas máquinas são, hoje, as fiéis depositárias do maior patrimônio de qualquer empresa: suas informações.

Ao longo dos anos, necessidades crescentes demandaram novas soluções em paralelismo (*Data-sharing, Sysplexes e Clusters*), conectividade (ESCON, FICON, FC, SATA, p.ex.) e integração entre as plataformas por exemplo. Isto fez com que, sua complexidade acompanhasse o crescimento, quase exponencial, de sua capacidade. Além disso, a constante pressão por redução de custos forçou os fabricantes a buscarem novas soluções, por exemplo trocando os tradicionais *SLEDs* (*"Single Large Expensive Disks")*, por simulações *RAID*.

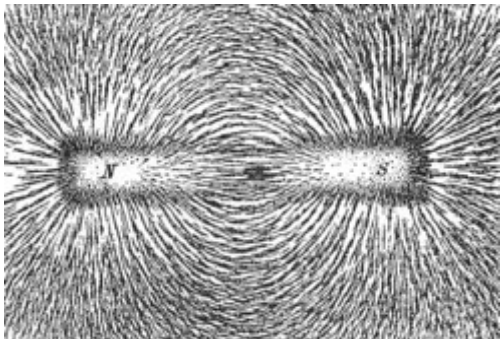
Alguns destes processos introduziram no mundo do armazenamento, variáveis com as quais os profissionais da área não estavam habituados anteriormente (Prioridade SCSI ?? Mudanças em taxas de transferência por alteração de densidade ??).

A gerencia, modelagem e avaliação de desempenho tornaram-se gradativamente mais complexas, chegando ao ponto, hoje, de poucos serem os especialistas realmente capacitados a darem um diagnóstico confiável, ou a sugerirem um modelo realmente eficaz.

A proposta deste trabalho é trazer aos profissionais da área de armazenamento, planejamento de capacidade e análise de desempenho, informações detalhadas sobre os principais subsistemas atuais, seus componentes e diferentes comportamentos, bem como guiá-los desde os métodos tradicionais de avaliação de desempenho até novas propostas, efetivas e mais adequadas aos dias de hoje.

II – Componentes

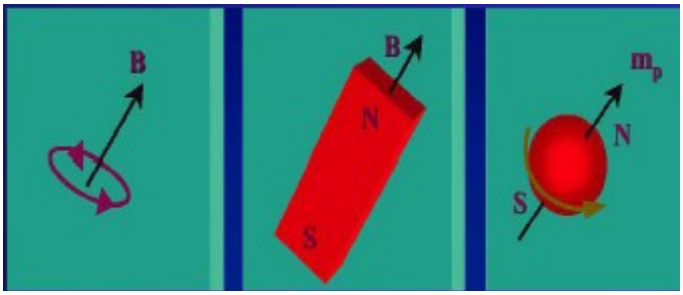
1 – Gravação magnética



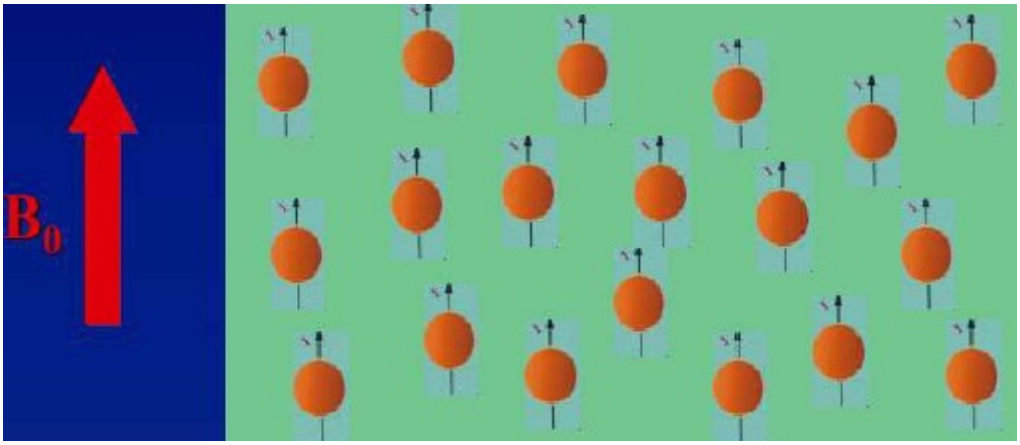
O conhecimento do magnetismo é bastante antigo, sendo suas primeiras referências encontradas já em textos de filósofos gregos datados de 600 AC. Também desta época datam as primeiras referencias às cargas elétricas, desenvolvidas por atrito. Aristophanes descreve a capacidade desenvolvida pelo âmbar quando esfregado com couros ou outros materiais fibrosos, de atrair materiais como penugens. A própria palavra "eléktron" significa âmbar em grego.

As propriedades de imãs naturais já eram conhecidas e utilizadas em passado tão remoto quanto 1175 DC, descritas pelo monge inglês Alexander Neckem, primeira referência conhecida às bussolas de navegação. No século XVIII já havia a fabricação de magnetos artificiais, induzidos por corrente elétrica, para uso industrial e científico.

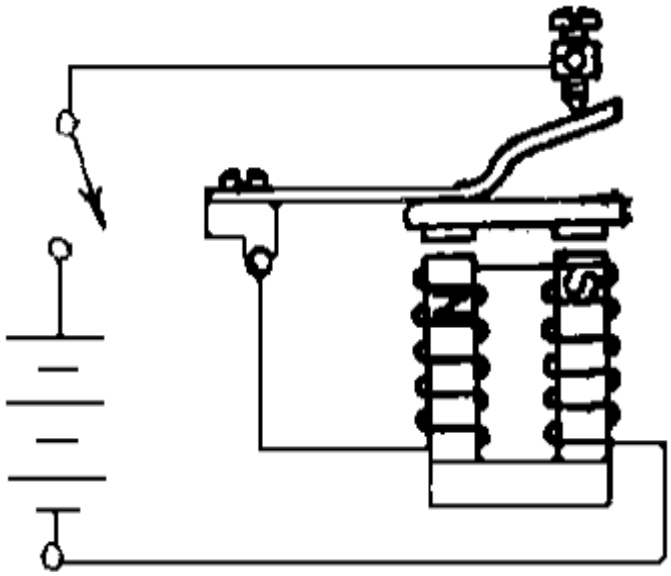
Toda carga elétrica, ao girar em torno de seu eixo (spin), produz um campo magnético. Seu efeito é tão maior quanto maior for o alinhamento das diversas cargas do elemento considerado:



O efeito de polarização magnética é obtido através da aplicação de um campo magnético externo ao material, que realinha os "spins" de suas cargas elétricas, produzindo assim um campo detectável (e utilizável), externamente:



Um tal campo externo pode ser produzido por magnetos naturais, ou por correntes elétricas circulando ao redor do material a ser induzido (eletro magnetismo), conforme demonstrado já em 1820 pelo físico Hans Christian Oersted. Tal mecanismo serviu de base para inúmeras aplicações, como os eletro-imãs utilizados por Samuel Morse no telegrafo.



Os materiais mais facilmente magnetizáveis são os ferromagnéticos, de estrutura cristalina, mais sensíveis ao alinhamento dos spins de suas cargas elétricas à de um campo externo. O nome "ferromagnético" deriva de seu efeito mais óbvio de atrair o ferro em sua direção.

A primeira gravação de voz humana em meio eletro magnético foi conduzida em 1898, e em 1927 a fita magnética, baseada em óxido ferroso era introduzida simultaneamente nos Estados Unidos e Alemanha, sendo popularizada já em 1947 pela 3M. Sua utilização em processamento de dados antecede os discos rígidos, sendo ainda reconhecida como elemento principal do ambiente TOS ("Tape Operating System"), dos sistemas IBM/360 de 1965 (8 anos após o anúncio dos primeiros discos rígidos).

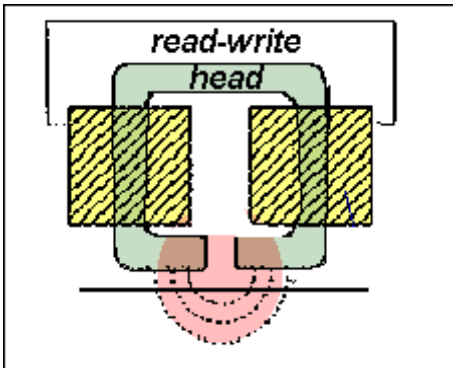
2 – Discos Rígidos

2.1 - Introdução

Os discos rígidos ainda constituem o elemento fundamental de todo e qualquer subsistema de armazenamento. Avanços tecnológicos permitiram ao longo dos anos que eles armazenassem quantidades cada vez maiores de dados, e que os mesmos fossem acessados em tempos cada vez menores. Um bom exemplo do mercado atual são os discos SATA 7.2 Krpm com até 750 Gbs, ou FC-146 a 15Krpm. É normal encontrar-se discos disponíveis ao mercado de varejo, modificados, ocupando o centro de subsistemas de armazenamento comerciais de outras empresas.

A mudança conceitual entretanto, foi pequena. Ainda estamos falando de "pratos" metálicos, interligados pelo centro por um eixo comum, girando a diferentes velocidades, recobertos de material magneto-resistivo, no qual circuitos ou bobinas gravam pequenos imãs, cuja polaridade ou presença indica 0´s e 1´s. Portanto, a velocidade de rotação e a de acesso do braço atuador, os conceitos mecânicos utilizados para a análise dos discos antigos continuam valendo, mesmo que parcialmente. Mas, algumas das mudanças tecnológicas introduzidas alteram características importantes, influenciando o resultado final desejado.

2.2 – Gravação magnética



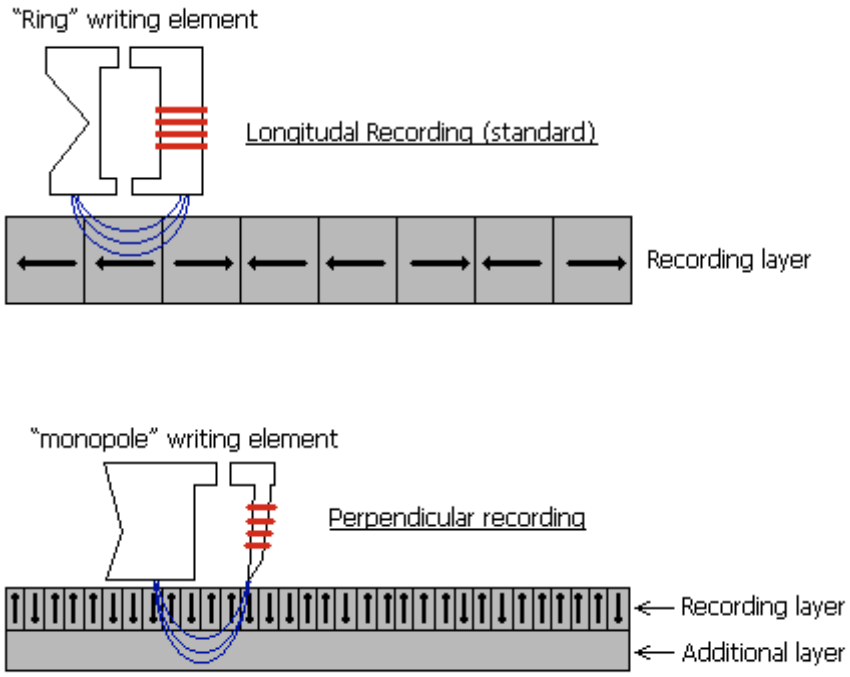
O esquema acima apresenta o desenho básico de uma cabeça de leitura-e-gravação, na qual bobinas envolvem o material em forma de "C", e que quando atravessadas por uma corrente elétrica criam um campo magnético que atinge a superfície do disco. Para leitura, o espaço entre os terminais "sente" a direção do campo presente na superfície, induzindo o campo em um ou outro sentido (polarização), que é detectado e amplificado na cabeça de leitura. Dessa forma, "0" e "1" podem ser gravados (por exemplo), com polarizações N-S ou S-N. Sua forma final no disco depende de sua forma de codificação.

Na prática, as cabeças de R/W não detectam realmente a polarização do sinal no disco, mas sim a "reversão" de sua polaridade (a transição entre N/S ou S/N), por este processo apresentar uma maior facilidade e confiabilidade. Além disso, há a necessidade de sincronização dos dados (gravação de 1.000 bits de "0"s pode apresentar dificuldades para detectar, digamos, o final do 895 e início do 896), e a separação dos campos (os mesmos 1.000 bits de zeros, polarizados no mesmo sentido, não criariam 1.000 pequenos campos N-S, mas sim um único, orientado para o mesmo lado). Todos estes aspectos criam a necessidade de

técnicas de codificação dos dados que os satisfaçam, garantindo a integridade e recuperabilidade das informações armazenadas desta forma.

As cabeças R/W são, atualmente, montadas como entidades separadas, nas tecnologias chamadas MagnetoResistive (MR) ou Giant MagnetoResistive (GMR). Durante a leitura, os pequenos sinais magnéticos da superficie altera a resistividade magnetica da cabeça de leitura, o que é indicada na forma de alterações na corrente de suas bobinas, amplificada e identificada de acordo.

Ainda nesta área, o avanço tecnológico mais recente é a *gravação perpendicular,* na qual a magnetização não mais se da ao longo da superfície do disco, mas sim em sua profundidade, criando assim reversões mais compactas, o que permitem um grande aumento de densidade nos discos (até 1 TB, atualmente).



2.3 – Codificação de Gravação

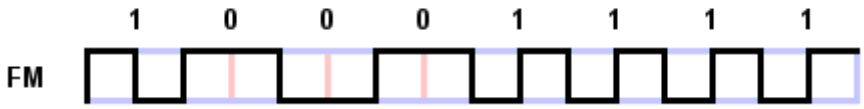
Dados os aspectos mencionados acima, a gravação digital de dados em meio magnético deve ser capaz de:

1. Registrar o dado através de uma mudança de polaridade (“flux reversals”)
2. Manter uma forma de sincronismo que permita a identificação de cada “bit”
3. Manter o número de polaridades iguais consecutivas ao mínimo possível

A primeira codificação utilizada para atender a esses requisitos foi a modulação de freqüência.

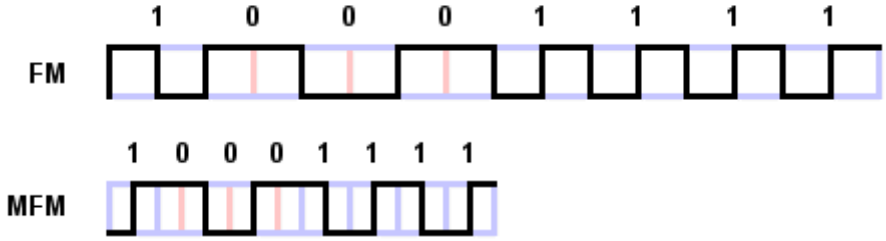
Frequency Modulation (FM) e Modified Frequency Modulation (MFM)

Na codificação FM, um digito “1” é identificado por 2 reversões de fluxo (N-S/S-N/S-N), na qual as “/” indicam o ponto de reversão (R), e o “0” é gravado como uma reversão e uma não-reversão (N):



O nome (FM), vem do fato dos 1´s terem o dobro da freqüência (2 reversões), quando comparados com os 0´s. Esta era a codificação utilizada nos antigos disquetes flexíveis, bem como nos primeiros discos rígidos, sendo posteriormente substituída pela MFM ("Modified Frequency Modulation").

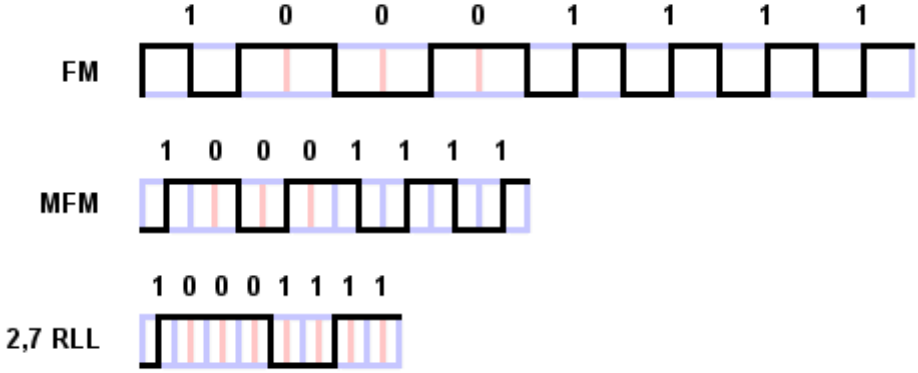
A modificação introduzida na MFM é que nela não se usa mais a inserção da reversão de fluxo de clock no início de cada bit, mas somente entre 0´s consecutivos (no caso de bits "1", já havia a reversão do próprio bit):



Com a redução do número de reversões para uma mesma área, o dobro de dados podia ser armazenado sob este esquema, quando comparado com o FM original.

RLL (Run Length Limited)

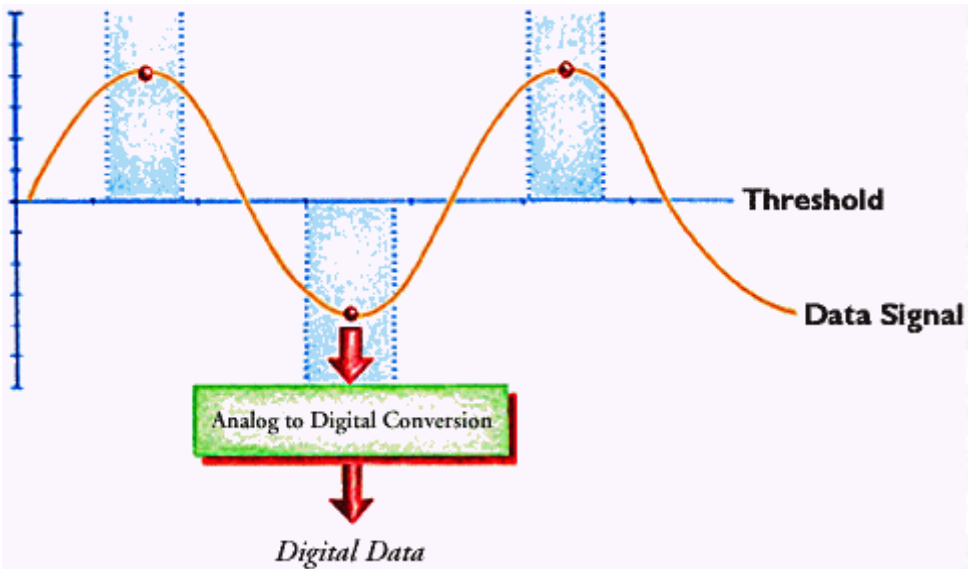
O desenvolvimento de cabeças e meios mais eficientes possibilitou também o surgimento de esquemas de gravação mais sofisticados, como o RLL. Nele, os bits são considerados em grupos, e a especificação vem na forma de valores RLL(X,Y), no qual X é o comprimento da corrida ("run length"), também identificado como o menor valor entre 2 reversões, e Y é o limite de corrida ("run limit"), ou maior valor entre reversões.



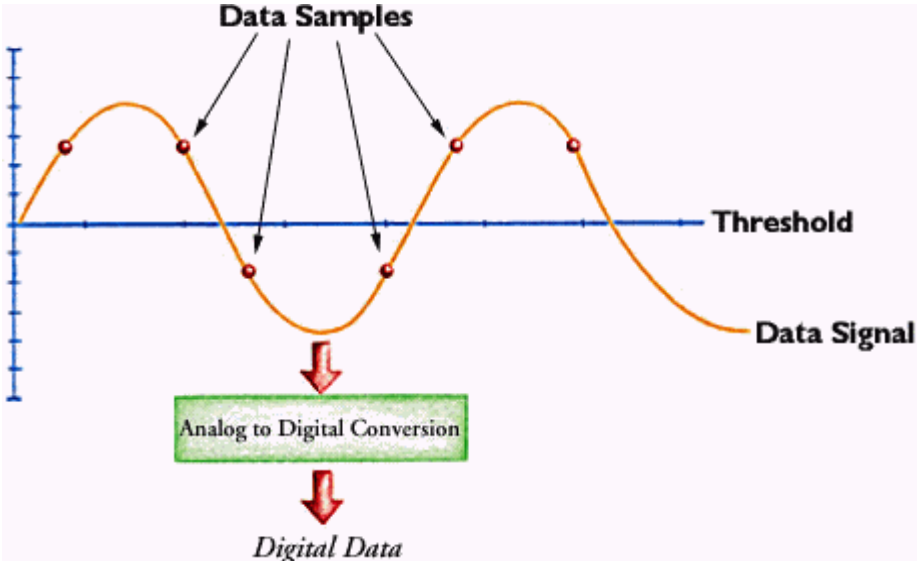
Sempre há reversões de fluxo para dígitos 1, portanto, na codificação acima (2,7), há uma reversão para indicar o primeiro "1", que se mantem igual (run limit=7), até o próximo "1". Depois disso se mantem igual por 2 dígitos 1 e reverte para o terceiro consecutivo (run length=2). Este é o padrão usado hoje para codificação de dados em discos modernos, e distoa bastante do conceito que se tem, a princípio, de que a cada "bit" corresponde um "imã" na superfície do disco, variando sua orientação de acordo com seu valor.

Partial Response, Maximum Likelihood (PRML)

As primeiras cabeças de leitura varriam a superfície do disco detectando as mudanças de fluxo que, ao atingirem um determinado valor limite ("threshold"), disparavam a indicação de presença da reversão. Esta técnica é conhecida como *detecção por picos* ("peak detection"). O problema com ela é que conforme aumenta a densidade de gravação, aumenta também a tendência do sinal de uma reversão influenciar a outra, criando ruido e dificultando sua interpretação.



Em função disso, e para permitir o aumento da densidade, uma nova técnica chamada de PRML foi desenvolvida, na qual algoritmos são usados para varrer os sinais de entrada conforme recebidos ("partial response"), e indicar a maior possibilidade vislumbrada ("maximum likelihood")



Mesmo que a principio um tal algoritmo soe como bastante sujeito a erros, o PRML mostrou-se bastante confiável, tornando-se o padrão para a industria por muito tempo, sendo posteriormente substituído por uma evolução do mesmo, chamado de EPRML ("Extended Partial Response, Maximum Likelihood").

2.4 – Braços atuadores

Para se movimentar sobre as superfícies dos diversos discos, as cabeças de leitura e de gravação são montadas juntas em um componente chamado "slider", o qual é preso a um braço atuador. Estes braços servem a duas funções básicas: comprimir o "slider" contra a superfície do disco, mantendo-o assim tão próximo da mesma quanto possível, e deslocar-se mecanicamente. Uma analogia sobre o comportamento mecânico causado pela pressão contra o slider seria a de um Boeing 747 voando a meia polegada do solo, e contando as folhas de grama em sua passagem (bits).

A unidade IBM 350, anunciada em 1957, é conhecida como o primeiro disco rígido introduzido no mercado. Possuía um total de 50 discos de 24 polegadas, somando 100 superfícies de acesso, montados em um conjunto que girava a 1.200 rpm, transferindo um total de 8.8 KB/s (7-bit bytes). A mesma possuia 2 braços atuadores, que primeiro deslocavam-se verticalmente (selecionando a superfície a ser acessada), e depois horizontalmente, posicionando-se sobre a trilha. A movimentação era controlada por servo-mecanismos, e acionada por um sistema hidráulico-mecânico. O movimento dos conjuntos é chamado de "seek".

Quatro anos mais tarde (em 1961), a IBM anunciava a unidade 1301, que diferia em desenho, de seus predecessores, por utilizar um conjunto (braço+cabeças), para cada superfície (evitando assim, o deslocamento horizontal). Tempos típicos de acesso nessa época variavam entre 300 e 900 mS (quase 1 segundo de "seek-time").

Atualmente, o conjunto atuador é acionado lateralmente por sistemas elétricos, o que lhes valeu o apelido inicial de "winchesters" referência ao movimento feito pelo usuário para recarregar a câmara de disparo destes rifles americanos.

2.5 – Formatações de superfície

Os primeiros discos magnéticos permitiam o que era conhecido como "low-level formating", ou LLF, que consiste, basicamente, na criação de entidades chamadas de SETORES, de 512 bytes de comprimento, ao longo de cada uma das trilhas do

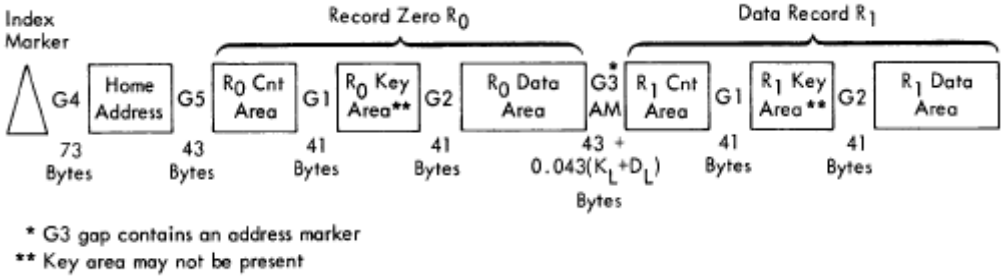
disco (posição estacionária do braço de acionamento), iniciando a partir de um "index-mark", ou marca lógica de início da mesma. Comumente, a mesma era feita com a gravação de zeros binários em cada setor, opção conhecida como "zero-fill".

A partir de meados dos anos 90, as LLFs tornaram-se progressivamente complexas, com o advento de novas codificações como o RLL, e o uso de densidades de gravação (e portanto, número de setores), variável. Por essa razão, ao invés de tentar adaptar a programação e os sistemas para lidarem com isso, as LLFs foram restritas às executadas nas próprias fábricas. Apesar disso, o termo LLF ainda é usado hoje para utilitários que gravam zeros em todas as localizações endereçáveis do disco (o que é, na verdade, uma *reinicialização* do mesmo).

A LLF é definida em oposição a HLF, ou "high-level format", utilizada para criar as áreas de controle sobre a superfície, utilizadas por cada sistema de arquivos, manipulados pelo sistema operacional ao qual o disco vai servir.

Formatação Mainframe

Uma das mais antigas formatações HLF é a utilizada pelos ambientes IBM Mainframe em seus discos. Chamada de CKD (Count-Key-Data), este formato cria entidades chamadas de registros físicos ao longo da trilha, com um campo contador (Count), indicando seu número, um campo chave (Key), para pesquisa e o campo de dados (Data), de tamanho variável. A figura abaixo foi extraída do manual de referência do IBM-2314, e ilustra o uso deste formato:



O Index Marker indica o começo da trilha, seguido pelo Home Address record (contem informações sobre o endereço da trilha, áreas com problema, etc), R0 com dados sobre o número de registros, e a partir daí, os dados (registros) propriamente ditos. Entre cada bloco são inseridos seqüências padrão (chamados de IBGs, ou Inter-block gaps), para espaçamento e identificação.

Para uso sob OS/VS2, MVS, OS390 e zOS (evoluções do mesmo sistema), os primeiros registros apontam à uma área chamada de VTOC ("Volume Table of Contents"), que contem blocos chamados de DSCB ("Dataset Control Blocks"), os quais descrevem o conteúdo do volume, endereço de início de cada arquivo, na forma CCHHR ("Cylinder-Head-Record"), seu espaço disponível e assim por diante.

Formatações OpenSys

Para uso em sistemas abertos, e sobre a formatação de baixo nível (LLF), o esquema mais amplamente usado define um MBR ("Master Boot Record"), como o conteúdo do primeiro setor do disco (Setor 0), conforme abaixo:

Structure of a Master Boot Record

Address		Description		Size in bytes
Hex	Dec			
0000	0	Code Area		max. 446
01B8	440	Optional Disk signature		4
01BC	444	Usually Nulls; 0x0000		2
01BE	446	**Table of primary partitions** (Four 16-byte entries, IBM Partition Table scheme)		64
01FE	510	55h	MBR signature; 0xAA55	2
01FF	511	AAh		
MBR, total size: 446 + 64 + 2 =				**512**

Originalmente, as primeiras linhas de código do sistema operacional eram carregados nos primeiros 440 bytes deste registro (Code Area), e eram responsáveis por iniciar as operações que carregariam o restante do sistema. Com sua evolução, a presença de partições na formatação da unidade apontavam para "discos lógicos" na mesma, cada qual podendo conter sua própria versão de sistema operacional, com sua posição física apontada a partir da MBR original:

Layout of one 16-byte partition record

Offset	Description
0x00	(1 byte) Status (0x80 = bootable, 0x00 = non-bootable, other = malformed)
0x01	(3 bytes) Cylinder-head-sector address of the first sector in the partition
0x04	(1 byte) Partition type
0x05	(3 bytes) Cylinder-head-sector address of the last sector in the partition
0x08	(4 bytes) Logical block address of the first sector in the partition
0x0C	(4 bytes) Length of the partition, in sectors

Neste caso, o setor apontado a partir de 0x01 conteria também uma MBR, desta vez local, e com código de carga em seus primeiros bytes.

A partir da MBR, cada ambiente cria um sistema de tabelas próprio, (como a FAT para os primeiros sistemas DOS e Windows, NTFS para o Win NT, ou o UFS para sistemas "Unix-like". Apesar das diferenças em formato, todos contem dados sobre o volume, identificadores para o sistema, e endereços físicos do início de cada arquivo gravado no volume representado.

2.6 – Taxas de Transferência

Durante sua operação, a velocidade de rotação (em rotações por minuto, ou RPMs) de um disco mantém-se constante (7.200, 10K e 15K sendo as mais comuns atualmente). Tomando-se como exemplo o de 10krpm, o disco ou o conjunto dos discos completa uma rotação a cada 6 mS, normalmente definindo-se em 3mS o período médio de rotação.

A figura abaixo mostra as especificações de alguns discos FC da Seagate:

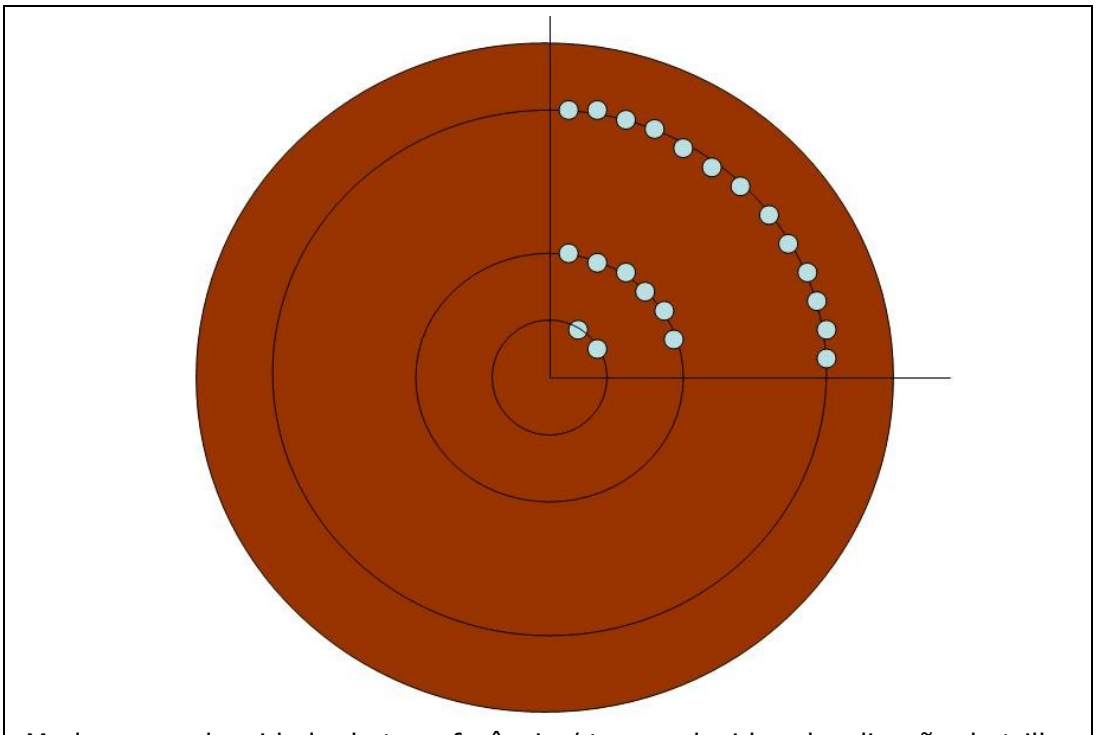
4.1 Internal drive characteristics

	ST3146807FC	ST373307FC	ST336607FC	
Drive capacity	146.8	73.3	36.7	Gbytes (formatted, rounded off value)
Read/write data heads	8	4	2	
Bytes per track	368	368	368	Kbytes (average, rounded off values)
Bytes per surface	18,350	18,350	18,350	Mbytes (unformatted, rounded off value)
Tracks per surface (total)	49,855	49,855	49,855	Tracks (user accessible)
Tracks per inch	64,000	64,000	64,000	TPI
Peak bits per inch	570	570	570	KBPI
Internal data rate	475-841	475-841	475-841	Mbits/sec (variable with zone)
Disc rotation speed	10,008	10,008	10,008	rpm (± 0.5%)
Avg rotational latency	2.99	2.99	2.99	msec

Características internas de drives Seagate FC

Como mostrado no exemplo acima, a velocidade de rotação do modelo ST3146807FC é de 10.008 rpm, sendo sua latência média, apresentada como metade do período (2,99mS). O número de Kbytes por trilha é apresentado como um valor médio (368).

Um aspecto interessante aqui é que, mantida a velocidade de rotação (10Krpm), e adensidade de Kbits por polegada (570), mas alterando-se a freqüência de gravação, conclui-se que as trilhas mais externas (de maior diâmetro) podem armazenar uma quantidade maior de dados que as mais internas, como mostra a figura abaixo :



Mudança na densidade de transferência / tempo devido a localização de trilha

Esta técnica é chamada de "zoned-recording", e é a causa dos valores de taxas de transferência diferentes (conforme mostrados na Figura 1) e apresentados como uma faixa (Internal Data Rate 475-841 Mbps).

As menores taxas de transferência ocorrem enquanto as cabeças de leitura e gravação estão posicionadas sobre as trilhas mais internas (mais próximas ao centro) e, as maiores sobre as mais externas. Valores médios podem ser utilizados, quando é necessário um único número (algumas modelagens podem exigir isso).

Ainda é importante notar que a variação é grande. No caso do exemplo acima, os cilindros mais externos transferem dados a quase 1.8 vezes a velocidade dos mais internos. Praticamente o dobro da velocidade, no mesmo disco físico.

2.7 – Acesso Mecânico ("seek time")

As trilhas e, conseqüentemente, os cilindros, são formados por posições estacionarias do braço atuador, que sustenta as cabeças de leitura/gravação, sobre a superfície dos discos. Logo, o tempo de deslocamento mecânico do atuador também afeta a velocidade com que os dados são gravados ou recuperados. A figura abaixo mostra as especificações do modelo de exemplo:

4.2.1 Access time

		Including controller overhead [1, 2] (msec)		Not including controller overhead [1,2] (msec)	
		Read	**Write**	**Read**	**Write**
Average	Typical[3,4]	4.90	5.50	4.70	5.30
Single track	Typical[3,4]	0.55	0.75	0.35	0.55
Full stroke	Typical[3,4]	9.20	9.70	9.00	9.50

1. Execution time measured from receipt of the FCP Command to the FCP Response.
2. Assumes no errors and no sector has been relocated.
3. Typical access times are measured under nominal conditions of temperature, voltage, and horizontal orientation as measured on a representative sample of drives.
4. Access time = controller overhead + average seek time.
   Access to data = controller overhead + average seek time + latency time.

Especificações de seek

Conforme mostrado acima, a movimentação do braço atuador de uma trilha para a trilha vizinha (anterior ou posterior), é chamado de *"Single track seek"*. Este modo de movimentação é mais lento que, por exemplo, o *"Full stroke seek"*, ou a movimentação desde a primeira até a última trilha do disco. Para se ter certeza, basta multiplicar o tempo típico de um *"single track"* (0,55mS) pelo número de trilhas em uma superfície (49855), e comparar o resultado com o valor médio do *"full stroke"* (9,2mS).

Isto se deve ao mesmo comportamento observável nos elevadores modernos. Quando se deslocando, por exemplo, do primeiro ao último andar, os motores recebem uma potência maior (ou pulsos mais freqüentes), do que quando se deslocam para próximo andar. De um modo geral, pode-se dividir as fases de energização do mecanismo atuador em 5 fases distintas :

- Estacionário (*"track-following"*) : o mecanismo recebe pulsos de curta duração, negativos ou positivos, apenas para manter o alinhamento com a trilha atual, de acordo com as informações da superfície de servo
- Carga plena (*"full-drive"*) : período no qual toda a potencia é aplicada ao mecanismo, para acelerá-lo ao máximo (controlado via servo), no menor tempo possível
- Manutenção (*"coarse"*) : manutenção da velocidade máxima, através de eventuais pulsos de grande potencia, porém pouca duração. A idéia aqui é apenas manter, e não acelerar.
- Frenagem (*"full-reverse"*) : pulsos de grande intensidade e freqüência, aplicados no sentido *OPOSTO* ao utilizado durante a aceleração. O objetivo aqui é frear o conjunto, devido à aproximação da localidade desejada.
- Aproximação : semelhante ao *"coarse",* porém já em velocidade mais baixa .

E, voltando ao Estacionário novamente, já posicionado sobre a trilha desejada.

N.A.: alguns fornecedores incluem uma fase adicional chamada "settle time" em sua especificação de seek. Este é o tempo que o conjunto mecânico leva para estabilizar na posição após sua movimentação, sendo atualmente da ordem de 0,1 mS.

Entretanto, quando a movimentação desejada é relativamente pequena, a fase de aceleração tem menor duração ou nem sequer existe, causando a grande diferença nos tempos apresentados na tabela da Fig.3. A decisão de quando utilizar maiores velocidades ou não, muda de modelo para modelo.

2.8 – Simulações de trilha, cilindros e volumes

Aliado às características de deslocamento mecânico, apresentadas no item anterior, um outro fator que influencia o comportamento final dos volumes, como vistos pelas partições lógicas, são as formas que estas simulações tomam.

Os sistemas MainFrame têm, ao longo dos últimos anos, utilizado o formato (ou "geometria") de trilha dos antigos IBM-3390, dando ainda bastante preferência ao seu modelo 3 (2.838 GBytes, 3339 cilindros, 15 trilhas / cilindro, 56664 bytes por trilha, formatação CKD).

Conforme mostrado na tabela da Fig.1, uma trilha do disco utilizado aqui como exemplo comporta até 368 Kbytes em média (ver 1.2 – Taxas de transferência). Isto chega a ser mais de 6.5 vezes o tamanho de uma trilha de um volume 3390.

Assumindo-se que nenhum esquema RAID seja utilizado, isto significa que várias trilhas lógicas do modelo simulado acabam por localizar-se fisicamente em uma única trilha física do disco rígido real.

Além disso, o mapeamento de dados é basicamente o definido no protocolo SCSI, com tamanhos de bloco ajustáveis entre 512 e 528 bytes, o que faz com que as trilhas dos volumes lógicos sejam fisicamente definidas em aproximadamente 110 blocos. As unidades de transferência, ou *"frames"*, podem variar de 256 a 2112 bytes. Portanto, a transferência de uma única trilha lógica pode implicar em quase 27 operações SCSI ou FC-AL para se completar. O mapeamento entre o número do bloco (*"lsid"*) e a trilha simulada correspondente fica por conta dos algoritmos de controle implementados no subsistema.

Conseqüentemente, os cilindros do volume lógico também não mantêm mais correspondência alguma com sua localização física final. Atualmente, reservar um único cilindro para evitar movimentações do atuador não funciona mais.

Um outro fator importante a se considerar nesta área é o mapeamento ou distribuição dos volumes lógicos no espaço físico disponível. Apesar de uma forte preferência pelos modelos IBM/3390-3, principalmente com o intuito de evitar concorrências em dados de alta prioridade, é importante manter-se em mente que no final, o disco físico é quem pode ser engargalado, independentemente do modelo de simulação utilizado.

Somente a titulo de exemplo, o disco de 146 Gbytes poderia conter 51 unidades 3390-3 (ainda sem quaisquer considerações sobre esquemas de proteção). Caso todos os 51 volumes apresentem altas taxas de utilização, o resultado final será a contenção apresentada por um único e fictício "3390-146", acrescido da maior dificuldade de gerenciamento.

É claro que a dispersão dos dados de uma instalação entre diversos volumes menores diminui a contenção interna (entre as aplicações) por um único volume. Mas o fato dos volumes lógicos atualmente compartilharem os recursos mecânicos com vários outros, aliado a novas facilidades nos sistemas IBM-OS/390 e IBM-z/OS, como o PAV-MA *("Parallel Access Volume-Multiple Allegiance")* apontam em outra direção.

Embora mais próximas da realidade física, as mesmas traduções de endereço acontecem também em ambientes OpenSys, nos quais os endereços de cada arquivo na superfície do volume lógico deve ser traduzido para seu posicionamento físico final no volume que o contem (seja por particionamento, utilização de metavols nos gerenciadores de volumes, grupos RAID ou outros)

2.9 – Limitações Físicas

Assim como quaisquer dispositivos mecânicos, uma unidade de disco rígido também apresenta suas limitações. Uma maneira simples de defini-las seria partir das especificações mostradas nas figuras acima.

A rotação de um disco de 10Krpm faz com que uma volta seja completada em, no máximo, 6 mS, e o maior deslocamento do atuador levaria entre 9.2 e 9.7 mS. Portanto, no pior caso, somando-se o maior seek com o maior atraso rotacional, teríamos um tempo mecânico de aproximadamente 16 mS para uma operação. De acordo com a Fig.1, a pior taxa de transferência de dados seria de 475 MB/S, o que faria uma leitura de 16 Kbytes levar algo em torno de 34 uS, acrescentando pouco aos valores acima (os valores de transferência vão subir posteriormente, durante a transmissão da unidade de disco à sua controladora, por exemplo a 100 MB/s em um link FC-AL; no caso de nosso exemplo, subiriam para 161,5 uS, utilizando-se o mesmo link mencionado anteriormente).

Portanto, levando-se em conta apenas os números referentes aos movimentos mecânicos, temos que um máximo de 62,5 I/O´s por segundo seria o máximo que o dispositivo poderia executar, dadas condições extremamente adversas. É importante lembrar que este número aplicá-se ao DISCO FISICO, e não aos volumes 3390 simulados nele. Voltando ao exemplo do item anterior, temos 51 volumes lógicos 3390-3 simulados em um único disco físico. Isto significa que 62,5 Iops seria o máximo de operações que os 51 volumes poderiam executar por segundo, ou 1,2 iops / 3390-3, aproximadamente.

Inversamente, caso utilizemos o menor tempo de seek (0,55 mS, *"single track seek"),* e 1 mS de atraso rotacional, teríamos um tempo de 1,55 mS por operação, o que nos levaria a espantosos 625 iops por disco físico, ou 12,3 iops em média por 3390-3. A média de tempos de resposta encontrada no mundo real para operações de leitura em blocos de 16Kbytes vai de 19 a 25 mS, dependendo do subsistema utilizado, os acréscimos ficando por conta das simulações (lembrar, por exemplo, do número de operações SCSI que tem de ser efetuadas para trazer os 16Kbytes do registro a ser lido, e que seria normalmente um único registro em uma única trilha – item 1.4 acima).

Um ponto importante aqui é que os discos físicos ainda levam tempos da ordem de dezenas de milisegundos para completar uma operação. Os tempos menores, por volta de 2 a 5 mS aos quais nos acostumamos só são possíveis devido a existência de caches. Um ambiente 100% CACHE MISS continuará a observar tempos iguais ou maiores aos que víamos na época dos 3390´s ou 3380´s, em média acima de 20 mS, sendo o "DISCONNECT TME" seu maior componente.

2.10 – Eletrônica

Os novos discos, além de maiores velocidades e capacidades, também apresentam uma serie de recursos adicionais em seus circuitos locais de controle, como o buffer local, ou ALB (*"Actuator Level Buffer"*), processamento de filas de operações armazenadas, pré-leitura de dados e mudança da ordem das operações. A figura abaixo mostra parte da lista destes novos recursos ainda do mesmo disco FC-AL 146 Seagate usado como exemplo até agora :

3.1 Standard features

Cheetah 10K.6 FC drives have the following standard features:

- Integrated dual port FC-AL controller
- Concurrent dual port transfers
- Support for FC arbitrated loop, private and public attachment
- Differential copper FC drivers and receivers
- Downloadable firmware using the FC-AL interface
- Supports SCSI enclosure services via interface connector
- 128-deep task set (queue)
- Supports up to 32 initiators
- Drive selection ID and configuration options are set on the FC-AL backpanel or through interface commands. Jumpers are not used on the drive.
- Fibre Channel worldwide name uniquely identifies the drive and each port
- User-selectable logical block size (512 to 528 bytes per logical block) in even numbers of blocks
- Selectable frame sizes from 256 to 2,112 bytes

Figura 4 – Especificações de funcionalidades

Como pode ser visto na figura acima, este dispositivo pode manter até 128 operações pendentes na memória (*"queue-depth"*) e atender a até 32 "*initiators*" diferentes. Um pouco além, na mesma lista, há a especificação de um buffer local de 8.192 Kbytes.

No momento em que um pedido de leitura é recebido de um *host* (ou, no caso dos subsistemas, das placas controladoras de discos), a lógica da unidade verifica se houve definição de modalidade em *"pré-fetch"* anteriormente (Mode Page 08h)caso sim, verifica-se primeiro se o bloco pedido já está disponível no cache. Se não , é feito acesso diretamente a mídia e os dados colocados no cachê e enviados ao host. Os blocos imediatamente seguintes são colocados também em cache, mesmo que o pré-fetch mode esteja desligado. Nos próximos requests, estes dados são reportados como pré-fetch hit, ao invés de cache-hits.

Também via Mode Set, a controladora pode especificar a quantidade de blocos a serem armazenados via pré-fetch, e se o drive deve ou não cruzar os limites de cilindro durante esta operação.

Quaisquer outras operações que sejam recebidas durante o processamento desta primeira serão enfileiradas na "*task-queue*" do dispositivo.

É comum a reordenação das operações pendentes na "task-queue", de forma a otimizar a movimentação mecânica. Ou seja, operações de leitura ou gravação pendentes são organizadas por localização física, de forma a aproveitar ao máximo a movimentação. É como organizar as pessoas em um elevador na ordem de saída por andar, ao invés da ordem em que entraram.

Alguns discos são modificados para suportarem mais de uma fila de tarefas. Deste modo, as operações de I/O iniciadas por hosts são tratadas em primeiro lugar (ainda ordenando-se por localização física), e posteriormente, as operações "internas" (como "cache pré-fetch reads", por exemplo). Até o momento não houve

qualquer anuncio oficial de unidades de disco que mantivessem filas de acordo com especificações de desempenho do sistema operacional (OS/390 ou z/OS WLM IRD, por exemplo). Portanto, mesmo que o subsistema como um todo suporte funções de WLM/IRD como o "I/O PRIORITY QUEUING", quando chegamos ao nível dos discos físicos, as operações podem ser executadas em uma sequência distinta.

2.11 – Discos de Estado Sólido ("Solid-state disks")

Introdução

Recentemente, uma nova família de componentes foi acrescida às tecnologias já conhecidas da área de armazenamento.

Cercada de promessas e (de um modo geral), desconhecimento sobre suas capacidades reais, os discos de estado sólido, ou *"ssd's"*, conforme sua sigla em inglês, dão sinais de terem vindo para ficar, iniciando sua ocupação por nichos específicos na hierarquia de armazenamento corporativo.

2.12 - Histórico

Apesar de muitos acreditarem que esta tecnologia seja nova, a ideia já conta com mais de 30 anos, tendo sua primeira versão comercial surgido na forma de um pacote de EAROMs (*"Electrically-Alterable Read-only Memory")*, oferecida somente nos Estados Unidos por uma empresa chamada *General Instruments* (a qual não logrou muito sucesso na época, pois a promessa de durabilidade do dispositivo, de pelo menos 3 anos, não conseguiu se confirmar na prática, o que causou sua retirada das prateleiras).

Em 1976 a empresa *Dataram* anunciou seu SSD (batizado de "BULK CORE"), voltado principalmente para os computadores modulares, e que emulavam os discos tradicionais, produzidos pela DEC e Data General (atual EMC).

Um outro marco histórico foi o da Curtis,Inc., que em 1985 introduziu seu "ROMDISK", o qual foi o primeiro SSD utilizado nos antigos IBM PCs.

Feitos de uma tecnologia diferente, apesar de chamados por nomes semelhantes, os NAND-based flash memory SSDs são bastante mais recentes que seus predecessores ("RAM"), tendo seu primeiro formato comercial apresentado pela Toshiba,Inc., na forma de um cartão de memória chamado de "SmartMedia", em 1995, com capacidades entre 0.5 a 128 MBs.

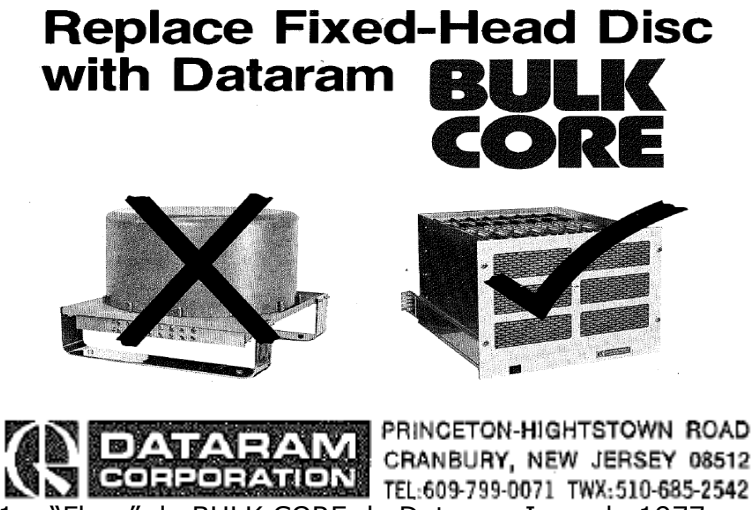


Figura 1 – "Flyer" do BULK CORE da Dataram,Inc., de 1977

Ao longo de sua história, a maior dificuldade no caminho desta tecnologia, tem sido a criação de um dispositivo de capacidade, compatibilidade e custo suficientemente atrativos para tornar seu uso uma realidade.

Os desenvolvimentos obtidos em cada uma de suas áreas componentes culminaram com um grande retorno ao mercado corporativo em 2003, início da fase de avaliação e conhecimento a qual vivemos ainda hoje.

2.13 – Tecnologia

Os primeiros modelos de SSD eram, basicamente, circuitos de memória RAM que se conectavam à CPU através de protocolos de acesso de periféricos (como EIDE ou SCSI), e que mantinham os dados armazenados empregando baterias ou outras formas suplementares de energia, para períodos em que as fontes principais estivessem desativadas.

Já as versões recentes, responsáveis pela nova onda destes dispositivos, são criadas com tecnologias "NAND-flash", que operam de forma distinta das primeiras, apresentando vantagens como seu custo mais baixo, e poder reter quase indefinidamente os dados, pelo uso de EEPROMs ("Electrically Eraseable and Programmable Read-Only Memories).

Os dados a seguir foram extraídos do documento TL/D/11951, publicado pela National Semiconductor Corporation em 1995, baseado no texto de Robert Frizzell de 1994, intitulado *"What´s all this Flash stuff?"*

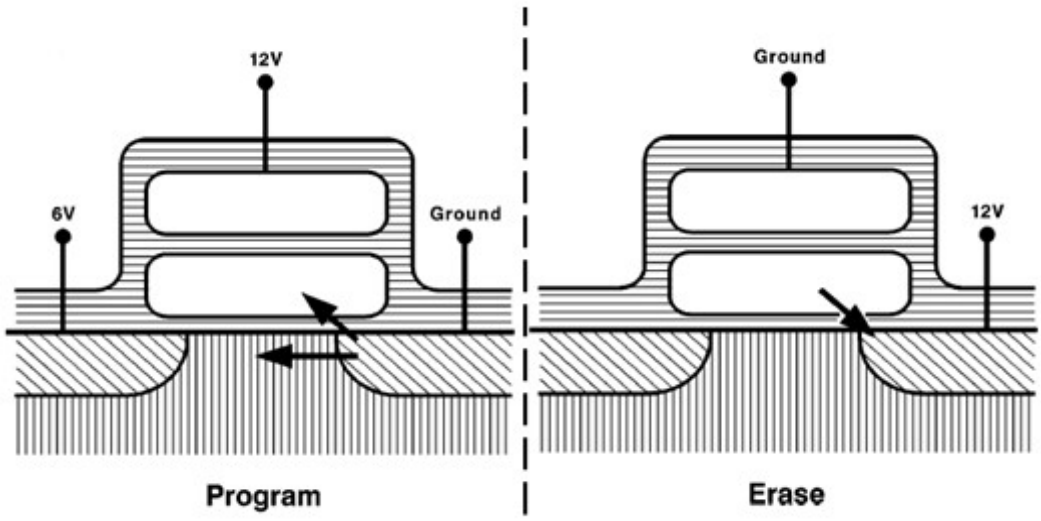


Figura 2 – Detalhes das portas de controle de um transistor flash

A maior parte da atual tecnologia flash se baseia em circuitos do tipo ETOX ("EPROM Tunnel Oxide"), para criar seus "bits" de dados. Na figura 2, a operação de "programação" ou gravação de dados é efetuada "forçando-se" os elétrons para dentro do campo de controle com voltagens altas, o mesmo se passando com o apagamento.

Uma vez carregado, o campo de controle facilitará (ou não), a passagem da corrente de leitura entre os polos do transistor, caracterizando seus 0´s e 1´s.

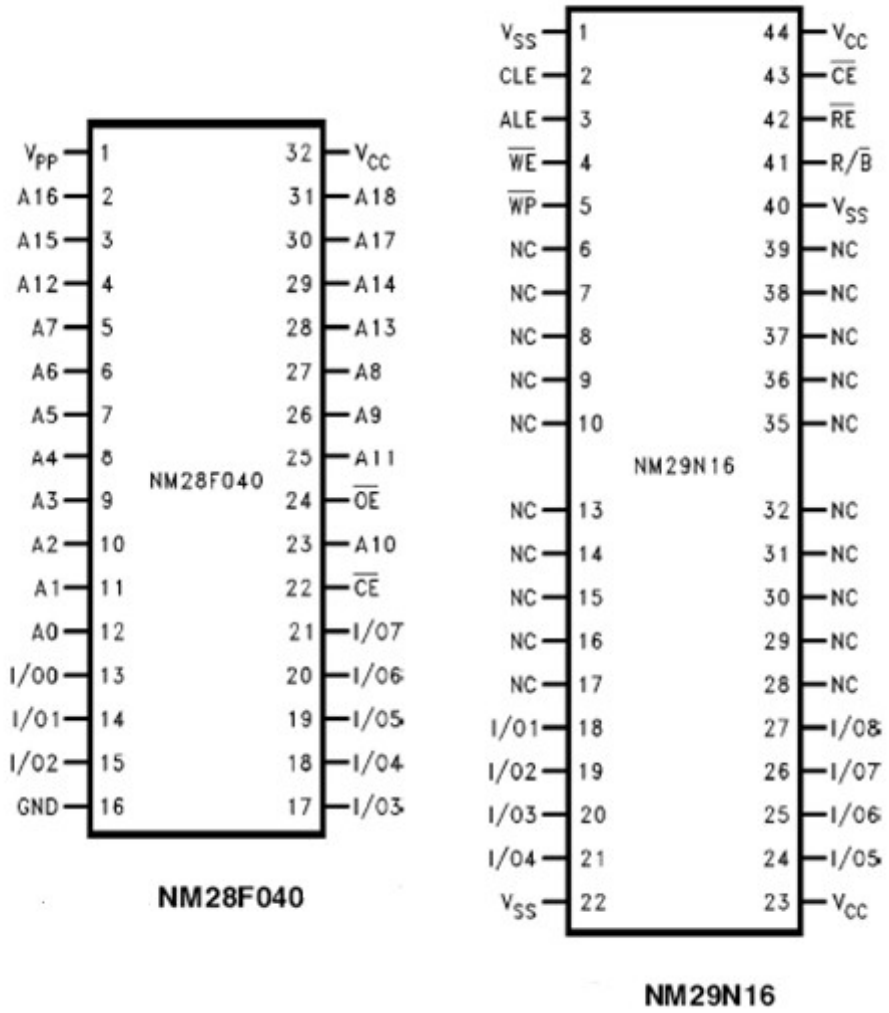


Figura 3 – Detalhamento de pinagem de flash NOR e NAND

Há memórias flash de 2 tipos, chamadas de NOR e NAND.

As NOR-flash apresentam-se muito semelhantes à circuitos EPROM tradicionais, como pode ser visto na figura 3, referente ao circuito NM28F040 da National Semiconductor. Sua grande vantagem reside no fato de permitir o acesso direto aos bytes armazenados, exatamente como ocorre em uma memória RAM normal. Entretanto, circuitos NOR, justamente por sua grande semelhança às EPROMs, consomem altas correntes de operação (principalmente para apagamento), e o tempo necessário para mudança de estado dos dados chega à ordem de segundos, o que os torna impraticáveis para uso em SSD.

Já as NAND-flash (figura 3, NM29N16), foram projetadas para operar como dispositivos periféricos de um computador, ao invés de uma área de memória. Por esta razão, a pinagem dos circuitos comerciais é diferente de uma EPROM, não apresentando, por exemplo, os pinos de endereçamento, típicos em memórias, mas sim um barramento para comunicação, através do qual recebe comandos, endereços (geralmente divididos em 3 ciclos, pois a maioria utiliza barramentos de 8 bits), e dados, exatamente como um periférico normal.

As NAND-flash, portanto, não permitem o acesso direto aos dados, na forma de bytes ou bits armazenados, exigindo uma sequencia de comandos e manuseio de ponteiros para sua obtenção. Além disso, sua tecnologia, baseada em EEPROM,

exige que os dados sejam COMPLETAMENTE APAGADOS (corrente de apagamento), antes que novos dados possam ser gravados. Em alguns circuitos, a sequencia exige 2 ciclos de apagamento completo, antes que a área possa ser reutilizada.
As altas correntes necessárias aos ciclos de gravação e apagamento são as principais responsáveis pelo encurtamento da vida útil de uma célula de dados, uma vez que geram desgastes no substrato, na forma de calor adicional.

Para se ter uma ideia da semelhança entre NAND-flash e periféricos, veja abaixo a sequencia de comandos necessários à sua operação (extraída do documento da National Semiconductor citado acima):

"Programming involves sending in a programming command (40H, this and the following NOR command examples are standardized commands for 1Mb devices) while strobing WE low. This is followed by the address and the data while again strobing WE low. A successful program can be verified by issuing a program verify command (C0H) and reading the data. The data read out then needs to be compared against the data inputted by the processor.

The erase operation involves writing two consecutive erase commands (20H). Again a verify command (A0H) is sent to check for a successful erase. There is typically a one second delay between the second erase command and the erase verify command.

Interfacing to a NAND device is similar to interfacing to a peripheral device on a PC motherboard. The device does not have any address pins but instead has eight I/O lines through which all data and commands pass..."

Os tempos necessários ao término de cada operação são bastante diferentes, conforme abaixo (tempos válidos para os circuitos da figura 3 acima):

- Leitura (00H + 3-cycle address) – 25uS
- Programação (40H) – 300uS
- Apagamento (60H) – 6mS

Os circuitos atuais são mais rápidos que os mencionados anteriormente, mas ainda mantem a mesma ordem de grandeza entre os tempos de suas operações.

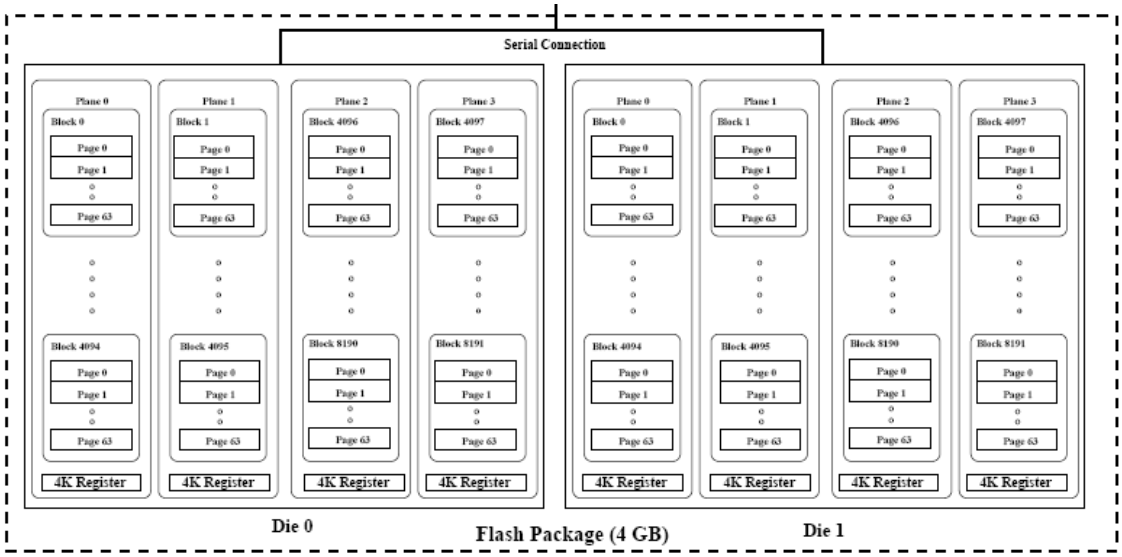


Figura 4 – Detalhamento (Samsung K9XXG08UXM flash drive)

Os transistores são chamados de "células", com opções de "single" ou "multi-level cells", sendo as últimas as de menor custo, por permitirem uma maior quantidade de dados por área de circuito. As células são grupadas em "páginas", as quais são grupadas em entidades maiores, chamadas "blocos". Estes, por sua vez,

são organizados em "planos", os quais, juntos formam um "die", ou circuito de memoria flash.

A quantidade de memória pode ser variada através de diferentes densidades em cada um desses elementos, bem como pela inserção de circuitos adicionais na mesma placa, conforme mostrado no esquema da figura 4.

Um aspecto facilmente notável nesta figura é que o barramento (8-bits), pode rapidamente tornar-se um ponto de estrangulamento para as configurações, a partir do momento em que passe a atender vários circuitos distintos. Por exemplo, no dispositivo Samsung mencionado acima, o barramento leva tempos da ordem de 100uS para transferir os dados do registrador de saída do "die" até o outro lado do barramento, o que causa enfileiramentos entre os acessos, e levando os projetistas a utilizarem esquemas de sobreposição ("interleaving"), entre as operações, de forma a explorar ao máximo a banda disponível no mesmo:

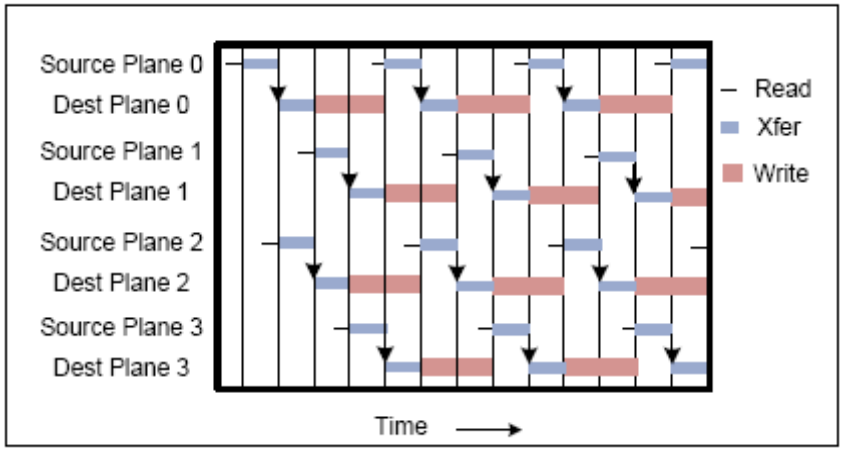


Figura 5 – Operations' Interleaving – Microsoft Research

Um outro recurso bastante utilizado hoje em dia é o de grandes quantidades de ALBs ("Actuator-Level Buffer"), ou áreas de cache nos próprios SSDs, que permitem que as operações de gravação, muito mais lentas, sejam efetuadas em tempos de memórias RAM, já que os blocos (que precisam ser gravados ou apagados inteiramente), utilizados são da ordem de 256KB.

Portanto, algumas das principais características destes dispositivos são:

- Leitura e gravação feitas por correntes altas, o que causa desgaste físico dos materiais. Isso dá origem aos mecanismos chamados de "wear-leveling", ou niveladores de desgaste, que sempre gravam dados em um bloco diferente do original, ao mesmo tempo que força os dispositivos a possuírem uma quantidade maior de memória do que sua nominal.
- Leituras muito mais rápidas que as gravações (comportamento EPROM e nivelamento de desgaste)
- ALBs maiores que os dos discos tradicionais, para mitigar o efeito da gravação mais lenta
- Montagens com diversos circuitos e barramentos, para evitar estrangulamentos na comunicação
- Conectividade (externa) via protocolos de mercado (FC, SCSI, SAS, etc)

2.14 – Implementações atuais

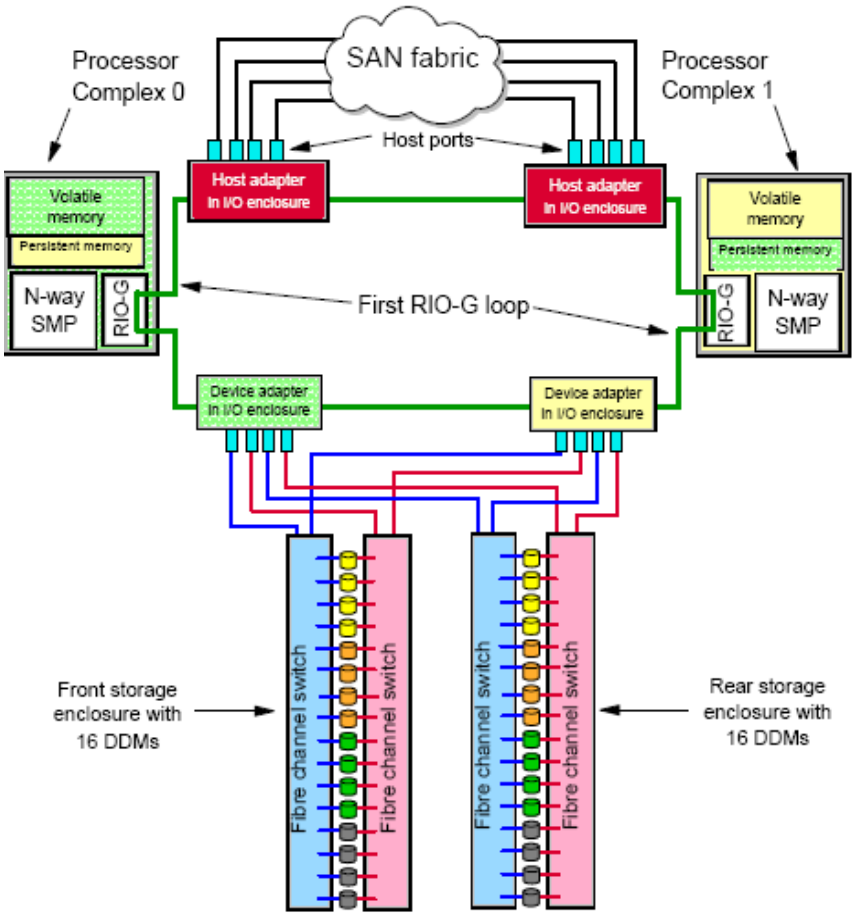
Os subsistemas de armazenamento corporativo atuais (2009), foram projetados para discos tradicionais, considerando, em sua engenharia, a vazão de que cada unidade é capaz, e ajustando o restante dos componentes para comportar esta vazão, vezes o número de unidades que comporte a capacidade final desejada.

O acréscimo de dispositivos de estado sólido ao projeto original, apesar de ter se tornado uma exigência do mercado, lembra bastante a chegada de carros de fórmula 1 às ruas de uma cidade normal. Além dos carros mais velozes terem seu rendimento reduzido, devido às proporções inadequadas do sistema viário, todos os outros veículos também serão afetados por seu uso (ninguém vai querer estar na rua de mão única a 80 por hora com seu automóvel, sabendo que um outro vem vindo em sua direção a 350 KM/H !!!!).

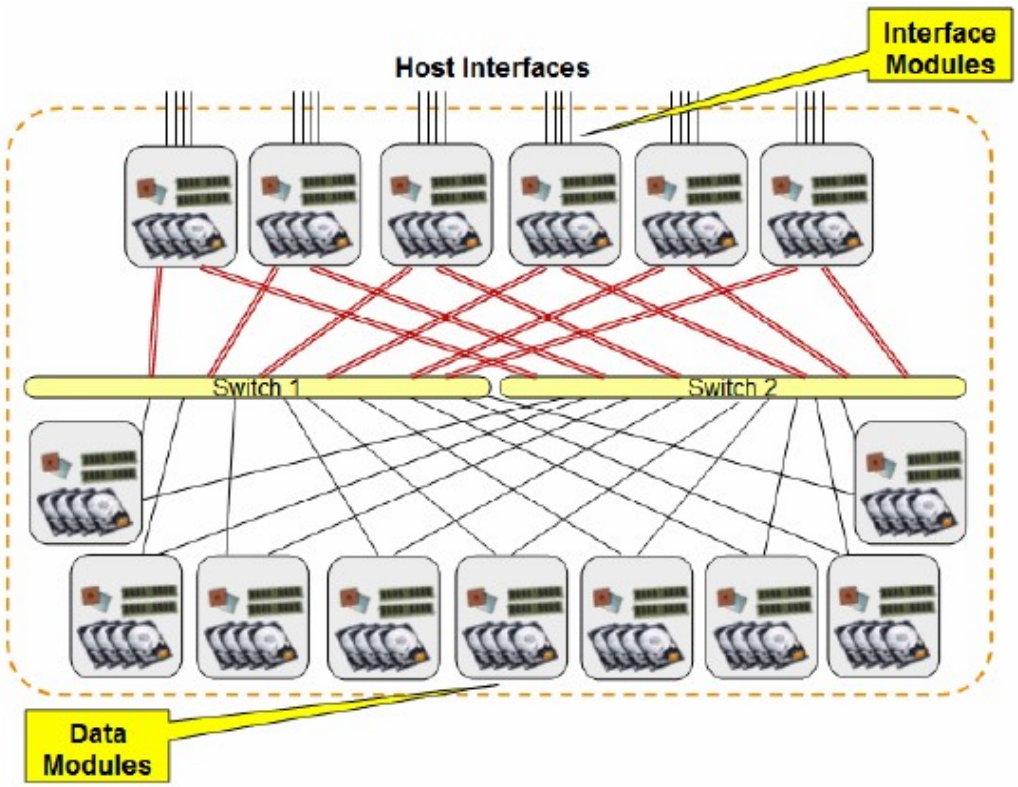
As figuras a seguir foram obtidas (respectivamente), das seguintes fontes:

-
- Manual IBM SG24-6452 DS8000 Concepts and Architecture
- Manual IBM SG24-7659 XiV Storage System: Concepts, Architecture and Usage
- Hitachi USP V Family Architecture Guide
- http://gestaltit.com/featured/top/stephen/emc-symmetrix-vmax-neither-nor/

IBM DS8K

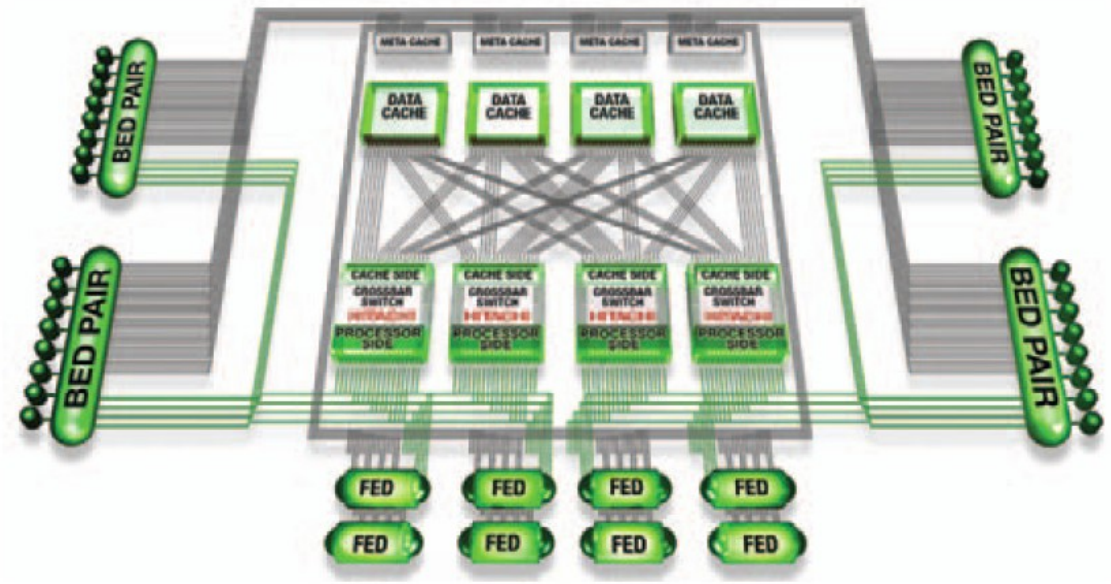


IBM XiV

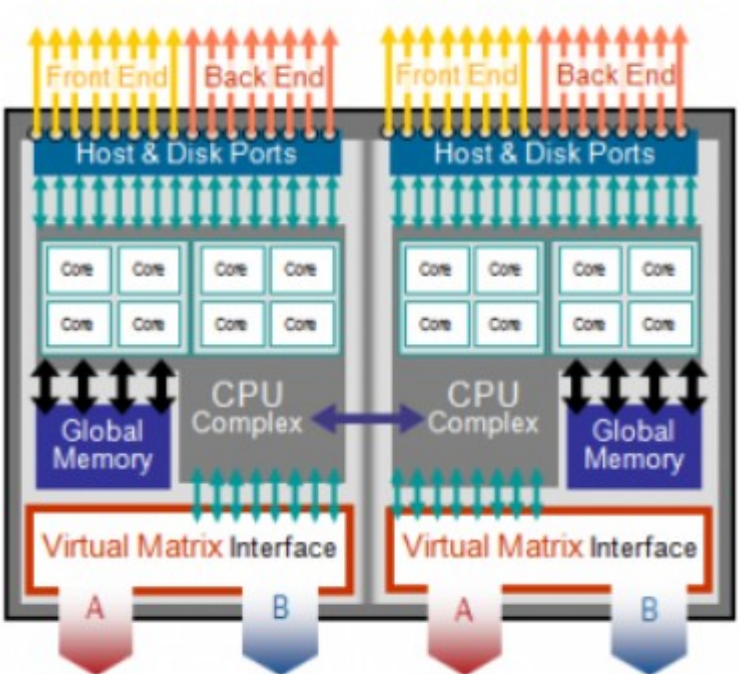


Interface and Data Modules are connected to each other through an internal IP switched network.

HITACHI USP V



EMC V-MAX



Todas as arquiteturas acima se conectam à grupos de discos através de portas (FC-AL, FC-SW, FC-ALsw), e definem a quantidade máxima de capacidade suportada por cada subsistema como uma fórmula simples, multiplicando o MÁXIMO de discos que uma porta pode suportar (vazão, operações por segundo), vezes o número MÁXIMO de portas em uma controladora.

Um disco atual normal (padrão FC, 15Krpm), é capaz de fornecer 250 operações de leitura aleatória por segundo (8KB), e até 130 operações sequenciais (256KB), com uma vazão de até (aproximadamente), 30 MB/S.

Já um SSD (padrão Stec Zeus), suporta até 5900 leituras aleatórias e até 240 MB/S em acessos sequenciais (novamente, em blocos de 256KB).

Caso as portas de acesso tenham sido dimensionadas para, por exemplo, um máximo de 16 discos por porta (padrão FC4Gb), o que seria razoável para discos tradicionais, esta mesma capacidade poderá ser consumida por um único disco de estado sólido.

Por isso, até que novos subsistemas, com conexões especiais ou dedicadas à este tipo de dispositivo, surjam no mercado, a utilização dos mesmos torna-se um exercício de equilíbrio entre custo e utilidade, uma vez que mesmo os maiores SSD disponíveis atualmente (400GB), caso configurados como 1 / porta de acesso (devido à sua vazão e velocidade), criariam uma máquina de alta velocidade, porém de capacidade muito baixa, não adiantando simplesmente acrescentar discos normais às mesmas portas, pois os mesmos interferem com o desempenho dos primeiros, e tornam-se significativamente mais lentos que suas especificações originais, devido às competições pelo acesso.

2.15 – Utilização

Cada organização apresenta um conjunto próprio de necessidades, no que tange à área de tecnologia de informação, e, mais especificamente, ao armazenamento de seus dados. Tais necessidades se derivam, diretamente, das características próprias de cada operação, e tornam, praticamente, impossível a criação de um corpo de regras ou definições que as abranjam.

Entretanto, dadas as características destes novos dispositivos, talvez seja possível o estabelecimento de algumas bases que possam servir de guias aos profissionais envolvidos na avaliação dos mesmos, por isso seguem aqui alguns comentários finais.

Discos de estado sólido apresentam seus melhores resultados em cargas de LEITURA ALEATÓRIA e de blocos pequenos, típicas de ambientes transacionais.

Seu uso deve ser pesado contra os impactos de custos maiores, não somente no preço por GB de área, mas também na área não utilizada que pode vir a se acumular por detrás dos mesmos, justamente devido a seu desempenho.

Uma opção recomendada de implementação seria a que segrega os recursos utilizados pelos SSD dos utilizados pelo restante da capacidade do subsistema (áreas de cache, portas de front e back end, etc), caso assim o permita a controladora.

Mesmo para ambientes transacionais, sua utilidade pode vir a ser apenas parcial, uma vez que a maior parte destes ambientes armazena seus dados em bases estruturadas (SQL, Oracle, DB2, Ingres, etc), os quais são sujeitos a constantes reorganizações e recargas, que podem impactar negativamente a vida útil destes dispositivos.

Uma última recomendação seria a manutenção de DUPLEX VOLS para os volumes criados em tais dispositivos, uma vez que são de tecnologia recente, e seu MTBF ainda não pode ser avaliado na prática (General Instruments, 1977?). Entretanto, tal arranjo ainda não é suficientemente comum no mercado para que os impactos de um tal cruzamento tecnológico sejam avaliados apropriadamente, e, portanto, deve ser adotada com cautela e restrições.

3 – Interfaces a disco

3.1 – Introdução

Uma vez entendidos alguns dos aspectos importantes das unidades de disco rígido (ver Cap.1 – Discos Rígidos), conforme utilizadas atualmente nos modernos subsistemas de armazenamento, podemos, agora, analisar suas interfaces.

O termo *interface* neste capítulo refere-se ao tipo de conectividade utilizado para acessar os discos. Os principais fabricantes do mercado "High-End" utilizam 3 tipos de conexão, a saber: "SMALL COMPUTER SYSTEM INTERFACE" (SCSI), "FIBRE CHANNEL – ARBITRATED LOOP" (FC-AL) e SERIAL STORAGE ARCHITECTURE (SSA). Apesar de haver várias outras opções de conectividade, até o momento em que este documento esta sendo escrito, não há informações de estarem sendo utilizados em quaisquer subsistemas que atendam o ambiente MainFrame IBM.

Estas interfaces conectam os discos rígidos de um subsistema às suas placas adaptadoras de disco ("DISK ADAPTERS" – DAs), as quais recebem nomes distintos nos equipamentos de alguns fabricantes (ACPs para HDS Lightning 9900V series, SSA ADAPTER para IBM Shark 2105 800T, DAs para EMC Symmetrix 5.5 e DMX, p.ex.), mas sempre cumprindo a mesma função; interligar os circuitos internos de CACHE aos discos do subsistema.

Perfeitamente adaptados às suas funções, os protocolos de interface a disco apresentam características operacionais que influenciam em maior ou menor grau o desempenho final dos volumes.

3.2 – SMALL COMPUTER SYSTEM INTERFACE (SCSI)

A sigla SCSI refere-se simultaneamente a um conjunto de padrões ANSI de arquiteturas físicas de interfaces, e ao protocolo utilizado nesta forma de conexão. Suas várias definições evoluíram ao longo dos anos, chegando hoje a 10 padrões distintos (veja figura abaixo).

Tecnologia	Comprimento Max. Conexão (metros)	Velocidade Máxima (MBps)	Número Máximo De Devices
SCSI-1	6	5	8
SCSI-2	6	5-10	8 ou 16
Fast SCSI-2	3	10-20	8
Wide SCSI-2	3	20	16
Fast Wide SCSI-2	3	20	16
Ultra SCSI-3, 8-bit	1.5	20	8
Ultra SCSI-3, 16-bit	1.5	40	16
Ultra-2 SCSI	1.5	40	8
Wide Ultra-2 SCSI	12	80	16
Ultra 160/Ultra 4	12	160	16

Figura 5. Tipos de interface SCSI

Em todos os casos, a interface SCSI compõe-se de um barramento de dados (de 8 ou 16 bits) e linhas de sinal para identificação de operações e status. As principais diferenças entre um tipo de SCSI e outro ficam por conta da largura do barramento de dados e da velocidade de "clock" utilizada. Cada dispositivo conectado a uma dada interface SCSI recebe assinalado um identificador ("SCSI-ID"), que corresponde a um bit do barramento de dados (de onde se concluí que padrões com 16 bits suportam até 16 dispositivos).

As operações sempre acontecem entre um "INITIATOR" e um "TARGET", ou seja, são sempre ponto-a-ponto. No padrão SCSI, enquanto um par esta em comunicação, todos os outros dispositivos tem de aguardar sua vez. Uma vez concluída a comunicação corrente, os outros dispositivos podem iniciar o que é

conhecido como *arbitragem*, ou seja, ativam os bits correspondentes aos seus identificadores, no barramento de dados, como forma de indicar uma operação pendente. O dispositivo com o MAIOR SCSI-ID vence (maior prioridade) e inicia a transmissão de dados, monopolizando, durante este tempo, todo o barramento.
Basicamente, *arbitrar* significa verificar se há, no barramento de dados, qualquer bit presente, de valor maior do que o do próprio dispositivo. Caso haja, o dispositivo tentando a arbitragem perde e tem de aguardar o próximo ciclo.

Nos grandes subsistemas, a placa adaptadora é definida como INITIATOR, e os discos como TARGETs. Uma vez vencida a arbitragem, o dispositivo inicia os ciclos de barramento, utilizando combinações das linhas de sinal ("BSY", "SEL", "REQ" e "ATN"), conforme descrição abaixo :

1. BUS FREE PHASE – estado de "repouso", durante o qual o barramento de dados está disponível aos dispositivos. Em um sistema com bastante carga, o barramento pode encontrar-se neste estado por períodos tão curtos quanto 1.2 pS (pico-segundos).
2. ARBITRATION PHASE – qualquer dos dispositivos conectados a interface pode arbitrar e ganhar controle sobre o barramento em até 3.6 pS ou menos. Os dispositivos que perderam a arbitragem tentarão novamente quando o barramento retornar ao estado de disponibilidade
3. SELECTION PHASE – esta fase pode ocorrer em até 580 nS (nano-segundos). Caso o TARGET não responda em até 250 mS (mili-segundos), o barramento retorna ao estado de disponibilidade
4. MESSAGE PHASE – esta é a primeira fase na qual há transmissão de dados no barramento, durante a qual o INITIATOR pode enviar mensagens ("IDENTIFY", p.ex.). Mensagens são sempre transferidas assincronamente (a transferência de cada BYTE é envolta por um par "REQ/ACK", sendo a direção indicada pelo sinal "I/O").
5. COMMAND PHASE – comandos ("INQUIRY") são trocados, identificando os dados a serem transmitidos
6. DATA IN/OUT PHASE – os dados requisitados são enviados pelo barramento
7. STATUS PHASE – informações sobre o andamento e a ocorrência ou não de erros
8. MESSAGE IN PHASE – final das transmissões, tipicamente com mensagem de "COMMAND COMPLETE"
9. BUS FREE PHASE – retorno à situação inicial

Nota : para iniciar a fase de arbitragem, os dispositivos devem aguardar que o barramento esteja na situação de "BUS FREE" por um tempo pré-determinado, (aprox. 800 nS), chamado de "bus settle delay". Isto é devido a um fenômeno conhecido como "wire-OR glitch", que pode fazer com que o sinal de BSY pareça falso, mesmo tendo sido ativado como verdadeiro.

Uma vez obtido o controle sobre o barramento, entra-se no que é chamado coletivamente de "INFORMATION TRANSFER PHASES (COMMAND, DATA, STATUS & MESSAGE)". Conforme mencionado acima as informações são trocadas entre o INITIATOR e o TARGET de forma assíncrona, significando um ou dois bytes por vez (dependendo da largura do barramento), envoltos por "handshakes" compostos pelos sinais de "REQ/ACK", e com o sentido da comunicação identificado pelo sinal "I/O", sendo "true" a transferência do TARGET para o INITIATOR, e "false" a transferência em sentido contrário.

O protocolo SCSI define que as comunicações de comandos sejam feitas através de um "COMMAND DESCRIPTOR BLOCK – CDB", o qual sempre começa com o primeiro BYTE do comando (OP.CODE), seguido pela identificação da unidade

lógica (LUN), uma lista de parâmetros para o comando, e terminando com um BYTE de controle. Os bits 5-7 dos OP.CODES identificam o grupo ao qual o comando pertence, e os restantes bits 0-4 identificam o comando em si, dentro do grupo. Os grupos variam em comprimentos de comando (grupo 0 – comandos de seis bytes, grupo 1 – comandos de 10 bytes, grupo 5 – comandos de 12 bytes, etc).

O CDB também especifica o tamanho da transmissão de dados a ser efetuada. Comandos que usam 1 BYTE como indicador de comprimento permitem a transmissão de até 256 bytes de dados (byte = 0 / 256 bytes). Já comandos que utilizam múltiplos bytes para essa indicação permitem até 65535 bytes em uma única operação. Neste caso, a especificação de "length" = 0 indica que nenhum byte será enviado.

Ao final da operação o TARGET deve retornar ao INITIATOR uma indicação de STATUS, que informa como foi o andamento da operação. As indicações do STATUS BYTE são : GOOD (operação OK), CHECK CONDITION (ocorrência de erros), BUSY (resposta à tentativa de inicio de operação por um INITIATOR, enquanto o TARGET ainda esta processando um comando anterior) e RESERVATION CONFLICT (conflito de RESERVE/RELEASE em uma LUN ou EXTENT).

Durante a execução de um comando, o TARGET pode decidir que irá precisar de um tempo adicional para obter os dados (uma operação de READ, por exemplo, que implique antes em um SEEK), e desconetar-se do INITIATOR, permitindo assim uma melhor utilização do barramento.

Um ponto importante a ser ressaltado aqui é que, em linhas gerais, este protocolo implementa um esquema de prioridades que deve ser levado em conta durante a implementação de um subsistema que o utilize. Os discos físicos de prioridade mais alta (SCSI-ID mais alto) sempre ganharão acesso ao barramento ANTES dos outros. Isto, quando bem utilizado, pode trazer vantagens substanciais ao ambiente como um todo.

Algumas implementações de protocolo SCSI acrescentam uma funcionalidade conhecida geralmente como "FAIRNESS SUPPORT". Em linhas gerais, esta função impede que um TARGET inicie a arbitragem antes que todos os outros tenham tido sua chance para fazê-lo, diminuindo o impacto causado aos discos de menor prioridade (SCSI-ID mais baixo).

Um exemplo de sistemas atuais que utilizam o barramento SCSI no acesso aos discos físicos é o EMC n-1 Symmetrix Geração 5.0 ou 5.5. Os últimos utilizam Wide Ultra-2 SCSI LVD (Low Voltage Differentiator) de 80 MB/S.

3.3 – FIBRE CHANNEL ARBITRATED LOOP (FC-AL)

Fibre Channel é tanto um padrão de comunicação (cabeamentos e conexões), quanto um protocolo de transporte aberto, conforme definido por normas ANSI (Comitê X3T11) e que pode ser implementado tanto em fibras óticas quanto em fiações de cobre, sendo portanto, um padrão para meios físicos de comunicação. Este padrão é amplamente utilizado na conectividade física do que é conhecido hoje com "*SANs – Storage Área Networks"*, ou seja, redes especializadas na interconexão de dispositivos de armazenamento a servidores. A palavra "FIBRE" foi criada pelo comitê quando o uso de fiações de cobre foi introduzida no padrão. Originalmente, referia-se somente a fibras óticas.

Fibre Channel Aribtrated Loop ou FC-AL é uma das topologias implementadas pelas normas FIBRE (sendo as outras a *POINT-TO-POINT* e o *FIBRE CHANNEL SWITCHED FABRIC*, ou FC-SW). Todas estas topologias são utilizadas para interconectar dispositivos de armazenamento a seus servidores, em plataformas baixas. Não necessitando de sobrecargas adicionais, como as geradas por seleção de rotas em ambientes FC-SW (como o FSPF-Fabric shortest path first, por exemplo) e ainda atendendo a um grande número de dispositivos, é a única topologia deste tipo em uso nos subsistemas de armazenamento para MainFrame atualmente.

Baseado nos padrões Fibre, o FC-AL também utiliza o conceito de *ports* (o que pode, neste caso, ser entendido como "portas"), utilizando as "L"-ports, como a L-port, NL-port (node port com capacidade de loop), FL-port (fabric port com capacidade de loop), etc. Nesta topologia, até 126 *nodes* (L ou NL ports) podem ser interconectados por uma rede, a qual é gerenciada como um barramento compartilhado, no qual os dados fluem em uma única direção de cada vez, transmitindo dados e *primitivas* (nome dado ao protocolo de troca de informações de comando e controle), a 100 ou 200 MB/S, dependendo do loop ser baseado em uma rede de 1 ou 2 Gb/S.

Uma vez conectados fisicamente, os dispositivos participantes de uma rede FC-AL iniciam o que é chamado de LIP *(Loop Initialization Primitive)*, durante o qual os mesmos são identificados e, opcionalmente, recebem um mapa de seu posicionamento no loop. Após o LIP, a rede entra em um estado de gerenciamento, controlado pelo dispositivo de mais baixo endereço físico, o qual foi identificado por uma outra primitiva, chamada LISM (Loop Initialization Select Máster). No caso dos subsistemas de grande porte, esta função cabe à placa adaptadora.

Após terem sido executadas as primitivas de inicialização, qualquer dispositivo que queira iniciar uma conexão terá, a semelhança do padrão SCSI, que arbitrar. Caso haja mais de um dispositivo tentando a arbitragem ao mesmo tempo, o que tiver o menor endereço físico (AL-PA – Arbitrated Loop Physical Address) vence, ganhando controle sobre o loop, e podendo estabelecer uma conexão com outro node e utilizar, durante este tempo, toda a largura de banda disponível para sua transmissão. A maior parte das implementações FC-AL conta com o FAIRNESS ALGORITHM, o que impede que dispositivos de mais alta prioridade (menor AL-PA) possam monopolizar a conexão física.

O endereçamento neste ambiente é feito através de endereços de port de 24 bits, dos quais o último byte é adquirido durante a execução do LIP, identificando o dispositivo para a rede. Este esquema é o escolhido para redes FC pois a alternativa de 64 bits utilizada no endereçamento via WWN (World-Wide Name) força a utilização de cabeçalhos maiores de roteamento, e conseqüentemente, maiores tempos de transmissão. Dos 24 bits utilizados no que é chamado de 24-bit Port Addressing Scheme, os bits (23-16) designam o domínio, (15-8) identificam a área e os restantes (7-0) o port, correspondendo ao endereço físico da porta no loop para o qual esta se identificou.

Uma vez estabelecida a conexão, os dados são transmitidos em unidades de 2112 bytes de cada vez, o que é conhecido como FIBRE CHANNEL FRAME, utilizando-se o que foi padronizado como Classe 1 de serviços do padrão FC-PH (para esta classe, uma conexão dedicada é estabelecida entre origem e destino durante o período de transmissão dos dados e primitivas). O maior tamanho de dados transmitidos a cada operação é definido pelas primitivas de inicialização, variando entre os subsistemas.

3.4 – SERIAL STORAGE ARCHITECTURE (SSA)

Das 3 formas de conexão apresentadas até agora, a SSA é a que mais se distingue das outras, tanto física quanto logicamente. Mesmo tendo a capacidade de mapear protocolos anteriores (SCSI-2), suas formas de conexão e controles tornam-na bem diferente das demais, sendo atualmente utilizada somente no IBM Enterprise Storage Server (ESS Shark), dentre os subsistemas que atendem a plataforma MainFrame.

Sua forma mais comum de implementação é a de um loop, com 2 conexões físicas de leitura e 2 de gravação, o que permite até 4 operações simultâneas, sendo 2 em cada direção do loop. Isso é possível porque diferentemente das arquiteturas que compartilham o meio físico de transmissão, na SSA cada participante conecta-se diretamente a seu "vizinho", tendo 2 caminhos (um para leitura e um para gravação) para cada um. Desta forma, os dados que saem da placa adaptadora para um determinado disco são transmitidos de disco a disco, até chegarem em seu destino.

Caso haja, por exemplo, 8 discos em um loop, e uma gravação esteja sendo efetuada no terceiro, isso deixa o caminho livre entre o retorno do loop e os outros 5 discos, permitindo uma outra operação simultânea de gravação aconteça. A figura abaixo exemplifica a operação de um node SSA.

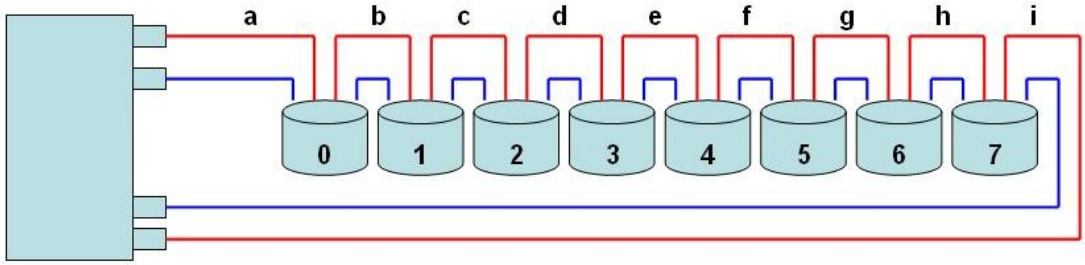


Figura 6 – SSA Loop : Se uma gravação estiver acontecendo entre a placa adaptadora e o disco físico 2, os trechos a-b-c do link em vermelho estarão ocupados. Entretanto, uma outra gravação pode ocorrer simultaneamente com o disco 4, seguindo a rota i-h-g-f, que estava livre. A mesma lógica vale para operações de leitura. Desta forma, até 4 operações (2 leituras e 2 gravações), podem acontecer ao mesmo tempo em um único loop. As placas adaptadoras SSA160 são compostas por 2 nodes SSA de 40 MB/S em cada link.

Nessa arquitetura, o conceito de arbitragem é substituído pelo uso de senhas ("tokens"), chamados de SAT e SAT´, um para cada sentido do loop, que circulam entre os dispositivos participantes. Quem tiver o SAT ou o SAT´ dará preferência ao envio de seus próprios frames à retransmissão dos frames de outros, ocorrendo o contrário aos dispositivos que não o tiverem naquele momento. A titulo de curiosidade, o nome "SAT" vem de *"SATisfy yourself"*, significando que quem estiver de posse da senha pode satisfazer sua necessidade de envio ou recebimento de dados naquele momento.

Apesar da banda nominal de cada link SSA ser de 40 MB/S, 20% da mesma é utilizada por uma forma de codificação de dados chamada de 8b/10b, na qual cada byte (8 bits de dados) é codificado na forma de 10 bits pela inserção de controles e sua substituição por valores especificados em tabela para cada byte. Isso é feito para atender necessidades de hardware, como por exemplo garantir a reconstrução do sinal de "clock" a partir dos dados enviados, o que seria difícil com

vários bits de mesma polaridade sendo transmitidos juntos, e a correção de um fenômeno chamado de "dc bias" que tende a surgir nesta mesma situação, dificultando a detecção da diferença de sinal entre 0´s e 1´s.

Cada frame SSA é composto por até 128 bytes de dados, acrescentados de cabeçalhos (com especificações do conteúdo do frame e o endereço do destinatário, chamado de UID), e verificações de erro CRC ao final.

Uma outra forma de conexão possibilitada por esta arquitetura insere uma outra placa adaptadora ao final de cada loop, ao invés de fechá-lo de volta na placa original. Esta forma é a utilizada no IBM-ESS e permite que cada placa defina seu domínio (grupos de discos), fazendo com que as transmissões de uma controladora não interfiram com as da outra. Ainda nesta forma de conexão, cada placa adaptadora liga-se a uma dos clusters do ESS durante operação normal, permitindo ainda que assuma o restante do loop, caso a outra placa ou seu cluster venha a falhar.

4 – Esquemas de Proteção (RAIDs)

O final dos anos 80 assistiu ao surgimento dos inicialmente conhecidos como *"Redundant Array of Inexpensive Disks",* e que foram rapidamente renomeados para *"Independent"*, uma vez que os controles adicionais necessários a sua operação muito cedo os tornaram bem mais caros que em sua concepção original. Um documento do Berkeley Research de 1988 é ainda hoje considerado como o início da definição destas estruturas, e compõe grande parte dos padrões conhecidos como "RAIDs Oficiais", de acordo com o RAB – RAID Advisory Board.

Projetados para substituir os que passaram a ser conhecidos como "SLEDs" ou *"Single Large Expensive Disks"* (e nos quais o termo "*expensive"* curiosamente não foi substituído), as matrizes de discos baratos propunham que um hardware mais barato, somado a algoritmos inteligentes de proteção poderiam prover os mesmos níveis de serviço e confiabilidade de seus irmãos maiores.

A idéia era bastante simples. Discos mais baratos quebravam mais, eram mais lentos e bem menores que os "gigantes" 3380 e 3390 da época. Portanto, para torná-los equivalentes, usar-se-ia vários deles simultaneamente, com algum esquema de proteção de dados, e de forma que a soma permitisse a emulação das capacidades. É importante lembrar-se de que naquele período, enquanto os 3390 dispunham de 1, 2 e 3 GigaBytes de espaço, os pequenos HDs ainda eram medidos em MegaBytes, com um MTBF ("*Mean Time Between Failures")* bastante reduzido.

Os esquemas de proteção criados para estas matrizes tornaram-se (infelizmente) conhecidos como NÍVEIS de RAID. Infelizmente porque o termo nível passa uma idéia de hierarquia, como se o RAID-4 fosse maior, mais completo, e de alguma forma superior aos RAIDs 1 ou 3, o que não poderia estar mais distante da verdade. Cada esquema RAID é diferente, não sendo maior, menor ou abrangendo as soluções de seu antecessor.

Níveis de RAID

RAID-0 (Striping) : Não fazendo parte da definição original, o RAID-0 deveria omitir o "R" de sua definição, pois não apresenta qualquer forma de redundância. Simplesmente divide os dados entre os discos que compõe a matriz. Uma vez que não há esquemas de proteção, é o nível que apresenta o melhor desempenho entre todos, sendo também o de custo mais baixo, pois utiliza 100% do espaço disponível em disco para dados.

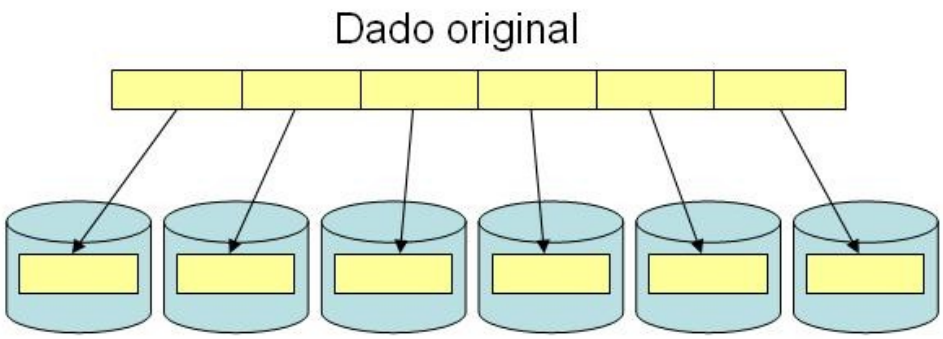


Figura 7 : RAID-0

RAID-1 (Espelhamento) : Todos os dados são gravados em duplicata, sendo cada cópia mantida em um disco físico diferente. Mais caro, pois utiliza somente 50% do espaço para dados (a outra metade destina-se somente à replicação). Em algumas implementações pode apresentar um desempenho melhor de leitura, pois se pode ler o mesmo dado de qualquer um dos dois discos envolvidos no espelhamento.

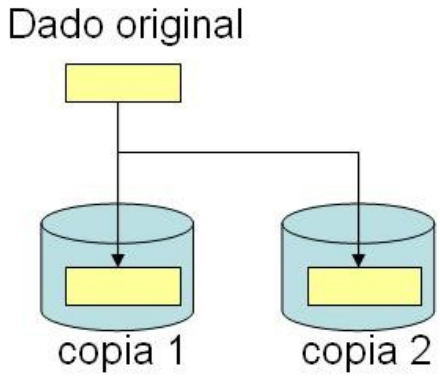


Figura 8 : RAID-1

RAID-2 (Bit-Striping + Hamming ECC) : Os dados originais são divididos ao nível de seus bits e espalhados entre a matriz de discos físicos. Adicionalmente, uma forma de ECC ("Error Checking and Correction"), chamada de *Hamming*, é calculada e gravada em discos adicionais. Esta forma de RAID foi há muito

abandonada por uma serie de razões. A controladora necessária para sua implementação era muito complexa, a quantidade de discos era bastante grande para a quantidade de dados suportada e, principalmente, porque os próprios discos físicos passaram a utilizar o ECC internamente, tornando-o redundante. Além disso, seu desempenho em ambientes transacionais mostrou-se inaceitavelmente inferior as outras implementações. Não é usado por nenhum dos subsistemas atuais.

RAID-3 (Byte-Striping + Paridade dedicada) : Nesta implementação, os dados são divididos em grupos de bytes (geralmente menores que 1024) e gravados nos discos de dados da matriz e, posteriormente, a paridade é gerada para esses mesmos dados, e gravada em um único disco dedicado a esta finalidade. De um modo geral, as implementações de RAID-3 e 4 são algo difíceis de distinguir

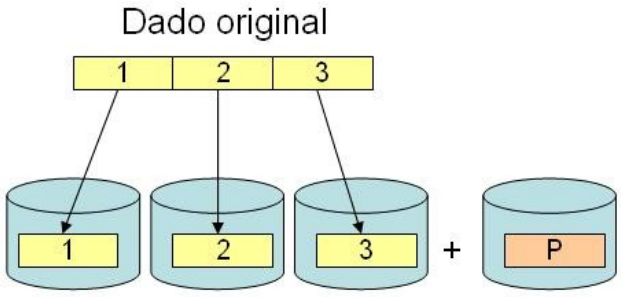


Figura 9 : RAID-3

RAID-4 (Block-Striping + Paridade dedicada) : Apresenta os mesmos mecanismos do RAID-3, com a única diferença notável ficando por cargo do tamanho dos blocos gravados em cada disco de dados (*"stripe size"*), geralmente maior que no anterior. Utiliza ainda o mesmo esquema de disco de paridade dedicado que, também como no último, pode resultar em um ponto de estrangulamento durante sua operação normal.

RAID-5 (Block-Striping + Paridade distribuida) : Esta forma implementa ainda a distribuição dos dados originais em blocos de dados através dos discos da matriz, porem, diferentemente das anteriores, o disco no qual a paridade é gravada não é mais dedicado. Neste caso, a paridade de cada grupo de dados é gravada em um disco diferente, evitando o engargalamento das anteriores.

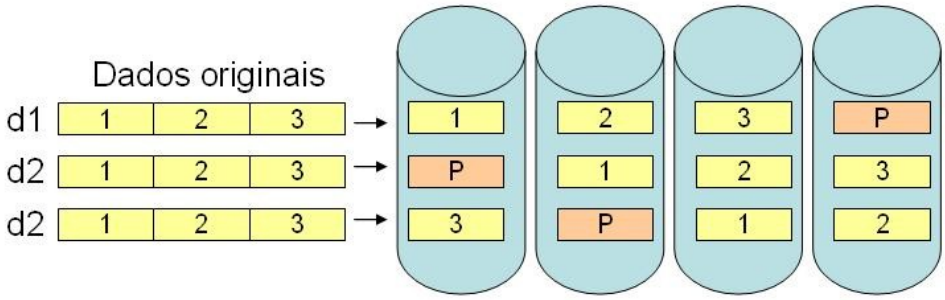


Figura 10 : RAID-5

RAID-6 (Block-Striping + Paridade dupla distribuída) : Bastante semelhante ao RAID-5, esta distribuição implementa a geração de uma paridade adicional, o que permite a recuperação da perda de até dois discos físicos na matriz.

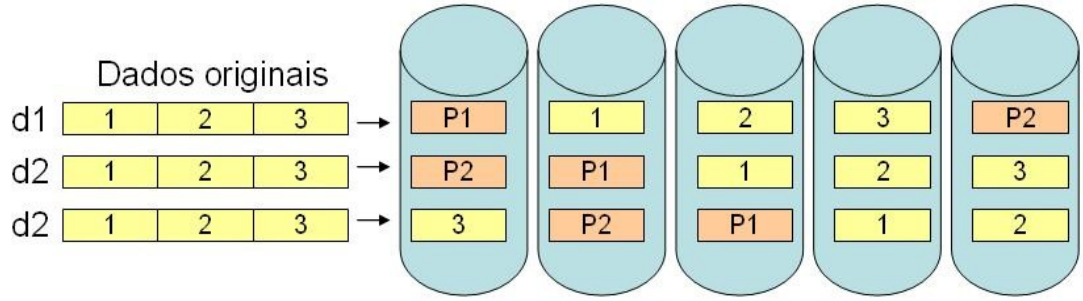


Figura 11 : RAID-6

O RAID-7 não é realmente uma implementação pública de RAID como as outras, mas um nome patenteado pela Storage Computer Corporation para sua implementação proprietária dos níveis 3 e 4.

Além das implementações simples conforme descritas acima, já há algum tempo utiliza-se a combinação de níveis de RAID para formar o que é conhecido como multi-RAID ou RAID ANINHADO (*"nested-RAID"*). Estes níveis são formados pela combinação de 2 das implementações simples, sendo a mais utilizada nos ambientes MainFrame a chamada RAID 0+1 ou 0/1 ou 10 (padronizações eficazes e universalmente aceitas para este tipo de definição ainda estão longe de ser obtidas). Esta forma de combinação permite que se explore as vantagens de uma sem (todas) as desvantagens que ela possa apresentar, compensadas parcialmente pela segunda. No caso do RAID 0+1, a grande vantagem é o paralelismo apresentado pelo RAID-0, sem o risco de segurança causado pela falta de redundância. Veja a figura abaixo :

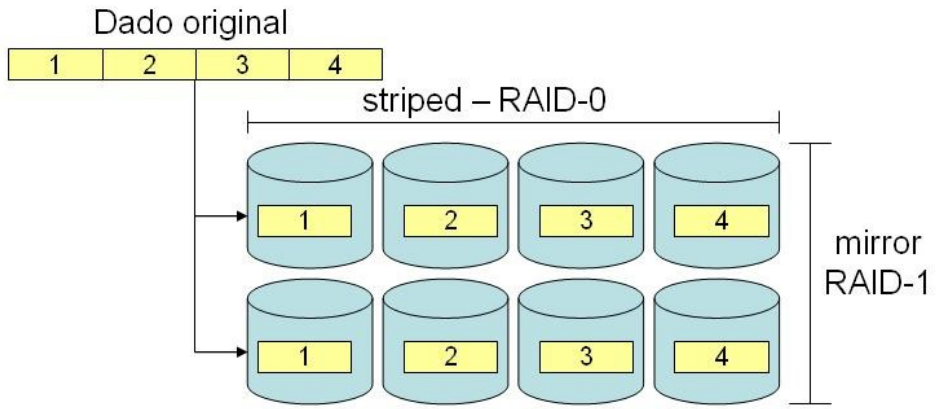


Figura 12 : RAID 0+1

Um ponto importante para se manter em mente em relação aos esquemas de proteção RAID é que eles alteraram a quantidade de operações físicas realizadas para uma dada operação de HOST. Por exemplo, uma única gravação de bloco vinda do sistema, em uma implementação em RAID-0 pode, dependendo do *stripe-size* definido para a matriz, resultar em varias operações de disco. De forma análoga, uma gravação em RAID-5 resultaria em pelo menos 3 operações a mais que a original (5 para RAID-6). Mesmo quando todos os blocos de um grupo devem ser gravados, haverá pelo menos n+1 operações (sendo "n" o número de discos de dados da matriz, mais a operação de paridade). Este é um ponto a ser considerado quando se pensa no número total de operações que se deseja que um subsistema execute, ou quando se necessita de funcionalidades adicionais, como replicação de dados (remota ou local), que também acrescentam ao número total de operações realizadas pelos discos no final. Operações de leitura também são afetadas da mesma maneira. Tomando-se, por exemplo, a leitura de um bloco de 32KB pelo host, em uma matriz com stripe-size de 8KB, serão necessários 4 acessos a disco para completá-la. A vantagem do paralelismo nessa forma de operação depende da implementação do subsistema. Caso todos os discos de um grupo residam sob controle de uma única placa adaptadora, as 4 operações (deste exemplo) terão de ocorrer uma após a outra.

5 – Estratégias de Cache

Mesmo que milhões de dólares sejam gastos na mais cara e poderosa controladora de armazenamento, junto com modernos canais FICON de 2Gb interligados a um *book* dedicado de uma T-Rex, caso o perfil de operações de entrada e saída de um ambiente seja 100% *"cache-miss"*, os tempos de resposta destas operações continuarão sendo da ordem de 20mS ou mais. Isto se deve ao fato de toda e qualquer unidade de armazenamento ainda ser baseada em discos físicos, com seus já conhecidos tempos de latência e acesso (veja Unidade 1 – Discos Físicos).

A única forma de obtermos os tempos aos quais temos nos acostumado, da ordem de 2mS por operação de disco é, em termos simples, os dados NÃO ESTAREM SOMENTE nos discos, mas também pré-armazenados em uma área de memória intermediaria chamada, genericamente, de CACHE. Evolução natural dos primeiros SMBs (*Speed Matching Buffers),* das controladoras IBM-3880, as áreas de cache, respondendo em velocidade eletrônica às operações viabilizaram os tempos de acesso necessários aos atuais ambientes transacionais, assim como as cada vez mais constritas janelas dedicadas ao processamento batch tradicional.

Novamente, a idéia aqui é bastante simples : caso seja detectado, pela controladora, um padrão de acesso seqüencial aos dados, esta iniciará um processo de pré-carga, lendo os próximos registros ou trilhas ANTES que os mesmos sejam pedidos pela aplicação. Desta forma, quando o pedido de operação for recebido, os dados já terão sido trazidos das unidades de disco para a memória, podendo ser imediatamente enviados em resposta.

Entretanto, a primeira parte desta descrição constantemente tem sua importância diminuída, ou seja, tudo se inicia com a DETECÇÃO DE UM PADRÃO SEQUENCIAL de acesso. Sem que isso ocorra (acesso 100% aleatório), os algoritmos e áreas de cache tornam-se quase inúteis, servindo somente como uma memória intermediaria para operações rápidas de gravação, o que, em média, representam algo da ordem de 20% das operações normais de um ambiente de produção.

Portanto, pelo exposto acima, há dois fatores que impactam fortemente o desempenho de uma controladora: a porcentagem de acesso seqüencial (que mais tarde se traduzirá em *cache-hits*), e a velocidade com que este acesso é feito (caso as operações, mesmo que seqüenciais, sejam feitas muito rapidamente, ou sob uma carga de sistema muito pesada, não haverá tempo hábil para a controladora pré-carregar os dados antes da chegada das operações), ou, devido à sua excessiva utilização, os dados terão de ser liberados antes do uso.

Uma outra função importante do cache é servir de área limítrofe entre a visão lógica das aplicações, e a implementação física de armazenamento (RAID-5 ou LSF, por exemplo). Durante uma operação de gravação, os dados chegam de acordo com o formato que os métodos de acesso de cada partição "pensam" que os dados finais tem (estruturas de CCWs). Deste ponto em diante, os algoritmos de simulação da controladora ganham controle e gravam cada bloco ou trilha, fisicamente, de acordo com seu próprio modelo de emulação, ocorrendo o exato oposto durante uma leitura. Os dados físicos são extraídos e convertidos de sua estrutura física, típica de cada subsistema, para o formato esperado pelas aplicações. Um ponto interessante sobre isso é que nenhuma das controladoras modernas realmente suporta o conjunto completo de instruções a disco disponível no z/OS (ou sua versão anterior, o OS/390). De um modo geral, opções do tipo

"bypass cache" ou *"inhibit cache load"*, apesar de entendidas, não tem mais como ser respeitadas, uma vez que TODOS OS DADOS PASSAM PELO CACHE. Outras, como as relativas a *"Record Level Cache"*, típicas em operações VSAM LDS para DB2, recebem suporte apenas parcial nas controladoras de alguns fornecedores.

À parte do algoritmo básico a partir do qual todos trabalham, os caches distinguem-se também por suas implementações fisicas.

Cache global : quaisquer das áreas de memória podem ser diretamente acessadas para leitura ou gravação por qualquer placa adaptadora (de disco ou canal), através de um *ASIC* (*Application Specific Integrated Circuit)*

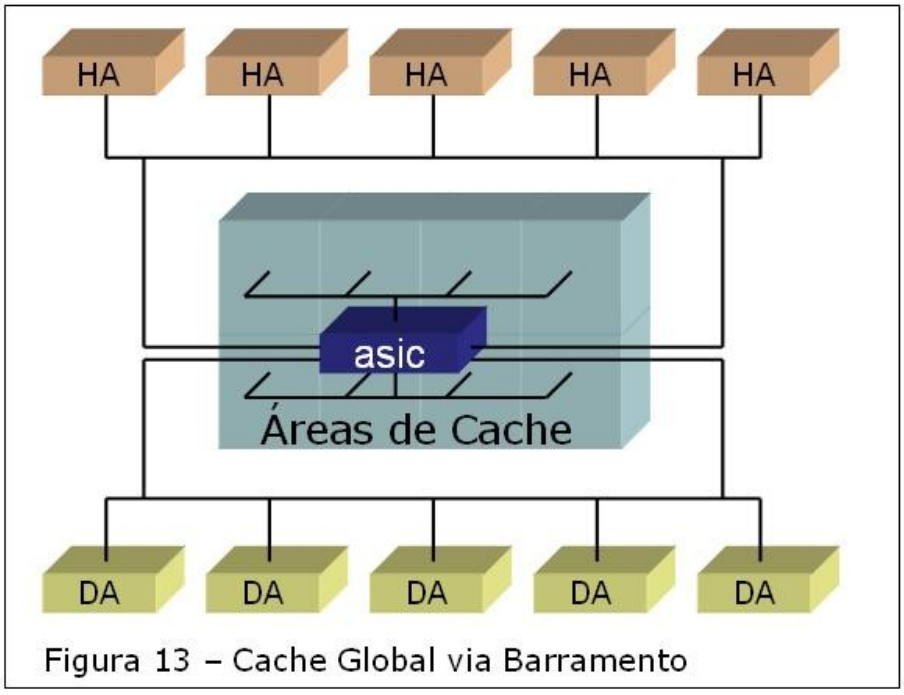


Figura 13 – Cache Global via Barramento

Cache "*Clusterizado*": há divisões físicas na memória, as quais são gerenciadas e acessadas separadamente, necessitando da intermediação de um processador de controle.

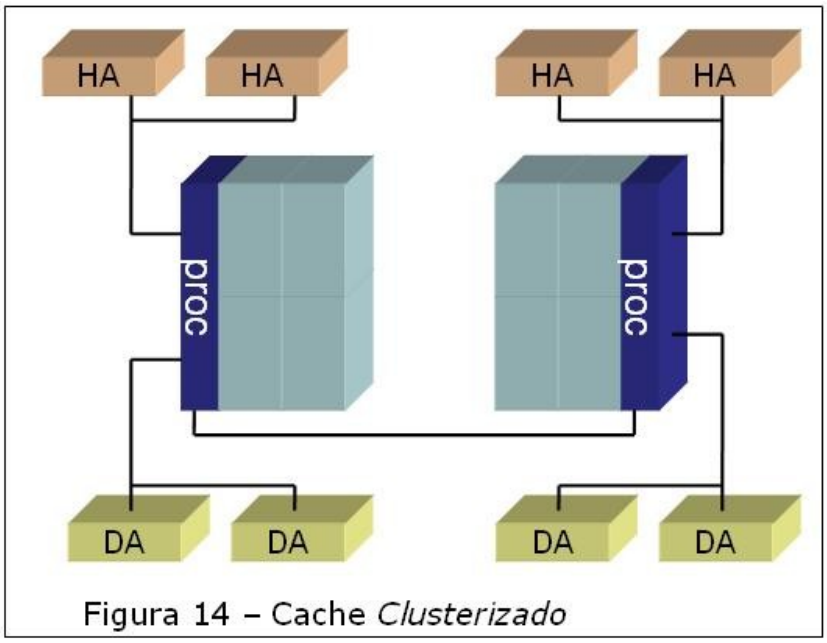


Figura 14 – Cache *Clusterizado*

Tamanho fixo : algumas implementações pré-reservam áreas de seu cache, de tamanho fixo (trilhas, por exemplo) e gravam os blocos de atualização dentro das áreas correspondentes. Geralmente dinâmicas, estas áreas não ocupam posições fixas na memoria, sendo alocadas (em seus tamanhos fixos), na medida em que dados destinados a ela tenham de ser carregados

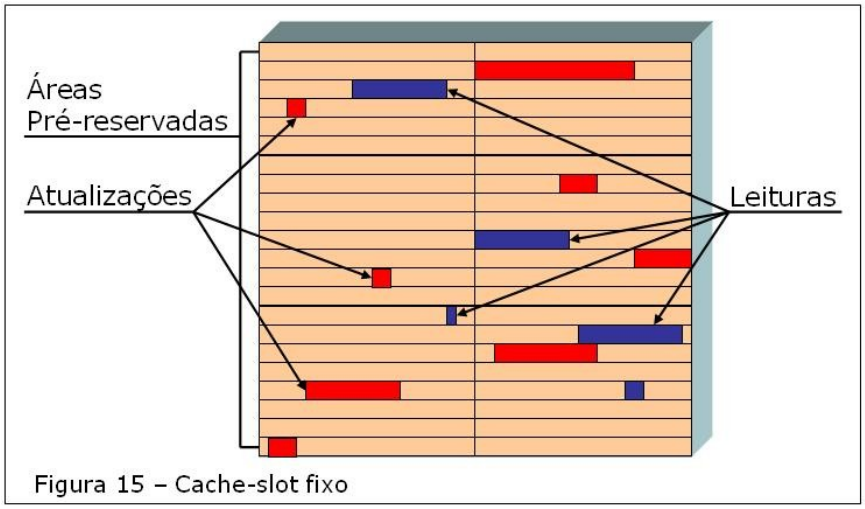


Figura 15 – Cache-slot fixo

Blocos : nesta implementação os blocos gravados / lidos são armazenados como entidades independentes na memória, sem uma pré-definição (áreas fixas) reservadas para entidades maiores (trilhas p.ex.).

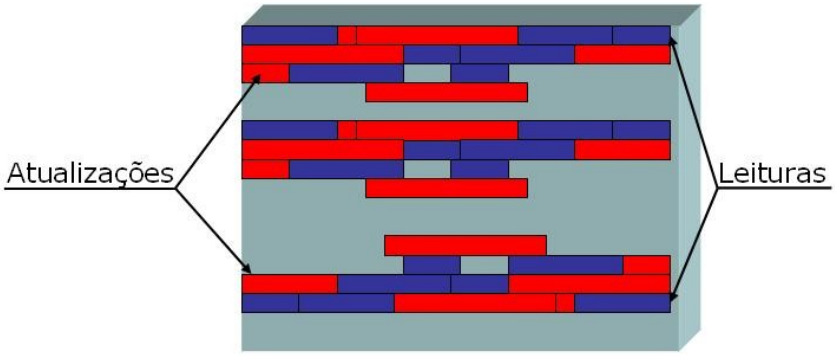


Figura 16 – Alocação em blocos

NVS (*Non-Volatile Storage*) : os dados recebidos do host são copiados para uma área não-volátil (mantida por baterias internas), geralmente menor e somente para gravação, que garantem que em caso de perda de força uma cópia integra seja mantida por um tempo determinado (duração destas mesmas baterias)

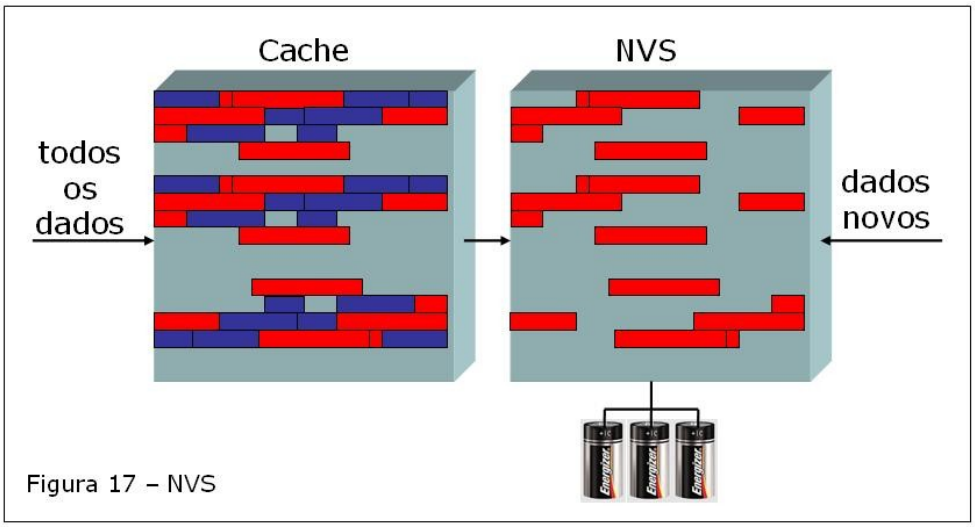


Figura 17 – NVS

NVSS (*Non-Volatile Subsystem)* : esta não é exatamente uma implementação somente de cache, mas o afeta diretamente. Nele, todo o subsistema é mantido por baterias em caso de falha no fornecimento de energia. O tempo é geralmente menor que o do NVS, mas suficiente para que todos os dados novos presentes no cache sejam gravados em disco físico, e os mesmos desligados de uma forma ordenada. Desta forma, os dados estão garantidos independentemente da duração da falta de energia.

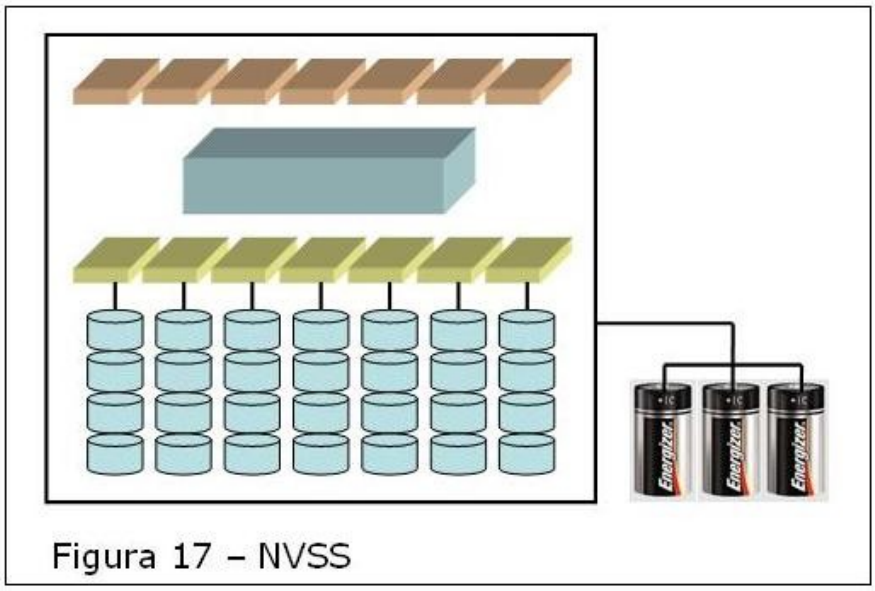


Figura 17 – NVSS

Além de diferenças na implementação física, os próprios algoritmos de manuseio de dados variam bastante. Algumas controladoras disparam a pré-leitura de dados mais cedo, e carregam valores fixos de dados a cada execução. Outras ajustam dinamicamente estes valores, tentando se adaptar a variações na carga.

Cada fornecedor tem a tendência a supervalorizar as qualidades e vantagens de sua implementação, em detrimento das de seus concorrentes, como dizer que armazenar trilhas inteiras favorece a taxa de *cache-hits*, ou que o armazenamento de blocos utiliza melhor o cache. Entretanto, o fato é que, dentro de certos parâmetros mais favoráveis, todas as implementações funcionam e desempenham bem suas funções. Como também é verdade que é possível a ocorrência de cenários suficientemente desfavoráveis para que qualquer uma delas apresente problemas. É meramente uma questão de adaptação ao ambiente atendido pelo equipamento.

6 – "BackEnds" – a parte oculta dos subsistemas

"Quanto mais complexo o encanamento, mais fácil é entupir o ralo"

Comandante Montgomery Scott
Star Trek II – A procura de Spock

Seja no encanamento de um prédio ou nos barramentos de um subsistema moderno, o "cano" de menor diâmetro, determina a capacidade máxima de fluxo do sistema. Infelizmente, ao contrário do que acontece na hidrodinâmica, nos subsistemas a "pressão" não é mantida constante, pelo aumento da velocidade no trecho de maior pressão (ou, no nosso caso, no cano mais estreito).

"Backend" é o nome genérico que os diversos fabricantes atribuem à parte de seus equipamentos que fica para "trás" do cache (quando olhados do HOST), ou seja, as conexões ao cache, as placas adaptadoras e os discos físicos. Ainda utilizando as mesmas bases do início, nesta área temos várias "caixas" ou "depósitos" (os discos físicos), os quais se conectam via encanamentos a um reservatório central (o cache), e onde o "fluxo" é regulado por diversas "válvulas" (as placas adaptadoras).

Implementados de forma distinta por cada fabricante, os backends são os responsáveis pela taxa final sustentável de transferência de dados que um subsistema é capaz de fornecer. Conhecer seus pontos de estrangulamento permite distinguir entre engenharia e propaganda.

Tipos de Conexão

Nesta área temos que considerar a conexão de duas áreas distintas. Uma é a que interliga os discos físicos às placas adaptadoras, conforme discutido no Capitulo 2 – Interfaces a disco. Uma outra é a que conecta as placas adaptadoras às estruturas de CACHE do subsistema. As implementações mais comuns são a de BARRAMENTO COMPARTILHADO (EMC Symmetrix 5.5 e anteriores), BARRAMENTO PCI (IBM ESS), e CHAVEADO (HDS 9980V Lightning).

Todas as arquiteturas atuais impõem a passagem de dados pelo CACHE, seja durante a leitura ou gravação. Portanto, um primeiro fator limitador de banda de backend é o número e a velocidade das conexões a ele. Por mais que o restante das ligações sejam rápidas, caso seu equipamento só disponha de 2 barramentos PCI para conexão ao CACHE, por exemplo, o máximo de taxa sustentável será de 133 MB/S (qualquer operação requer pelo menos dois acessos ao CACHE, uma entre o HA ou host adapter e o CACHE, e a outra entre ele e os discos, portanto, dos 2X133 disponíveis, somente a metade pode ser considerada – veja abaixo).

Em relação às arquiteturas de barramento, o ponto importante é que apenas uma operação pode ser conduzida de cada vez, pois todos funcionam com implementações de BUS MASTERING, o que exige arbitragem e serialização do

recurso, restringindo o número máximo de operações SIMULTÂNEAS ao número total de barramentos disponíveis. No ambiente chaveado, o número de portas ao CACHE torna-se o fator limitante, uma vez que independentemente da quantidade de ligações que um SWITCH possa ter, uma dada área de CACHE somente poderá estar efetuando UMA transferência de dados de cada vez.

Em alguns casos (HDS e EMC), cada placa de CACHE dispõe de ASICs individuais para mais de uma região (4 para os exemplos acima), o que permite que 4 comunicações ocorram simultaneamente por placa. Se quiser saber o limite de operações simultâneas de seu backend (sustentáveis), basta multiplicar o número de regiões individuais de cada placa, pelo número disponível de placas de CACHE de seu equipamento.

Operações de HOST X Operações de BACKEND

O número de operações executadas pelo backend é, geralmente, diferente do número de operações enviadas pelo HOST, e deve ser considerado desta forma quando se dimensiona a capacidade necessária em um equipamento.

Tomando-se por exemplo uma operação de leitura, na qual os dados encontram-se no CACHE (READ HIT), os dados terão de fluir pelo equipamento apenas uma vez (em função da operação, uma vez que a leitura dos discos ao CACHE foi causada ou pelo algoritmo de pré-carga ou por uma outra operação). Neste caso, há uma correspondência 1X1 entre os dois. Entretanto, já um READ MISS fará com que :

1 - os dados sejam lidos do disco ao CACHE e
2 - do CACHE ao HOST ADAPTER.

Caso ambos os fluxos compartilhem dos mesmos caminhos (como no Symmetrix 5.5 e anteriores), a banda total da conexão tem de ser dividida por 2.

Uma operação de gravação, por sua vez, é influenciada por fatores como o esquema de proteção escolhido para o equipamento e o tamanho do bloco a ser gravado. No caso de RAID-0, como os dados são divididos entre um grupo de discos, haverá tantas operações de gravação quantas forem necessárias ao tamanho do bloco sendo gravado (p.ex. para um Blksize de 32K e um StripeSize de 8K serão necessários 4 acessos ao disco, mais a movimentação dos dados do HA ao CACHE, totalizando 5 acessos). Para um subsistema protegido por RAID-1, uma operação de gravação totaliza 3 movimentações de dados pelo subsistema. Uma do HA ao CACHE, e uma para cada cópia em disco.

Como regra geral, pode-se manter a relação 1X1 para CACHE HITS, 1X2 para operações CACHE MISS e gravações em RAID-1 e 1X5 para gravações em RAID-4 e 5. Equipamentos em RAID-6 apresentam relações maiores, caso não seja utilizada qualquer forma de atenuação do mesmo (p.ex. o LOG STRUCTURED FILE do equipamento STK, que sempre grava blocos inteiros, evitando as penalidades de leitura de dados antigos e paridade). A nova implementação de PARITY RAID do EMC DMX utiliza circuitos nos próprios discos para gerar a paridade, resultando em um impacto 50% maior que o RAID-1 (3 operações comparadas com 2).

Operações em Placas Adaptadoras

Conforme visto na primeira unidade, cada disco físico é capaz de efetuar por volta de 150 operações por segundo em um ambiente típico de produção. Quantos

então poderiam ser interligados a uma placa adaptadora, de forma que as limitações físicas desta última não implicassem em enfileiramentos aos mesmos ?

Experimentalmente, o número a ser usado vai de 1600 a 2200 operações por segundo, variando de acordo com o fabricante e modelo do equipamento considerado. Equipamentos de última geração tendem a suportar valores próximos ao maior limite desta faixa. Portanto, se considerarmos 130 iops nos discos físicos e uma maquina de tecnologia média (1800 iops, p.ex.), 14 discos físicos seriam o limite para esta placa.

Um ponto importante aqui é que há implementações nas quais uma mesma CPU de placa adaptadora é responsável por responder às operações de mais de um grupo de discos (por exemplo, dois barramentos SCSI em uma mesma CPU). Neste caso, o número a ser considerado é o total para a CPU da placa. No caso de nosso exemplo acima, 7 discos por barramento, no máximo.

Melhorias de backend

Há duas formas básicas de se obter melhorias de backend. Uma é acelerando suas operações (barramentos ou discos mais velozes, por exemplo), e a outra é desengargalando estas operações (mais barramentos ou mais discos, seguindo a linha do exemplo anterior). Mesmo sem a aquisição de novos equipamentos ou componentes, ainda é possível melhorar o desempenho de um subsistema, utilizando-se técnicas tão antigas quanto a dispersão dos volumes mais acessados.

Ainda hoje a regra 80/20 se mantém (80% das operações são feitas contra 20% dos volumes), o que permite que as unidades com maior concentração de carga sejam espalhadas pelo sistema mais facilmente. Caso as opções de proteção tendam a concentrar o acesso de volumes lógicos em poucos físicos, um SORT decrescente pela coluna DEVICE ACTIVITY RATE do RMF-I PP DASD REPORT basta para isso. Tal dispersão, abrangendo também unidades de controle, canais e partições distintas, otimiza o acesso e evita o enfileiramento de operações por falta de recursos. É importante ter em mente que assim como qualquer equipamento pode ser "afinado" para o melhor desempenho de sua arquitetura, também pode ser exposto a um cenário no qual apresentará problemas. Não há uma única solução "certa" ou "melhor" para todo e qualquer ambiente.

Todo e qualquer subsistema pode apresentar atrasos de backend. Braços mecânicos se movendo, setores de disco girando para a posição desejada, barramentos SCSI ou links FC-AL ocupados, placas de CACHE com todas as regiões já atendendo outras operações são apenas alguns dos exemplos do que pode causar isso. Quanto mais baixa for a taxa de CACHE-HIT do ambiente, mais grave o problema tende a se tornar. E com tantas implementações distintas, utilizando diferentes tecnologias, como avaliar o desempenho ? A resposta é simples : use sempre a mesma moeda !

Conforme descrito no Capitulo I – Discos Físicos, uma operação de I/O a um destes dispositivos pode levar tempos em torno de 15 mS (9,70 Full Stroke Seek + 5,98 Max.Latency). Concedendo-se acréscimos devido a atrasos de protocolo, repetição de operações por formatação, etc, é razoável utilizar-se, como unidade padrão de medida algo em torno de 22mS (quase 38% de acréscimo sobre os tempos mecânicos). De fato, tempos por volta de 25mS são os mais comumente medidos para operações a disco em grandes subsistemas. Pode-se optar por um ou outro, mantendo-se em mente que este valor aplica-se à controladoras utilizando os discos de exemplo do primeiro capitulo. Outros dispositivos, com tempos

diferentes, devem ter este valor recalculado. Para efeitos deste trabalho, vou me referir a este tempo como DISK IOTIME.

Portanto, assumindo-se que 22mS seja o DISK IOTIME apropriado para o subsistema em análise, uma outra informação à ser obtida é o REAL DISCONNECT TIME para o ambiente.

Da forma como é reportado pelo RMF, há uma certa tendência a se interpretar de forma incorreta o tempo de DISC de uma unidade ou controladora. Veja o exemplo abaixo :

D I R E C T A C C E S S D E V I C E A C T I V I T Y

z/OS V1R2 SYSTEM ID CPUA START 05/07/2004-05.00.00 INTERVAL 000.59.59
RPT VERSION V1R2 RMF END 05/07/2004-06.00.00 CYCLE 1.000 SECONDS

TOTAL SAMPLES = 3,600 IODF = A2 CR-DATE: 04/06/2004 CR-TIME: 12.14.02 ACT: POR

STORAGE GROUP	DEV NUM	DEVICE TYPE	VOLUME SERIAL	PAV	LCU	DEVICE ACTIVITY RATE	AVG RESP TIME	AVG IOSQ TIME	AVG CMR DLY	AVG DB DLY	AVG PEND TIME	AVG DISC TIME	AVG CONN TIME	% DEV CONN	% DEV UTIL	% DEV RESV	AVG NUMBER ALLOC	% ANY ALLO
	A0E4	33901	UPG51E		0071	0.001	2.4	0.0		0.0	2.2	0.0	0.2	0.00	0.00	0.0	0.0	100.
	A0E5	33901	UPG51F		0071	0.001	0.8	0.0		0.0	0.5	0.1	0.2	0.00	0.00	0.0	0.0	100.
	A0E6	33901	UPG520		0071	0.001	0.4	0.0		0.0	0.2	0.0	0.2	0.00	0.00	0.0	0.0	100.
	A0E7	33901	UPG521		0071	0.001	0.5	0.0		0.0	0.3	0.0	0.2	0.00	0.00	0.0	0.0	100.
	A0E8	33901	UPG522		0071	0.001	0.5	0.0		0.0	0.3	0.0	0.2	0.00	0.00	0.0	0.0	100.
	A0E9	33901	UPG523		0071	0.001	1.0	0.0		0.0	0.7	0.0	0.2	0.00	0.00	0.0	0.0	100.
	A0EA	33901	UPG524		0071	0.001	0.5	0.0		0.0	0.3	0.0	0.2	0.00	0.00	0.0	0.0	100.
			LCU		0071	405.151	15.4	2.0		0.0	1.9	7.0	4.6	0.78	1.99	0.0	547	100.

Figura 18 – DASD REPORT

A LCU 0071 apresenta para este período (05:00:00 – 06:00:00 do dia 07/05/2004) e esta partição (CPUA) um tempo médio de DISC de 7.0 mS, para uma taxa de operações de mais de 405 iops. Entretanto, este NÃO É o tempo REAL de DISC destas operações, por uma razão muito simples. As operações CACHE-HIT não têm DISC, e deveriam ser descontadas deste valor. Veja a figura a seguir

C A C H E S U B S Y S T E M A C T I V I T Y

z/OS V1R2 SYSTEM ID CPUA START 05/07/2004-05.00.00 INTERVAL 000.59.59
RPT VERSION V1R2 RMF END 05/07/2004-06.00.00
SUBSYSTEM 3990-06 CU-ID A000 SSID A000 CDATE 05/07/2004 CTIME 05.00.00 CINT 00.59.59
TYPE-MODEL 3990-006

CACHE SUBSYSTEM DEVICE OVERVIEW

VOLUME SERIAL	DEV NUM	RRID	% I/O	I/O RATE	---CACHE HIT RATE-- READ	DFW	CFW	----------DASD I/O RATE------ STAGE	DFWBP	ICL	BYP	OTHER	ASYNC RATE	TOTAL H/R	READ H/R	WRITE H/R	% READ
*ALL			100.0	566.7	341.5	158.3	0.0	45.4	0.0	5.5	15.9	0.0	0.0	0.882	0.886	0.992	70.7
*CACHE-OFF			0.0	0.0													
*CACHE			100.0	566.7	341.5	158.3	0.0	45.4	0.0	5.5	15.9	0.0	0.0	0.882	0.886	0.992	70.7
DB2061	A000	N/A	0.0	0.1	0.0	0.0	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.295	0.289	1.000	96.7
DB2062	A001	N/A	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.554	0.548	1.000	97.8
DB2063	A002	N/A	1.4	7.9	6.5	0.0	0.0	1.4	0.0	0.0	0.0	0.0	0.0	0.824	0.824	1.000	99.8
DB2064	A003	N/A	0.0	0.2	0.0	0.1	0.0	0.1	0.0	0.0	0.0	0.0	0.0	0.578	0.340	0.997	56.1
DB2065	A004	N/A	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	N/A	N/A	N/A	N/A
DB2066	A005	N/A	0.2	1.0	0.1	0.0	0.0	0.0	0.0	0.0	0.9	0.0	0.0	0.067	0.822	1.000	97.9
DB2067	A006	N/A	0.2	1.2	0.9	0.0	0.0	0.2	0.0	0.0	0.0	0.0	0.0	0.786	0.789	N/A	100.0
DB2068	A007	N/A	7.2	40.9	5.3	34.8	0.0	0.5	0.0	0.0	0.4	0.0	0.0	0.980	0.963	0.992	13.5

Figura 19 – CACHE REPORT

Esta mesma controladora (para o range A000 – A0FF), no CACHE REPORT é identificada pelo seu CU-ID A000 (endereço do primeiro dispositivo a ser acessado durante a ativação do subsistema), apresenta um CACHE-HIT RATIO de 88.2%, o que significa que das 405 operações por segundo, somente 11,4% ou 46.2 iops sofreram qualquer atraso por DISC. Portanto, para determinação do REAL DISC, deve-se desconsiderar as outras operações na hora de dividir o tempo acumulado, conforme os passos descritos a seguir (para o RMF de exemplo)

1. Calcula-se o DISC TOTAL do ambiente, multiplicando-se o IORATE pelo AVG DISC – no caso do exemplo, 405.151 X 7 = 2836.057 mS
2. Extrai-se a %CACHE-HIT do CACHE SUBSYSTEM ACTIVITY REPORT, coluna TOTAL H/R. No nosso exemplo, 0.882 ou 88.2% de operações em CACHE
3. Calculá-se o número de operações por segundo que efetivamente sofreram DISC, como a porcentagem de CACHE MISS acima. No caso, 11.4% de 405 ou 46.2 iops

4. Determina-se o valor REAL de DISCONNECT, dividindo-se o DISC TOTAL pelo numero de operações CACHE MISS. Do exemplo, 2836.057 / 46.2, ou 61.4264.

Portanto, o que o RMF indica como um DISC TME médio de 7.0 mS na verdade foi causado pela distribuição dos 61.4 mS de DISC que cada uma das 46 operações CACHE-MISS sofreu, entre as 405 operações efetuadas no período.

É importante lembrar que as controladoras reportam informações de CACHE para a IMAGEM COMO UM TODO, sendo incorreta a extração deste relatório por todas as partições de um ambiente data-share. Como o próprio manual do RMF informa, ele deve ser tirado por APENAS UMA das partições que tem acesso ao subsistema. No exemplo acima, assumimos que apenas a partição CPUA tinha acesso ao subsistema, e, portanto, seus dados são válidos para comparação com o DASD REPORT.

Agora, de posse destes dois valores, DISK IOTIME (para nosso exemplo, 22mS), e o REAL DISC TME (61,4mS), podemos definir um consenso de medida que chamo de BACKEND QUEUING. Ou seja, enfileiramento de backend.

Semelhante ao IOSQ que determina quantas operações ficam, na media, aguardando enfileirados na UCB aguardando seu SSCH, o BK-Q determina, em média, quantas operações ficam aguardando execução. Para determiná-lo, basta dividir-se o REAL DISC TME pelo DISK IOTIME. No nosso caso (61,4 / 22) temos 2,792109, ou 2.8 IOs na fila do backend, em média, para aquele período.

Este valor, por tratar exclusivamente de cargas CACHE-MISS é bem mais apropriado à avaliação de placas adaptadoras, discos e barramentos, que, por exemplo, o RT médio. A sugestão aqui é que o BK-Q fosse calculado para cada subsistema presente no ambiente, e para cada período (BATCH, ONLINE, FECHAMENTO, etc), de forma a se ter uma referencia para casos da performance piorar. Se o RT aumenta, por exemplo, foi a CARGA TOTAL que aumentou, o CACHE HIT que baixou, ou o BKND da máquina que atingiu seu ponto de saturação ? Para tal controle, pode-se plotar uma curva comparando IORATE por BK-Q, por exemplo, e fica-se sabendo de antemão quando o cotovelo é atingido. Uma curva 3D seria ideal, comparando IORATE, CACHE-HIT e BK-Q.

Mais que isso, o BK-Q serve também para se comparar resultados entre arquiteturas completamente distintas. Se um fornecedor possui BKND FC-AL com discos de 45Krpm e o outro somente SCSI com 7.2 ATA drives, mas os dois lhe dão o mesmo nível de BK-Q para uma mesma carga (e o mesmo cache-hit, é claro), do ponto de vista de desempenho, o ATA substituiria o outro tranqüilamente, e geralmente com significativos ganhos financeiros !

Uma outra utilidade é avaliar as especificações de um novo produto. Caso o fornecedor garanta que seu novo equipamento é, digamos, duas vezes melhor que o anterior ou o competidor, o BK-Q pode fazer parte dos números apurados para averiguação destas afirmações.

7 – Canais ao MainFrame

Ao final dos anos 60, os grandes frames do IBM/360, que executavam a função de interconectar as CPUs às unidades de controle dos dispositivos de entrada e saída (como o 2560 – canal seletor e o 2570 – byte multiplexor), começaram a ser modificados, e uma "nova" linha de canais paralelos foi introduzida. Chamado de *OEMI – (Original Equipment Manufacturer Interface)*, este padrão iniciava uma nova era de conectividade.

O padrão OEMI baseava-se na utilização de dois cabos, um chamado de BUS e o outro, TAG. Ambos continham vários fios de cobre em seu interior, envoltos por camadas protetoras que os tornavam muito rígidos e de difícil manuseio. As linhas eram divididas em dois grupos, sendo um o grupo IN (na direção da CPU) e o outro designado OUT (na direção da unidade de controle). O cabo de BUS carregava dados que eram identificados pelas combinações de linhas ativas no TAG. Para enviar um comando a uma controladora, por exemplo, a CPU colocaria o comando nas linhas de BUS-OUT (02 – read, por exemplo), e ativaria a linha de COMMAND-OUT no cabo de TAG. Este padrão permitia um tipo de conexão chamada de *daisy-chainning* ou *multi-drop*, na qual o mesmo CANAL poderia ser conectado à varias controladoras serialmente, sendo terminados por uma peça especialmente desenhada para isso (TERMINATOR). Ao tentar se comunicar com uma controladora em particular, o endereço da controladora em questão era colocado no BUS-OUT, e identificado pelo TAG ADDRESS-OUT. A controladora com o endereço especificado reconheceria a seleção e iniciaria a comunicação com a CPU, enquanto todas as outras aguardavam.

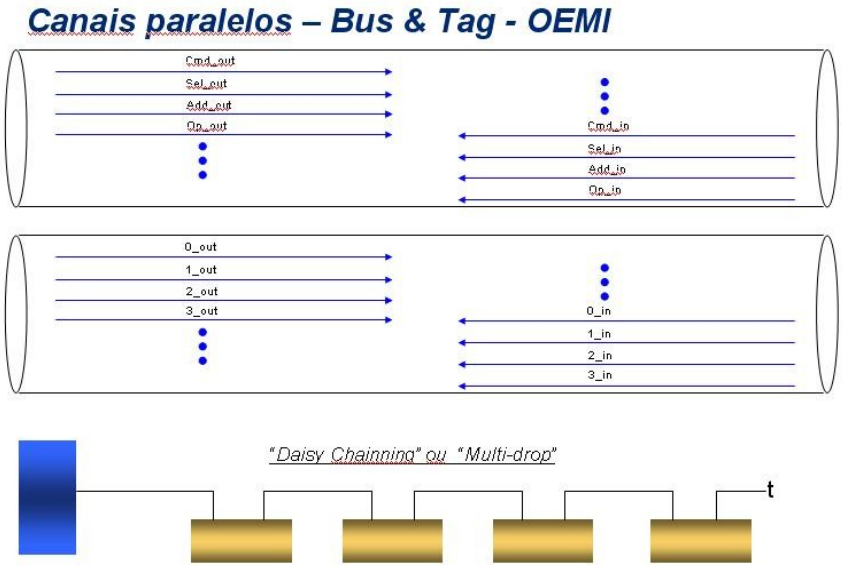


Figura 18 – BUS & TAG

O padrão OEMI também especificava o protocolo de comunicação a ser seguido utilizando-se seus meios físicos, o qual ficou conhecido como *Mother-May-I* ("mamãe-eu-posso?"), basicamente por ser bastante simples e exigir confirmações e reconexões constantes. Neste protocolo, as transmissões eram iniciadas com a fase de seleção inicial, que identificava a unidade de controle e o dispositivo alvo da operação, depois o envio de um comando *CCW – Channel Command Word*, o qual era interpretado pela unidade e executado pelo dispositivo final. Terminada a operação, o dispositivo e a controladora deveriam indicar os seus terminos através do envio de fim-de-operação (*CE – Channel End, DE – Device End)*, quando então

uma outra operação poderia ser iniciada. Apesar de parecer estranho nos dias de hoje, esta maneira de coordenar as operações era bastante apropriada às controladoras da época, dotadas de pouquíssima capacidade de processamento (*dumb controllers*).

Enterprise Systems CONnectivity

Anunciado em 5 de Setembro de 1990, o ESCON foi uma mudança bastante radical na forma de se executar operações de entrada e saída nos ambientes MainFrame, sendo a primeira grande modificação deste aspecto do ambiente feita em 25 anos, durante os quais apenas pequenas evoluções do OEMI foram apresentadas.

Os anos 90 assistiram a notáveis mudanças no ambiente MainFrame, incluindo-se o inicio dos novos processadores C-MOS que atualmente dominam o mercado. Também na área de conectividade à dispositivos de E/S uma grande revolução estava se iniciando, com a utilização de canais de fibra de vidro e transmissões de sinal por luz de LASER ou LED.

A luz se propaga a 300.000 Km/S no vácuo, e a aproximadamente 200.000 Km/S em uma fibra de vidro, o que impõe um atraso de aproximadamente 5 uS (microssegundos) por quilômetro a qualquer transição de sinal feita em sua entrada. Apesar de teoricamente ser possível o envio de quaisquer freqüências de sinal por um meio físico qualquer, experimentalmente determinou-se que os comprimentos de onda de 850 e 1300 nM (nanômetros) eram os que sofriam menos atenuação, dai serem os utilizados atualmente, movidos a LED ou LASER de potencias da ordem de 10 a 50 mW (miliwatts). A largura de banda destas mídias define-se em H/Km (Hertz por Kilometro), sendo que 500 MHz/Km é uma figura típica para fibras chamadas de multi-modo, enquanto que 100 GHz/Km é a mais característica de single-modes. As fibras multi-modo são de maior diâmetro, recebendo a luz em um ângulo diferente de seu eixo central, o que causa a propagação em padrão de zig-zag, enquanto que as single-mode, bem mais estreitas, propagam a luz em linha reta.

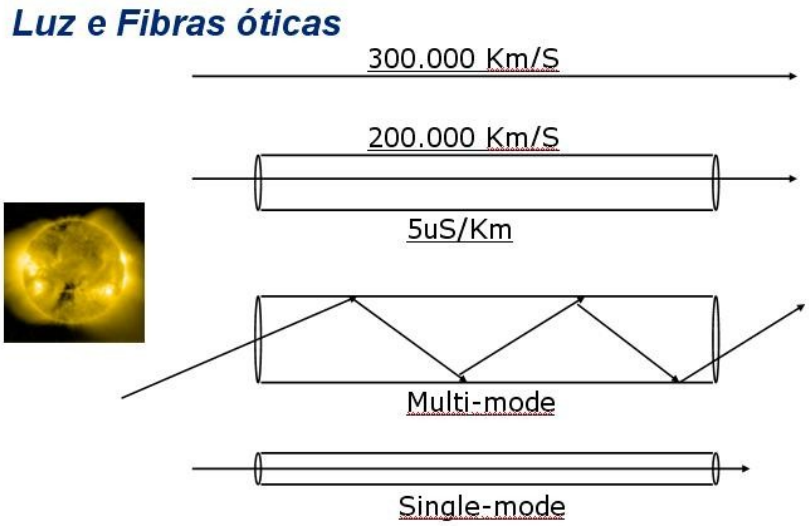


Figura 19 - Luz e Fibras

Os antigos OEMI transmitiam dados a taxas da ordem de 3 a 4.5 MB/S, e eram limitados na distância que poderiam atingir, devido à degradação do sinal eletrônico, facilidade em captar ruídos ambientes e a dificuldade em manter o

alinhamento (*data skew*) entre todos as linhas de sinal. A solução destes problemas físicos veio com a utilização de um par de fibras de vidro envoltas em camadas protetoras, muito mais finas e leves que os cabos paralelos, e que permitiam o trafego de dados a taxas de 20 MB/S. Por ser uma comunicação serial, não sofria com alinhamento. Por ser óptico, não tinha problemas com motores ou outros campos elétricos presentes nos ambientes pelos quais passasse.

É importante lembrar que os canais ESCON, que vieram resolver problemas da camada física, ainda estavam sujeitos às mesmas limitações do protocolo anterior (Mother-May-I), o qual não foi substancialmente modificado. Portanto, um canal ESCON executa apenas UMA OPERAÇÃO DE CADA VÊZ, e somente em um sentido, apesar de ser composto por duas fibras (a outra é usada apenas para sinalizações de controle). Esta forma de conexão é conhecida como *circuit-switched*, na qual todo o circuito é "chaveado" (ligado ou desligado) para um determinado usuário, e enquanto estiver assim, nenhum outro pode acessá-lo. Assemelha-se bastante às antigas mesas de telefonistas, nas quais a operadora ligava fisicamente os fios de um aparelho a outro, e enquanto durasse a ligação, o canal não poderia ser utilizado por mais ninguém. Neste caso, o tempo medido como CONNECT TME pelo RMF é uma medição de ocupação da "linha", e não de utilização ou transferência real de dados (como se você mantivesse sua ligação telefônica sem dizer nada).

O uso de novos hardwares por protocolos antigos resultou em formas de atraso novas. Uma delas, chamada de *ESCON droop* é característica dos canais ESCON a grandes distâncias. Este fenômeno ocorre devido ao atraso imposto ao sinal de luz pela distância a ser percorrida na fibra, aliado ao fato do protocolo OEMI aguardar confirmações para continuar sua transmissão.

Sobre canais ESCON, os tempos através dos quais se mede uma operação de I/O, elaborados durante o período do OEMI, continuam claramente delimitados. O período de PEND é definido como todo o tempo entre o recebimento do comando SSCH vindo da CPU até que o primeiro CMR (Command Response) chegue do dispositivo desejado. Ele representa a soma de todos os atrasos causados por falta de recurso de acesso ao mesmo, como canais, portas de chaveamento, controladoras e dispositivos (ocupados com operações de outras partições). Uma vez recebido o CMR inicial, começa-se a medição do CONN, que inclui todas as transferências de dados necessárias à operação. Este tempo só é interrompido por uma eventual necessidade da controladora de executar uma operação mecânica no dispositivo, durante a qual, por levar relativamente muito tempo, ela se desconecta para otimizar o uso do canal.

FIber CONnectivity – o padrão para os dias atuais

O admirável mundo novo trouxe também ao MainFrame grandes inovações tecnológicas, como suas mais novas CPUs ou CECs, como são chamados atualmente. Estas novas máquinas, capazes de velocidades acima de 2000 MIPS por FRAME, vieram também acompanhadas de atualizações em seus sistemas de E/S, como subsistemas de canal adicionais e mais rápidos.

Para estes novos ambientes, os (agora) antigos ESCON começavam a impor serias limitações. Dotados de um máximo de 17 MB/S úteis e uma única operação de cada vez, e tendo de acessar subsistemas cada vez mais complexos e rápidos, como os de armazenamento, tornaram-se pontos de estrangulamento, chegando a ponto de várias instalações verem-se na situação de aumentar o número de CECs somente para dispor de banda adicional de conexão. É neste cenário que os canais FICON entram em cena, utilizando novos protocolos de comunicação, e

aumentando em várias vezes a capacidade de comunicação de cada porta disponível nas modernas CPUs.

FICON é uma implementação no nível 4 do protocolo FIBRE (*ULP – UPPER LEVEL PROTOCOL)*, chamado de FC-SB-2. Baseado em *SBCSS – Single-Byte Command Code Set,* ele equivaleria, a grosso modo, à camada das aplicações em um ambiente IP. Isso significa que tanto canais FICON quanto canais FIBRE compartilham as características das camadas mais baixas *(FC3 – Common Services, FC2 – Signalling, FC1 – Transmission e FC0 – Interface and Media)*.

Portanto, pode-se dizer que muito mais que uma mudança física, o canal FICON implementa uma mudança de protocolo de comunicação sobre o antigo OEMI. Um canal FICON executa várias operações simultaneamente nos dois sentidos (FULL-DUPLEX), a velocidades nominais de 100 ou 200 MB/S, tendo "pacotes" das várias operações fluindo simultaneamente em suas fibras (PACKET SWITCH ao invés do CIRCUIT SWITCHED do ESCON, ou conexões TDM).

Agora, o CONN TME é medido como o tempo desde que o primeiro pacote foi enviado pela fibra até que o último pacote tenha sido recebido, acrescido dos tempos de todos os outros pacotes de outras operações transmitidos entre eles. Isso muda radicalmente a forma de se entender dados de RMF.

As informações utilizadas para o acesso a ambientes mainframe, como endereço do dispositivo, número de CCW, dados e controles são incluídos na parte da DIB (DATA INFORMATION BLOCK), que o SB-2 utiliza para este mapeamento, consumindo parte da área de dados.

Cada canal FICON dispõe de 32 entidades chamadas de *OpenExchanges*, cada uma delas podendo conduzir uma operação independentemente das outras. As OEs podem ser entendidas como o nível de multi-programação suportado pelo canal. Quando os 32 OEs estão ocupados, novas operações são enfileiradas na *Initiative Queue* da unidade de controle, como sempre foram, gerando PEND TME por CHANNEL BUSY. Cada OE dispõe de até 16 *IU´s – FC-4 ANSI FC-FS Information Units*, que correspondem a um FICON FRAME (ou a um Fibre Channel FC-2 layer FRAME) de 2148 BYTES, dos quais 2112 são para a formatação SB2, e o restante são headers e controles das camadas físicas (FC-2). Destes, o protocolo SB2 utiliza 64 bytes para cabeçalhos e controles, deixando 2000 BYTES para comandos ou dados em cada FRAME. Quando transmitindo dados, cada OE pode grupar suas IUs em unidades de até 8KB chamadas de FICON WORK UNITs, as quais ocuparão de 1 a 4 créditos (áreas de buffer do canal). Cada canal FICON dispõe de até 64 créditos, os quais, quando em uso, serão ocupados por IUs de suas diversas OEs. A cada novo SSCH, uma nova OE é alocada no canal, e a primeira IU de CCW enviada ao dispositivo. Ao final (recebimento da IU de FINAL STATUS), a OE é liberada.

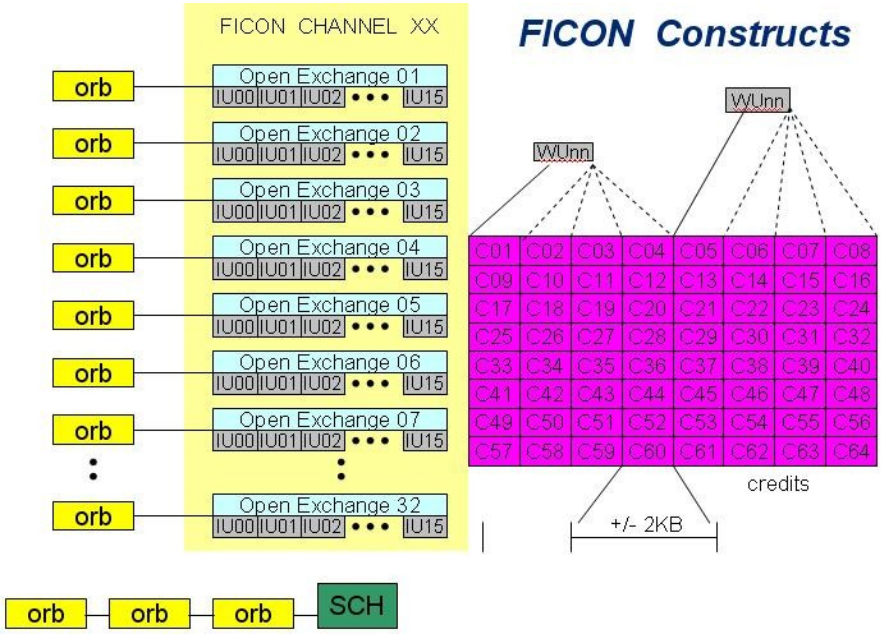


Figura 20 – FICON Constructs

Um canal FICON opera de modo diferente de um ESCON. Caso nada seja especificado em contrário, sua operação normal é a de enfileiramento de comandos (*CCW Pipelining)* e não requisição de sincronismos (*No-Synch)*, o que significa que TODAS AS CCWs de um programa serão enviadas ao dispositivo alvo (respeitadas as limitações de créditos), e somente ao término de todo o programa a controladora retorna um STATUS final.

Esta é uma característica que permite ao FICON chegar mais longe. Sem as idas-e-vindas do protocolo anterior, os dados e comandos são transmitidos às taxas nominais, sem que uma das partes tenha que aguardar a confirmação da outra, atrasada pela maior distância (*droop)*. Caso a aplicação assim o deseje, o canal FICON ainda pode operar como o MOTHER-MAY-I do OEMI, especificando isso em seu pedido de operação (ORBP/Y/M=0). Desta forma, os programas de canal não tem de ser modificados, exceto os que dependem de informações intermediarias sobre o andamento de suas operações (INTERMEDIATE STATUS).

O significado dos números muda quando se analisa um canal FICON. Um deles é o CHANNEL BUSY %. Como o FICON está sempre transmitindo IUs, quer contenham dados ou não, há sempre uma porcentagem de atividade, da ordem de 10%, mesmo que nada esteja ativo neste canal. A %BUSY para o ambiente FICON é relação entre o número de FRAMES que contenham dados, pelo número TOTAL de FRAMES que podem ser transmitidos (estimativa de laboratório). Portanto uma outra relação que não funciona mais é a da SOMA dos CONN TME ser igual à %BUSY. Estes valores hoje não apresentam mais qualquer relação entre si.

Topologias

As três topologias principais suportadas pelos canais FICON são ponto-a-ponto, ponto-a-ponto chaveada e encadeamento de directors.

Na ponto-a-ponto, dois nodes são interligados diretamente. Não há chaveamento, os cabeçalhos FC-2 não precisam conter tantas informações de endereçamento, e o trafego total do canal é somente o gerado pelo canal e seu dispositivo.

No caso da ponto-a-ponto chaveada, a parte das conexões que fica entre o SWITCH e os dispositivos apresenta um tráfego que DEVERÁ SER SOMADO para se obter o tráfego total entre o SWITCH e o canal. É importante lembrar que numa ligação CONTROL-UNIT => SWITCH, apesar de não haver a carga de outras controladoras, poderá haver a carga de várias operações simultâneas (OEs).

Nesta topologia, dois SWITCHES são interligados através de um ISL (Inter-Switch link) de forma a aumentar a conectividade. Este link é um canal da mesma forma, e deve ser dimensionado de forma a suportar TODA a carga vinda da CPU para os dispositivos acoplados ao segundo SWITCH.

A estrutura de programação é em grande parte mantida, sendo a operação iniciada pela aplicação através de MACROs ou SVCs, que causam a emissão de uma SSCH (START SUBCHANNEL INSTRUCTION), com seu ORB (OPERATION REQUEST BLOCK) apontado via GPR1. Via bits da ORB (P, Y e M) pode-se especificar que um canal FICON funcione como um ESCON. Se não forem desligados, o encadeamento de dados e comandos, além da não-exigência de sincronização, características do FICON, serão seguidas. O nível 4 do protocolo (SB-2) se incumbirá da criação das IUs, as quais serão posteriormente formatadas e enviadas pelo link de acordo com as especificações dos restantes dos níveis FIBRE.

O canal opera diferentemente, portanto, os tempos tem significados diferentes. No FICON, PEND TME é o tempo medido pelo canal entre o recebimento da SSCH e o retorno do CMR (COMMAND RESPONSE) inicial do dispositivo. CONN TME é medido pelo canal, começando no início do envio da primeira IU daquela operação, até o término do envio da última. Dada a característica de operação do link, este tempo poderá conter também os tempos de envio de várias IUs de várias outras operações, não significando mais tempo de ocupação, como nos ESCON. Já o DISC TME é medido PELA CONTROLADORA, e enviado junto com o STATUS final da operação. Este tempo é então subtraído ao inicialmente medido como CONN TME pelo canal, e reportado via RMF.

Como qualquer hardware inserido no ambiente, os SWITCHES ou DIRECTORS também apresentam atrasos, e seus números podem ser medidos via RMF 74, subtipo 7. Uma informação importante em relação a eles é que, como parte do protocolo FIBRE, um SWITCH pode descartar FRAMES recebidos em excesso à sua capacidade de enfileiramento, desde que informe o remetente do mesmo. No caso dos sistemas MainFrame, este descarte de FRAMES é interceptado e gera uma indicação de ICC – INTERFACE CONTROL CHECK no CHANNEL SUBSYSTEM, que, no caso, não significa um erro, mas sim um alerta. Esta indicação deve ser modificada no futuro.

Quando uma topologia chaveada é utilizada, ocorrerão atrasos naturais entre os FRAMES que convergem a um mesmo link. Isso se assemelha ao engarrafamento causado por várias vias convergindo a uma única avenida de mesmo tamanho. Este efeito deve ser levado em conta quando dimensionando-se os links FICON de uma instalação, e, é claro, leva-se em consideração a ocupação do link mais congestionado.

Novamente, os números adquirem significado diferente. A coluna "G" indica a geração do canal, 1 para FICON, 2 para FICON EXPRESS. As colunas de utilização indicam a %BUSY da CPU do CANAL, e não a % de uso do mesmo. A coluna BUS indica a porcentagem de vezes em que o barramento foi encontrado em uso, medido contra um limite teórico de utilização (não corresponde a um valor de ocupação medido). Os dados sobre o montante transferido (READ e WRITE) podem ser utilizados para se avaliar uma média de BLOCKSIZE para o canal, quando

dividido pelo CHP TAKEN do IOQ REPORT para todas as LCUs atendidas pelo mesmo.

Uma informação interessante em relação aos dados RMF sobre canais FICON é que, em uma análise correlacional (analise comparativa entre duas séries de dados, na qual -1 significa que são inversamente proporcionais, +1, diretamente proporcionais, e ZERO que não tem qualquer relação entre si), o nível de correspondência entre qualquer um deles e uma curva de tempo de resposta é baixíssimo. Ou seja, não se pode utilizar nenhum deles como referência para prever o comportamento do canal.

Test Series	*% Chan Busy*	*% Bus Busy*	*Read MB/Sec*	*Write MB/Sec*
4K Read-Hit	***88.9***	23.6	23.6	0.0
4K R/W[4]: 50% Hit	15.2	8.7	2.0	2.0
4K R/W: 90% Hit	46.0	16.1	6.9	6.9
HT Read-Hit	58.4	31.5	66.4	0.0
HT R/W: 50% Hit	15.7	16.7	13.1	13.1
HT R/W: 90% Hit	39.9	27.0	24.3	24.3
HT Read Sequential	26.1	21.6	39.0	0.0
HT Write Sequential	26.7	20.3	0.0	29.8

Table 2. Channel Metrics at Saturation for PAI/O Driver Test Series

Test Series	*% Chan Busy*	*% Bus Busy*	*Read MB/Sec*	*Write MB/Sec*
4K Read-Hit	0.01	-0.07	-0.05	0.00
4K R/W: 50% Hit	0.74	0.74	0.65	0.85
4K R/W: 90% Hit	0.76	0.74	0.75	0.75
HT Read-Hit	0.76	0.50	0.55	0.00
HT R/W: 50% Hit	0.74	0.74	0.70	0.78
HT R/W: 90% Hit	0.84	0.59	0.65	0.64
HT Read Sequential	0.71	0.74	0.74	0.00
HT Write Sequential	0.65	0.29	0.00	0.33
All Tests	**0.08**	**0.23**	**0.10**	**0.36**

Table 3. Channel Metrics and Subsystem Service Time Correlation Coefficients

Figura 21 – Resultados de Análise Correlacional

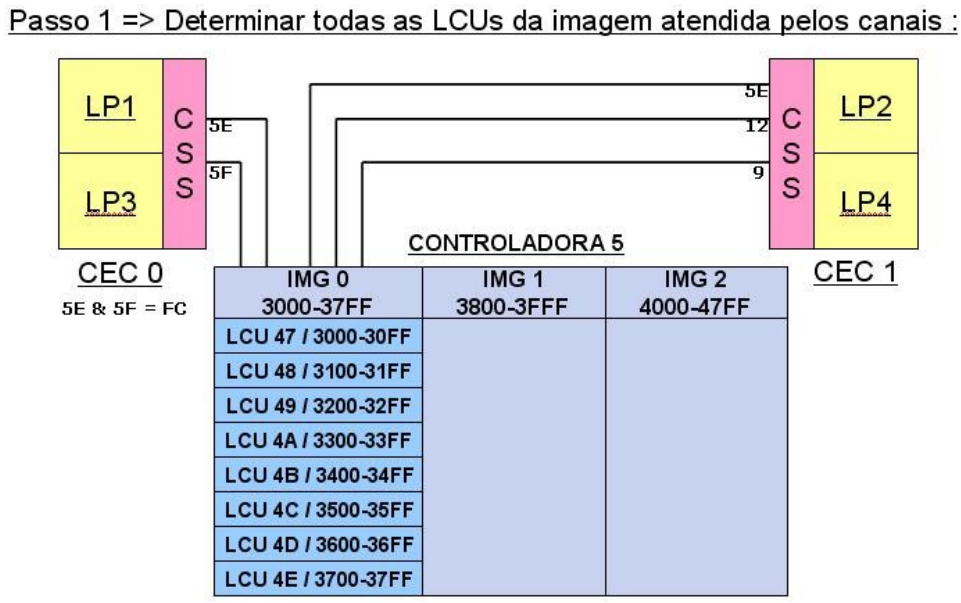
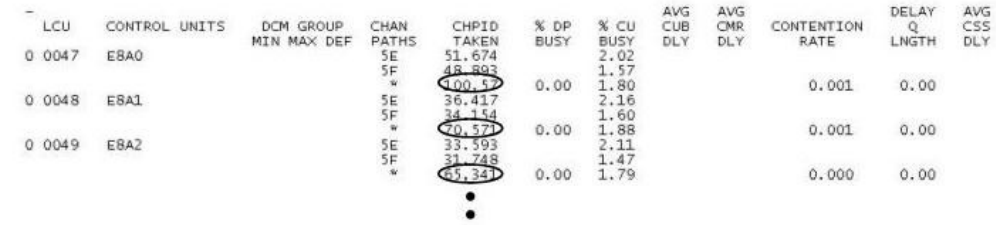


Figura 22 – Ambiente de exemplo

No exemplo acima, deseja-se medir o número de OEs ativos para os canais que atendem os devices da IMAGEM 0 DA CONTROLADORA 5, conforme visto pela partição LP1. Canais (FICON, ESCON ou OEMI) NÃO SE CONECTAM A LCUs. Eles atendem IMAGENS, as quais são grupos, de 2048 ou 4096 unidades (de disco), logicamente divididos em grupos menores, chamados de *Logical Control Units.* A maioria dos relatórios RMF também vem em LCUs, portanto, o primeiro passo é determinar a IMAGEM atendida pelos canais os quais se deseja medir.

Neste exemplo, um subsistema de discos foi dividido em 3 IMAGENS (0, 1 e 2), sendo que a primeira (IMAGEM 0) contem 2048 endereços (3000-37FF), os quais foram subdivididos em 8 LCUs (47, 48, 49, 4A, 4B, 4C, 4D e 4E). Estes LCU IDs são válidos SOMENTE QUANDO VISTOS PELO CEC 0, pois o CEC 1, pelo uso (possível) de um IODF distinto pode atribuir códigos diferentes de LCUs para os mesmos grupos de discos. Ou pior, pode atribuir o mesmo código (p.ex. LCU 49) para um grupo DIFERENTE de discos, o que completa a confusão. Portanto, no caso acima, para os canais e LCUs, devemos usar somente os relatórios referentes ao CEC 0, e os LCU IDs conforme referenciados pela LP1 (por exemplo, o canal 5E visto pela LP2 do CEC1 NÃO É O MESMO que o do CEC 0, mas enxerga os mesmos discos ****)

Passo 2 => Obter o “CHANN I/O RATE da imagem



LCU	CONTROL UNITS	DCM GROUP MIN MAX DEF	CHAN PATHS	CHPID TAKEN	% DP BUSY	% CU BUSY	AVG CUB DLY	AVG CMR DLY	CONTENTION RATE	DELAY Q LNGTH	AVG CSS DLY
0 0047	E8A0		5E	51.674		2.02					
			5F	48.893		1.57					
			*	100.57	0.00	1.80			0.001	0.00	
0 0048	E8A1		5E	36.417		2.16					
			5F	34.154		1.60					
			*	70.571	0.00	1.88			0.001	0.00	
0 0049	E8A2		5E	33.593		2.11					
			5F	31.748		1.47					
			*	65.341	0.00	1.79			0.000	0.00	

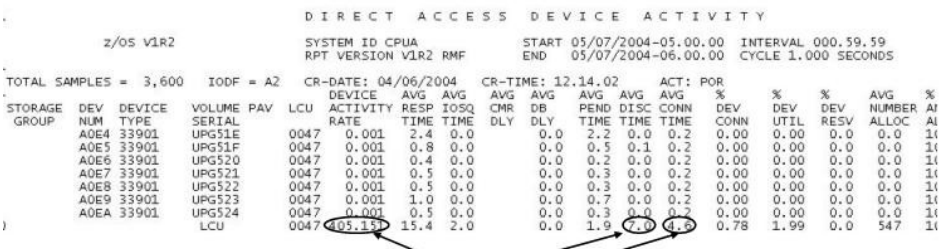
CHPID TAKEN LCU 0047 (100.57)
\+ CHPID TAKEN LCU 0048 (70.571)
\+ CHPID TAKEN LCU 0049 (65,341)
.....
= CHANN IORATE DA IMAGEM
P.EX. 1200 IOPS

Figura 23 – Obtenção do Channel IO Rate

Uma vez determinadas as entidades, conforme o slide anterior, o próximo passo é obter o numero de I/Os por segundo (IORATE) sendo executado pelo grupo de canais em questão (CPG – CHANNEL PATH GROUP). Isso é feito utilizando-se o relatório I/O QUEUING ACTIVITY REPORT do RMF MONITOR I POST-PROCESSOR para o período desejado. Note que esta informação é valida somente para o IORATE iniciado por esta partição. Caso os canais sejam EMIF, você pode optar entre somar os relatórios de TODAS as partições do CEC que tenham acesso ao CPG, ou (o que eu prefiro), utilizar, para informações de canal, a coluna PART ao invés da TOTAL ☺

O IORATE representa a SOMA da linha “*”, para a coluna CHPID TAKEN (iops), o que lhe dá o IORATE por LCU. Somando-se este valor ao das outras LCUs (conforme slide anterior), obtém-se a atividade, em iops, para aquele grupo de canais e para a imagem atendida por aqueles canais. A grande vantagem deste método é que no caso de um SWITCHED POINT-TO-POINT, onde um mesmo canal pode atender varias controladoras (1-n), os números reportados aqui se referem somente à atividade desta controladora. Portanto, uma regra muito importante é “conheça suas LCUs”.

Passo 3 => Obter o “OE DURATION” da imagem



D I R E C T A C C E S S D E V I C E A C T I V I T Y

z/OS V1R2 — SYSTEM ID CPUA — START 05/07/2004-05.00.00 INTERVAL 000.59.59
RPT VERSION V1R2 RMF — END 05/07/2004-06.00.00 CYCLE 1.000 SECONDS

TOTAL SAMPLES = 3,600 IODF = A2 CR-DATE: 04/06/2004 CR-TIME: 12.14.02 ACT: POR

STORAGE GROUP	DEV NUM	DEVICE TYPE	VOLUME SERIAL	PAV	LCU	DEVICE ACTIVITY RATE	AVG RESP TIME	AVG IOSQ TIME	AVG CMR DLY	AVG DB DLY	AVG PEND TIME	AVG DISC TIME	AVG CONN TIME	% DEV CONN	% DEV UTIL	% DEV RESV	AVG NUMBER ALLOC	% AI
	A0E4	33901	UPG51E		0047	0.001	2.4	0.0		0.0	2.2	0.0	0.2	0.00	0.00	0.0	0.0	1(
	A0E5	33901	UPG51F		0047	0.001	0.8	0.0		0.0	0.5	0.1	0.2	0.00	0.00	0.0	0.0	1(
	A0E6	33901	UPG520		0047	0.001	0.4	0.0		0.0	0.2	0.0	0.2	0.00	0.00	0.0	0.0	1(
	A0E7	33901	UPG521		0047	0.001	0.5	0.0		0.0	0.3	0.0	0.2	0.00	0.00	0.0	0.0	1(
	A0E8	33901	UPG522		0047	0.001	0.5	0.0		0.0	0.3	0.0	0.2	0.00	0.00	0.0	0.0	1(
	A0E9	33901	UPG523		0047	0.001	1.0	0.0		0.0	0.7	0.0	0.2	0.00	0.00	0.0	0.0	1(
	A0EA	33901	UPG524		0047	0.001	0.5	0.0		0.0	0.3	0.0	0.2	0.00	0.00	0.0	0.0	1(
			LCU		0047	105.15	15.4	2.0		0.0	1.9	7.0	4.6	0.78	1.99	0.0	547	1(

OE_Duration = DISCimg + CONNimg

LCU 0047 (disc+conn)
\+ LCU 0048 (disc+conn)
\+ LCU 0049 (disc+conn)
...
\+ LCU n (disc+conn)
= OE DURATION
P.EX. 4.0 milisegundos

Figura 24 – Cálculo do OE DURATION

O segundo número para este cálculo vem do DASD ACTIVITY REPORT, também do RMF MON-I PP, e mantidas as mesmas regras do slide anterior. Este número, chamado de "OE DURATION" ou duração de OEs, vem da soma dos CONN e DISC TME para todas as LCUs da imagem.

Portanto, toma-se o AVG CONN da LCU (no exemplo acima, LCU 0047), soma-se ao AVG DISC e obtem-se o tempo MÉDIO de ocupação da OE por cada operação realizada nesta LCU. Repete-se este procedimento para cada LCU da imagem e soma-se os resultados ao final. Este número indica o tempo médio em que os OEs ficaram ocupados por operação para o período medido, seja transmitindo dados (CONN) ou aguardando operações físicas do dispositivo (DISC). Para este calculo não se utiliza o PEND TME (o qual, pela teoria clássica, também faz parte do tempo de serviço) porque o mesmo reporta um período ANTERIOR À ocupação da OE.
Com esta soma, temos o OE DURATION para a imagem.

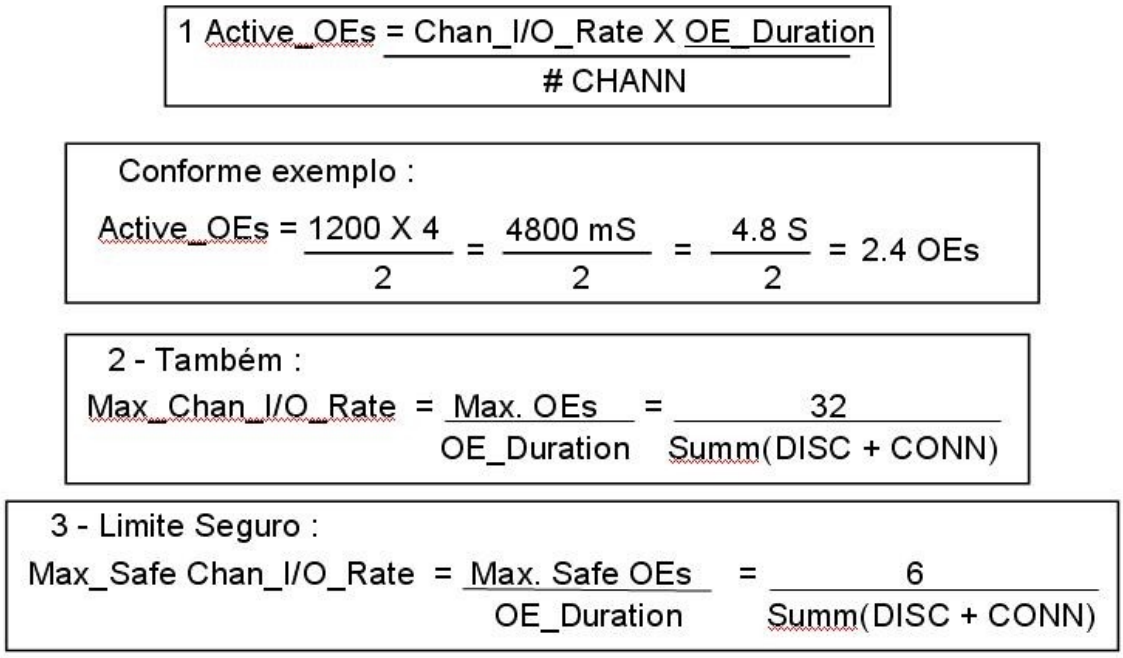


Figura 25 – Fórmulas para cálculo

Agora, para se obter o número de OEs ativos durante o período, multiplica-se o IORATE dos canais (CHANN IORATE) pela duração de cada OE (OE DURATION), e divide-se pelo número de canais da imagem.

Dois números surgem como sub-produtos destes cálculos. Um é o número máximo (teórico) de IORATE para este grupo de canais, conforme a formula 2. Neste caso, foi utilizado o número disponível de OEs nos canais FICON atuais, cujo valor deve ser corrigido, na medida em que o mesmo aumente. O segundo número, pela formula 3, é o número de IOPS que pode ser mantido por esta canalização SEM CONTENÇÃO (igual ou abaixo de 6 OEs). É a mesma fórmula, aplicando-se como limite máximo o atualmente definido pelos testes.

Test Series	% Chan Busy	% Bus Busy	Read MB/Sec	Write MB/Sec	Active OEs
4K Read-Hit	**88.9**	23.6	23.6	0.0	7.5
4K R/W: 50% Hit	15.2	8.7	2.0	2.0	31.5
4K R/W: 90% Hit	46.0	16.1	6.9	6.9	10.0
HT Read-Hit	58.4	31.5	66.4	0.0	28.5
HT R/W: 50% Hit	15.7	16.7	13.1	13.1	26.3
HT R/W: 90% Hit	39.9	27.0	24.3	24.3	29.1
HT Read Sequential	26.1	21.6	39.0	0.0	3.6
HT Write Sequential	26.7	20.3	0.0	29.8	13.5

Table 4. Channel Metrics and Estimated OEs at Saturation for PAI/O Driver Test Series

Test Series	% Chan Busy	% Bus Busy	Read MB/Sec	Write MB/Sec	Active OEs
4K Read-Hit	0.01	-0.07	-0.05	0.00	0.41
4K R/W: 50% Hit	0.74	0.74	0.65	0.85	0.99
4K R/W: 90% Hit	0.76	0.74	0.75	0.75	0.92
HT Read-Hit	0.76	0.50	0.55	0.00	0.99
HT R/W: 50% Hit	0.74	0.74	0.70	0.78	0.99
HT R/W: 90% Hit	0.84	0.59	0.65	0.64	0.99
HT Read Sequential	0.71	0.74	0.74	0.00	0.99
HT Write Sequential	0.65	0.29	0.00	0.33	0.99
All Tests	**0.08**	**0.23**	**0.10**	**0.36**	**0.85**

Table 5. Channel Metrics and Estimated OEs and Subsystem Service Time Correlation Coefficients

Figura 26 – Análise Correlacional com # OPEN OEs

Esta análise mostra que o número de OEs ativas simultaneamente apresenta um alto nível de relacionamento com o tempo de resposta (da ordem de 95%), podendo ser utilizado para avaliações e projeções.

Migração de ESCON – FICON

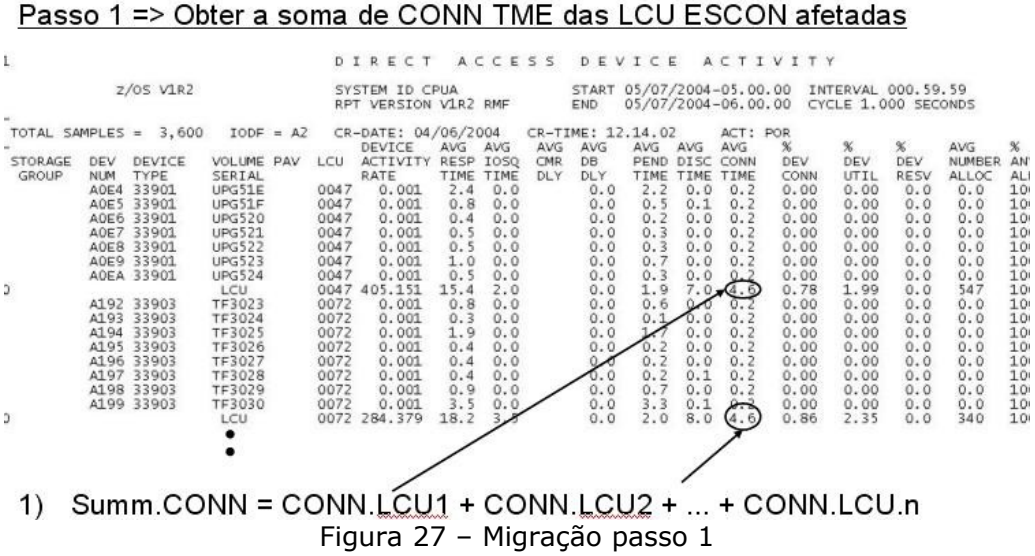


Figura 27 – Migração passo 1

Cálculos semelhantes auxiliam na conversão de um ambiente ESCON para FICON. Novamente, os cuidados tomados nas fórmulas anteriores devem ser mantidos aqui. Para se planejar a migração de canais, deve-se saber quais LCUs e

IMAGENS serão afetadas, pois todos os cálculos são baseados na somatória de suas operações. O primeiro passo é calcular a SOMATORIA dos CONN TME para todas as LCUs. No caso da conversão, o CONN é somado separadamente do DISC pois os canais ESCON medem estes tempos diferentemente.

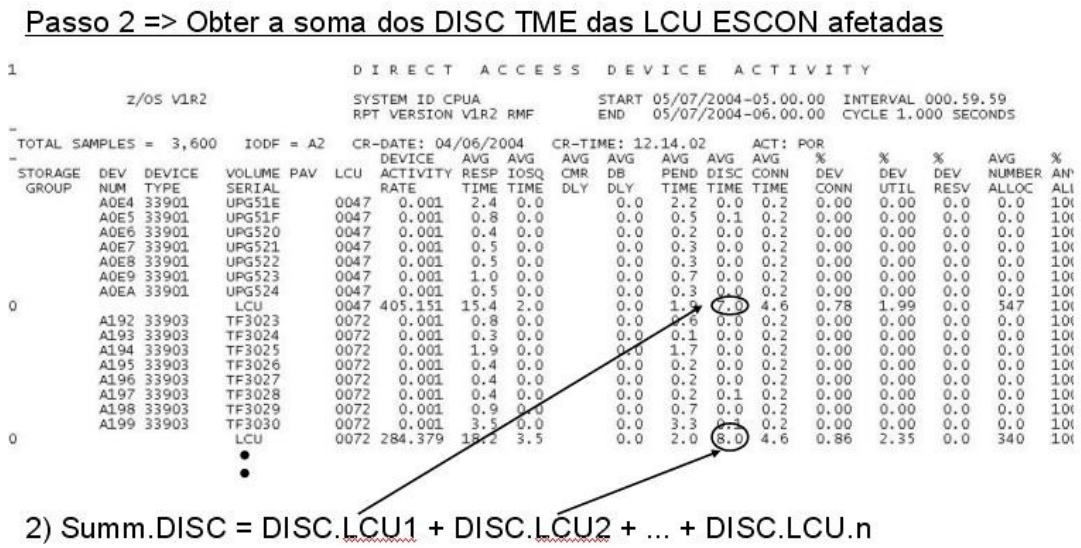


Figura 28 – Migração passo 2

O segundo passo é análogo ao primeiro, somando-se agora os DISC TME das LCUs envolvidas na migração.

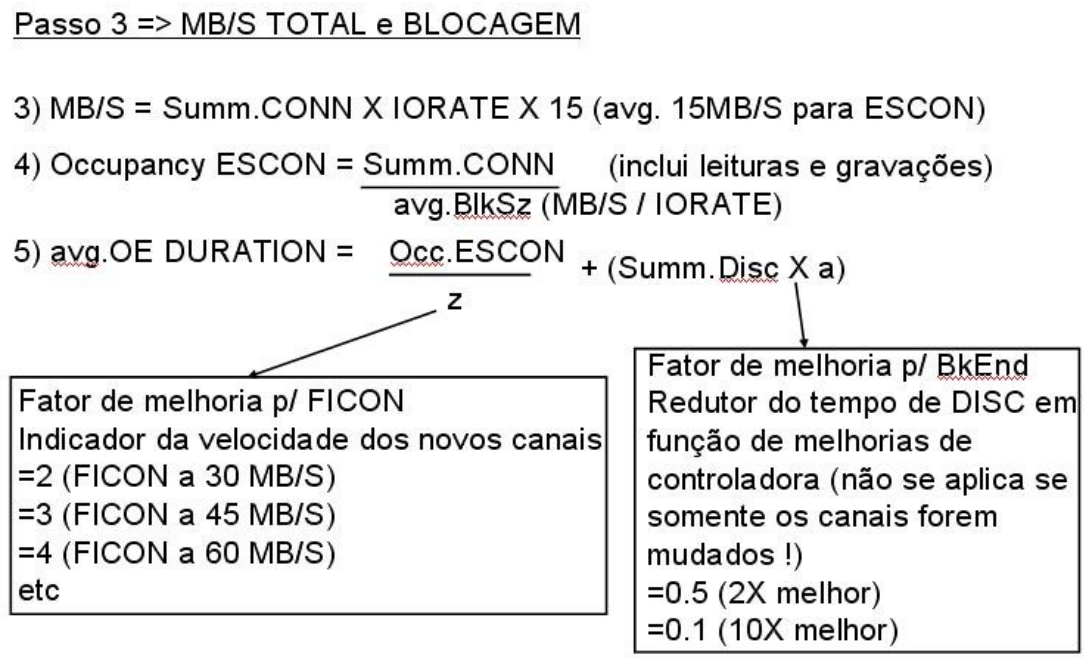


Figura 29 – Fórmulas para migração

De posse dos resultados das duas etapas anteriores, agora inicia-se a fase de cálculos. O primeiro valor a ser calculado é o THROUGHPUT ou taxa de transmissão para o ambiente em planejamento de migração. Para isto, multiplica-se a SOMA dos CONN pelo IORATE (obtido dos CHPID TAKEN do CHANNEL REPORT ou do DEVICE ACTIVITY RATE do DASD), obtendo-se assim o tempo TOTAL DE CONN para as LCUs. Isto, vezes a taxa de 15 MB/S (média de dados úteis para ambientes ESCON), obtém-se a taxa de transferência para leitura e gravação somadas (lembre-se de ajustar as unidades do Summ.CONN OU do MB/S).

Tendo este valor, podemos agora dividi-lo pelo numero de operações (IORATE) e obter o tamanho médio de blocos transferidos (Avg.Blksize). Dividindo-se o tempo de CONN apurado anteriormente pelo tamanho médio do bloco tempos o tempo médio de cada bloco, ou o tempo de OCUPAÇÃO do canal com transferência de dados, para cada bloco ESCON.

O valor médio de duração de ocupação de um OE pode ser estimado agora, somando-se um tempo de CONN projetado, dividindo-se a ocupação ESCON por uma constante que estime o aumento de velocidade fornecido pelo novo canal FICON, ao tempo de DISC (que no caso do FICON também causará ocupação do OE). O DISC pode ser multiplicado também por uma "constante de melhoria", se alguma causa de contenção de BACKEND venha a ser também abordada na migração (troca de controladora por uma mais rápida, maior dispersão de volumes, melhoria da taxa de CACHE-HIT, p.ex.). Caso contrario, o mais seguro é manter-se o valor de Summ.DISC conforme obtido dos RMFs originais.

$$\#\ CHANN = \frac{Chan\_I/O\_Rate\ X\ OE\_Duration}{Active\_OEs\ (6)}$$

Figura 30 – Cálculo do número de canais

Calculada a duração média de OEs, e com uma pequena alteração da fórmula apresentada anteriormente, temos uma estimativa para o número de canais necessários à migração (use o número máximo de 6 OEs simultâneas, a menos que tenha informações seguras dos fornecedores de que patamares maiores podem ser utilizados). O valor de Chann_I/O_Rate é o mesmo obtido dos RMFs anteriormente.

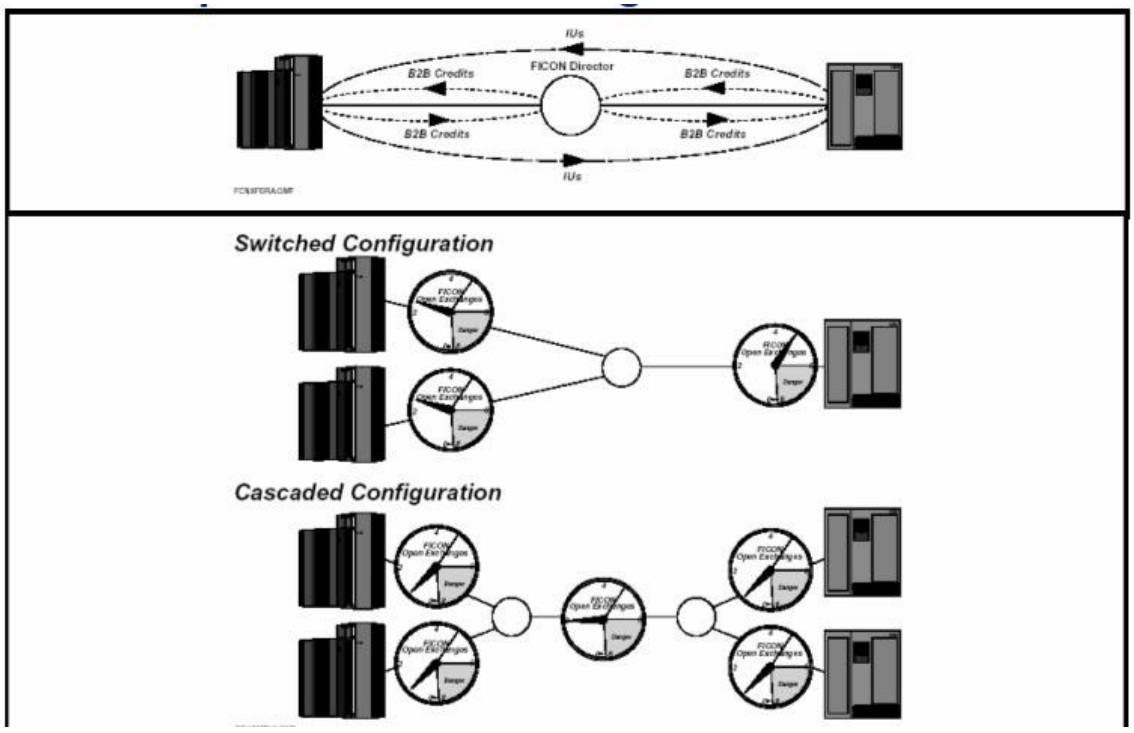


Figura 31 – Canais convergentes

O número de OEs ativos simultaneamente deve ser sempre calculado, e aplicado ao trecho mais congestionado para estabelecimento de números mínimos de canais necessários. Cálculos semelhantes podem ser efetuados quando migrando de ESCON para FICON. Caso o ambiente passe por SWITCHES, os números devem ser projetados para o trecho com o MENOR NÚMERO DE LINKs.

Há FICONs e FICONs por ai. Não são a mesma coisa. Cada fornecedor tem sua própria implementação para o protocolo, variando tamanhos de buffer, velocidade de cpu, largura de barramento, formas de enfileiramento e assim por diante. Alem disso, há as variações físicas, como short-wave, long-wave, multi-mode, single-mode, standard connector, lucent connector, etc. Conexões via PATCH PANEL e TRUNKS aumentam a complexidade.

Apesar da maioria dos fornecedores utilizar os valores nominais do FICON (100 e 200 MB/S), há grandes variações, em função da implementação e do tipo de carga aplicada a cada adaptador. As placas FICON das primeiras CPUs G5 e G6 diferem em velocidade de CPU das utilizadas nas novas. O barramento foi duplicado para o FICON Express, além do uso de LOCAL CACHE. Ou seja, nem nas CPUs IBM os FICONs são iguais. Tenha sempre em mente que as diferenças de implementação são grandes. É sempre uma sábia política conferir se o que recebemos foi realmente o que pedimos e pelo que pagamos.

III – Ambientes

1 – Configurações atuais (física, lógica e por períodos)

É comum hoje em dia encontrarmos ambientes computacionais responsáveis pelo processamento de dados de mais de uma empresa. Os antigos "*bureaus*" ou atuais "terceirizados" ("*outsourced"),* compartilham recursos físicos como CPCs, canais e unidades de armazenamento entre as cargas de empresas diferentes, procedimento do qual provem sua lucratividade.

É também bastante comum, uma única grande empresa possuir mais de uma unidade (ou instalação física) com ambientes de processamento. Um exemplo típico seria os locais para recuperação de desastres (ou continuidade de negócios, como prefere o marketing mais atual).

Estes acréscimos, mais que divisões, impõe um nível maior de complexidade à análise, pois agora pode-se ter que sumarizar dados de origem bastante distinta, como companhias diferentes, para se obter os resultados de um único equipamento para, por exemplo, estimar as características de um outro que venha a substituí-lo.

Além disto, ambientes unificados em *sysplex* também acrescentam uma complexidade adicional, pois para se estudar o perfil da "produção" (p.ex.), tem-se que reconciliar as visões que partições distintas tem do uso e desempenho de um único equipamento. Também existem funcionalidades no *hardware* de CPCs que contribuem para isso, como o *"EMIF"*, permitindo que mais de uma partição, talvez de *sysplexes* diferentes, compartilhem os mesmos canais e controladoras.

Configurações Físicas

No nível mais alto da hierarquia das configurações físicas, ao menos do ponto de vista de uma análise de desempenho ou modelagem, está o que se pode chamar de PROJETO DE ANÁLISE. Sob um único projeto pode-se analisar dados de EMPRESAS diferentes, cada qual com UNIDADES distintas, tendo em cada unidade SITES (os antigos CPDs), os quais tem instalados um ou mais CPCs e SUBSISTEMAS DE ARMAZENAMENTO. Cada CPC tem configurado em si uma ou mais PARTIÇÕES LÓGICAS, as quais utilizam (e algumas vezes compartilham) CHANNEL PATH GROUPS (grupos de até 8 canais indo atendendo partições e indo até uma única controladora ou à uma unidade de chaveamento). Por sua vez, cada SUBSISTEMA DE ARMAZENAMENTO tem suas unidades de disco agrupadas em IMAGENS de 2048 ou 4096 unidades lógicas (que são na realidade subdivisões do espaço em discos físicos reais; veja DISCOS RIGIDOS acima). As imagens também se subdividem em unidades menores chamadas de *LOGICAL CONTROL UNITs (LCUs)*, de 256 dispositivos em cada uma.

Com uma configuração dessas, tentar entender o porquê de, por exemplo, o *on-line* do *help-desk* está com um tempo de resposta ruim, ou o *batch* de faturamento está demorando torná-se uma tarefa no mínimo hercúlea.

A seqüência seguinte apresenta uma proposta bem-sucedida em inúmeras análises práticas, de nomear e hierarquizar os diferentes elementos envolvidos nestas configurações :

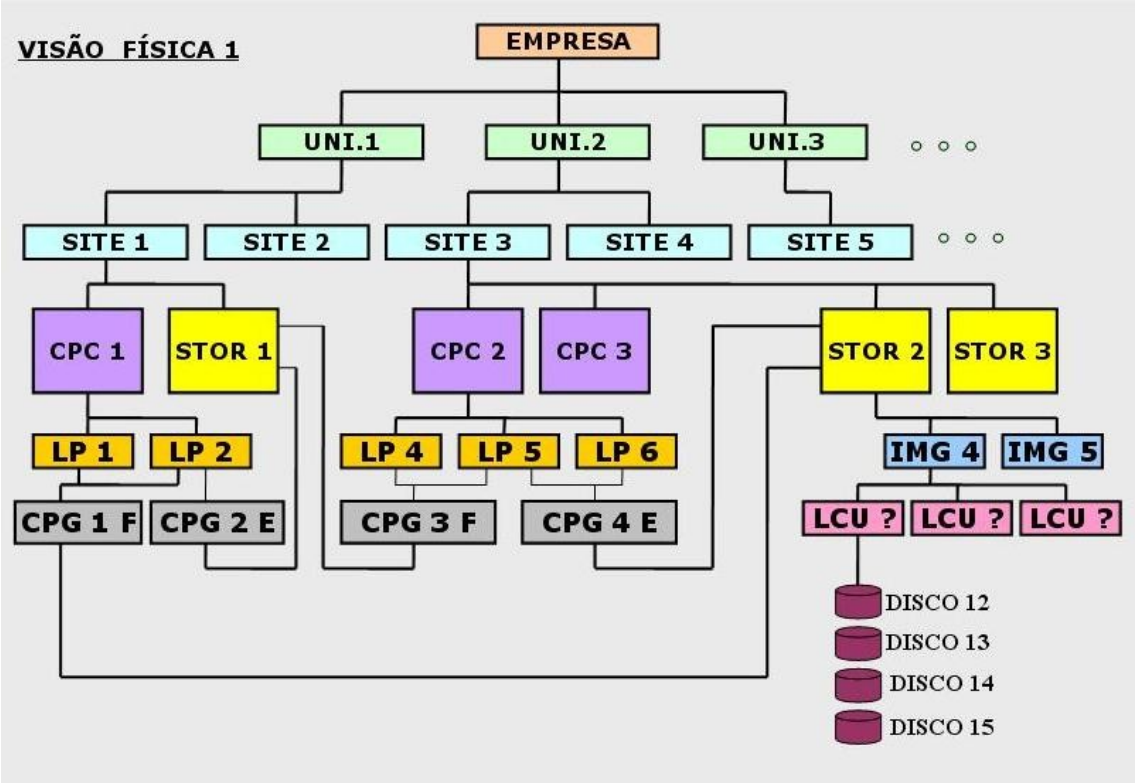


Figura 32 : Visão física

Portanto, iniciando-se ainda acima do mostrado na figura, temos :

1. Projeto de análise
2. Empresa
3. Unidade
4. Site
   - 4.1 CPCs
     - 4.1.1 – LPARs
     - 4.1.2 - CPGs
   - 4.2 STORs
     - 4.2.1 – Imagens
     - 4.2.2 – LCUs
     - 4.2.3 – Unidades (de disco, em nosso caso)

Configurações Lógicas

Independentemente de qual CPC uma LPAR pertença, ela pode ser (lógica ou funcionalmente) agrupada para operar como um único sistema, a outras partições de CPCs ou Sites ou Unidades distintas, como um SYSPLEX. Uma vez que o identificador de LCU que uma partição utiliza para um grupo de discos é dependente do CPC no qual ela executa (ou, mais especificamente, da geração feita no CPC), é possível e bastante freqüente que um grupo de partições, executando atividades comuns sob um único SYSPLEX, "chamem" os grupos de discos sob nomes diferentes. Esta é uma das complexidades que devem ser superadas ao se

analisar relatórios de desempenho de ambientes distintos. A figura abaixo ilustra o texto :

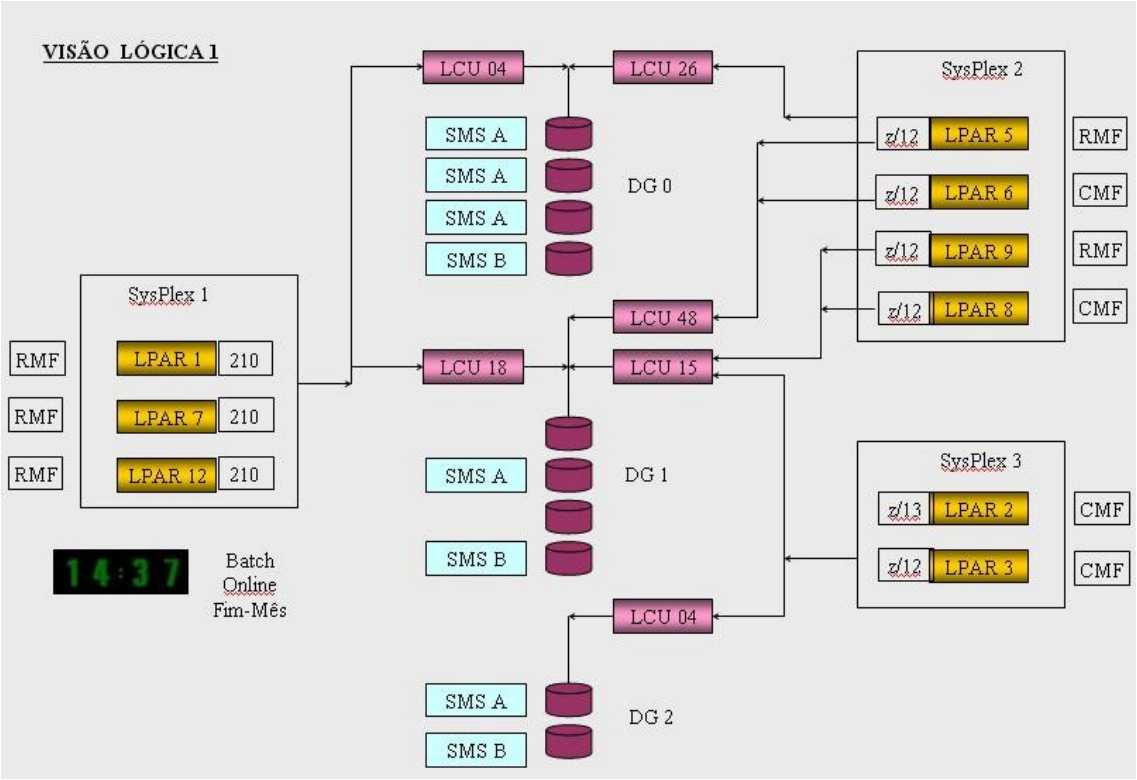


Figura 33 : Visão lógica

Além do agrupamento de LPARs em SYSPLEXEs, temos também a divisão operacional, na qual um ou mais SYSPLEXES respondem pelo ambiente de "PRODUÇÃO", outro(s), pelo "DESENVOLVIMENTO", outros, "HOMOLOGAÇÃO", e assim por diante.

No final da hierarquia das configurações físicas, as unidades lógicas de disco ainda podem ser grupadas lógica ou funcionalmente, sob definições como do IBM DFSMS ou CA SAMS-Vantage, de forma que algumas unidades respondem pelo grupo ("*pool*") de batch, outras pelo DB2 do DW, etc.

Todos estes agrupamentos e divisões têm de se levados em conta quando da avaliação de desempenho de um ou mais equipamentos, afim de se obter resultados na mesma linguagem utilizada pela instalação.

Configurações de períodos

Um último fator a ser considerado aqui é o TEMPO.

O perfil dos dados processados em instalações modifica-se conforme o passar das horas, acompanhando, de certa forma, a dinâmica das pessoas que os criam. Assim, durante o dia, tem-se o perfil on-line, enquanto as pessoas estão nas ruas, acessando contas, fazendo compras e viajando. Mais à noite, a atividade interativa tende a ser menor, período este do qual as instalações tradicionalmente se valem para a execução de suas atividades de manutenção. *Backups*, testes de novos produtos ou correções, "*clean-ups"* ou cargas de bancos de dados, etc.

Uma vez que o perfil de utilização muda, também mudam os números de desempenho, e a forma de analisá-los. Um bom exemplo disso é o tamanho médio de bloco, geralmente menor durante o on-line (4 a 16 KB), e maior para o batch (32 a 64kb). Uma taxa de operações de IO (IORATE) tem um significado bastante diferente, dependendo do tamanho médio utilizado.

2 – Anatomia de um I/O MF

Sem dúvida, a grande maioria das operações de entrada e saída de um sistema tradicional é gerada pelas transações de negócio, sejam operando sob a forma de JOBs batch, interações on-line ou movimentação de dados de/para arquivos auxiliares, como as do SPOOL e paginação. Não considerando operações de bancos de dados, as outras são em menor número e de menor impacto, como JOURNALING para o CICS.

Os modernos ambientes z/OS apresentam um misto das atividades tradicionais, tipicamente voltadas ao manuseio de arquivos seqüenciais ou particionados, operações destinadas à otimização de desempenho (carga da LLA, manuseio de VSAM BUFFERS e STAGE de arquivos de dados de BATCH ou PROCEDURES via VLF), e intensas atividades de bancos de dados (gravação de LOGFILES, PRE-FETCH de dados para seus buffers locais e posterior DESTAGE).

Neste tópico são abordadas algumas das características principais destas operações, de forma que seus impactos sobre o desempenho dos subsistemas de armazenamento, possam ser entendidos e avaliados corretamente.

Parte 1 – Seqüenciais tradicionais

Ainda hoje, há uma parcela considerável do total de operações de entrada e saída de um sistema (ou complexo de sistemas), voltada ao manuseio de arquivos seqüências tradicionais (DSORG=OS). Estes arquivos são utilizados através de programas especializados em seu manuseio, chamados de METODOS DE ACESSO (em inglês, *Access METHODS* ou AMs). Abaixo é descrito o fluxo de uma operação, considerando-se um JOB BATCH e programação ASSEMBLER.

Antes que qualquer operação seja possível, o programa, ao ser submetido, especifica através de "cartões" DD (DATA DEFINITION), as especificações dos arquivos que irá utilizar. A interpretação destes DDs cria blocos de controle (FCBs, etc), que serão mais tarde carregados para áreas específicas no ADDRESS SPACE de execução, pelo INITIATOR. Durante a carga, blocos como a TIOT (TASK INPUT/OUTPUT TABLE), e a DCB (DATASET CONTROL BLOCK), são criados, para o mapeamento dos arquivos necessários. A DCB é parcialmente preenchida em tempo de carga, contendo campos em branco, os quais serão preenchidos por macros posteriores.

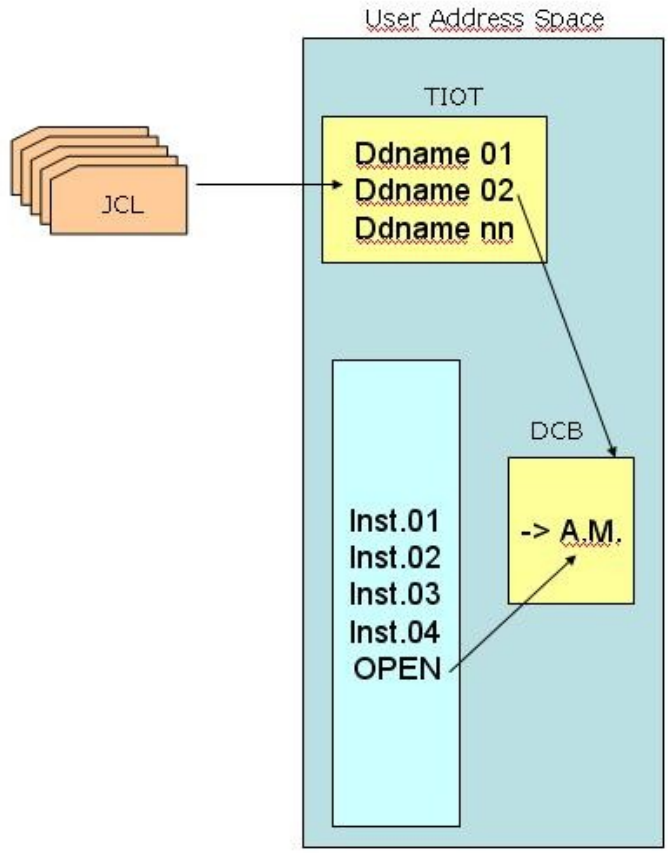


Figura 34 - OPEN

Uma vez recebendo o controle, e desejando iniciar uma operação de I/O, o programa emite uma macro OPEN, que faz com que informações adicionais sobre o arquivo em questão sejam obtidas. Neste momento, pode-se iniciar uma pesquisa em catálogos do sistema, leituras de diretório, e preenche-se um campo na DCB com o endereço do AM correspondente a organização de dados do arquivo. Até esse momento, nenhuma operação de transferência de dados ocorreu contra o próprio.

Para que a operação seja efetuada, o programa emite uma macro de transferência de dados (READ/WRITE, GET/PUT), dependendo do estilo de serialização desejado. As primeiras (READ/WRITE) causam a passagem de controle ao BSAM (BASIC SEQUENTIAL Access METHOD), e as últimas ao QSAM (QUEUED SEQUENTIAL ACCESS METHOD). A diferença entre eles sendo basicamente o tipo de retorno que efetuam ao programa. O método BASIC retorna o controle ao programa ANTES da operação ter sido completada, deixando ao mesmo a responsabilidade pela serialização ou verificação de termino da mesma. Já o método QUEUED coloca o programa em estado de espera (WAIT), somente voltando a ativá-lo quando a operação terminou.

Assumindo-se um GET, o programa especifica, como parâmetro, uma área de memória para receber o registro desejado. O desdobramento da macro causa um BRANCH (desvio), para o código do método, geralmente localizado na LPA, que passa a executar daí em diante.

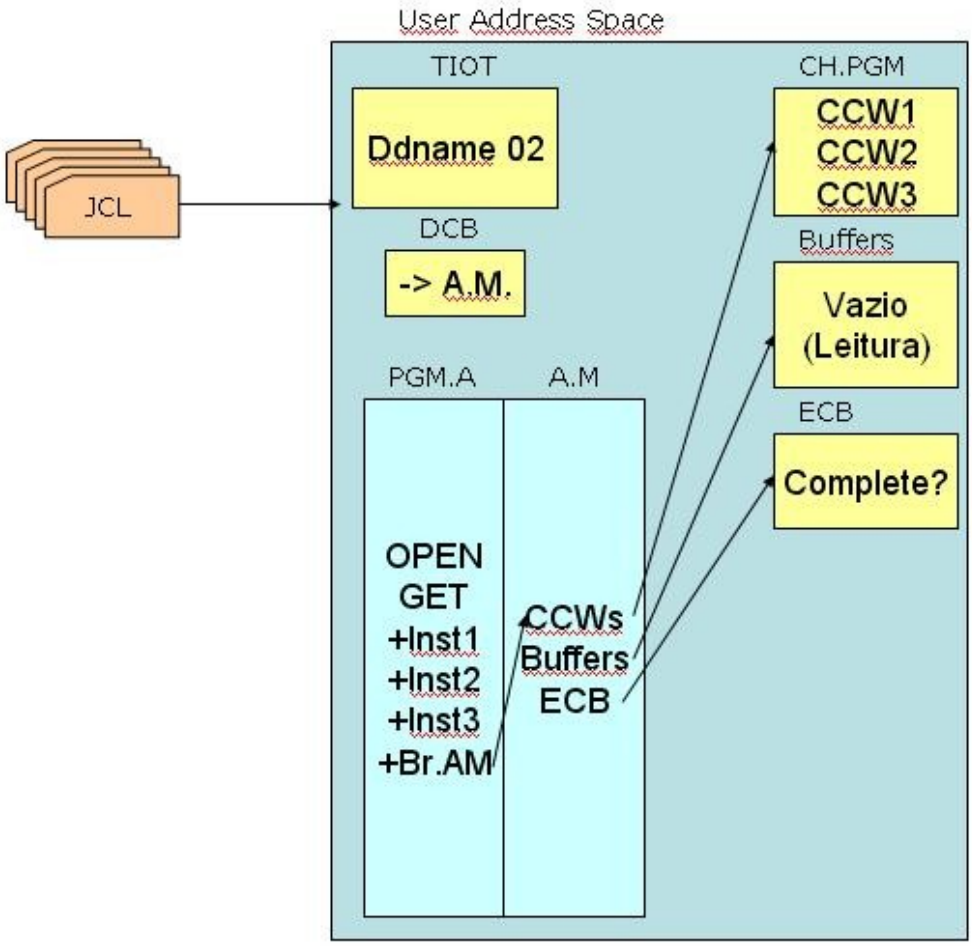


Figura 35 - GET

Levando em consideração o tipo de pedido e as características do arquivo em uso, o AM constrói áreas de controle importantes para o manuseio da operação. Uma destas áreas é a cadeia de CCWs (CHANNEL COMMAND WORDs), ou programação de canal, que será enviada ao subsistema de armazenamento, afim de executar a operação desejada. Outras áreas importantes são os BUFFERs de dados, cujos tamanhos e quantidades podem ser parametrizados manualmente, ou escolhidos pelo método. É para esta área que o canal enviará os dados ao final de sua operação (e NÃO para a área especificada pelo programa; esta função é do método). No caso do QSAM, um outro bloco importante construído neste momento é a ECB (EVENT CONTROL BLOCK), que servirá para serialização desta operação.

Concluídas as atividades de preparação, o AM emite a macro EXCP (EXECUTE CHANNEL PROGRAM), a qual causa uma interrupção no processador (SVC INTERRUPT) de código 0. Esta interrupção, após ter sido tratada pelo processador de interrupção, faz com que o controle seja passado para um componente de sistema chamado DRIVER. Este é o componente responsável por adaptar a operação pedida pelo programa (e preparada pelo método de acesso), a um formato compreendido por outro componente do sistema, chamado IOS (INPUT OUTPUT SUPERVISOR). O IOS é o único que interage com o CSS (CHANNEL SUBSYSTEM), e somente compreende o pedido de operação quando o mesmo lhe é passado sob um certo formato, incluindo blocos de controle e ponteiros pré-determinados. Esta adaptação é feita para todas as operações de IO (portanto, para o JES, ASM em paginação e outros), sendo que cada requisitante utiliza seu próprio "tradutor". Este modelo facilita bastante a programação necessária a uma

operação de IO, pois cada usuário pode formatá-la na forma que lhe for mais conveniente, sendo todas, posteriormente, traduzidas por seu DRIVER.

O EXCP DRIVER, como o método de acesso, também cria as áreas ou blocos de controle necessários à padronização da chamada para o IOS. Entre esses, a IOSB e a IOQ são as mais importantes. A IOQ é o bloco compreendido pelo IOS como a representante de uma operação de IO. A mesma aponta para a IOSB que lhe é correspondente, a qual aponta para a ASCB (ADDRESS SPACE CONTROL BLOCK) do chamador original, e a ECB (EVENT CONTROL BLOCK) a ser utilizada para a serialização, entre outras coisas. A IOSB apresenta uma extensão (IOSBE), que contem informações sobre como uma operação deve ser executada quando utilizando canais FICON (ver capítulo sobre canais). O ponteiro para o programa de canal a ser utilizado também fica na IOSB.

Outra operação importante efetuada aqui é a fixação das páginas correspondentes ao IO BUFFER (área para a qual os dados do GET devem ser enviados pelo canal), e da CCW CHAIN (programa de canal), na memória real. O subsistema de canal não opera com endereços virtuais, por isso todas as páginas a serem utilizadas pelo mesmo devem ser fixadas nos frames de memória real que residiam quando da construção da cadeia de ponteiros, ou seja, não podem sofrer paginação.

Terminadas estas atividades, o EXCP DRIVER emitirá a macro STARTIO, que causa a entrada no código do IOS, apontando via parâmetros todas as áreas criadas anteriormente.

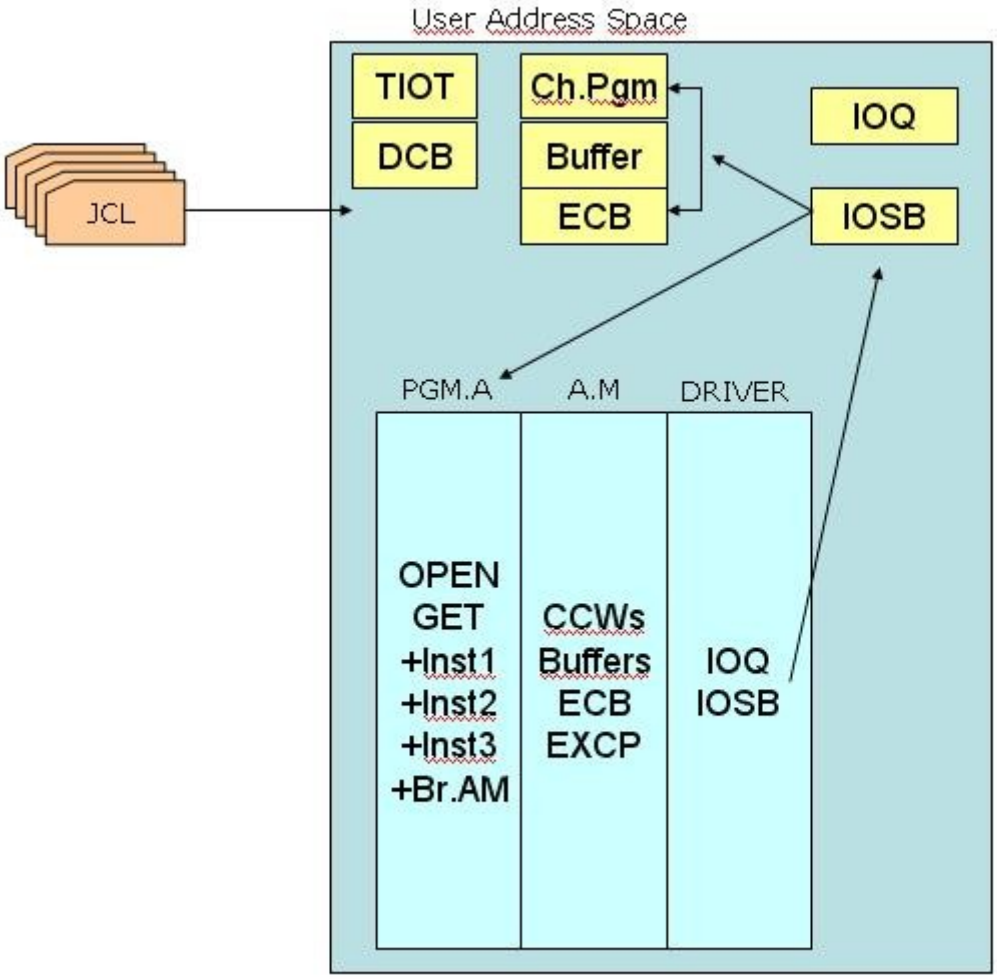


Figura 36 - EXCP

O IOS divide-se em duas grandes partes, chamadas de FRONT-END e BACK-END. A primeira é responsável por iniciar as operações de IO. A segunda por receber controle quando a operação termina, analisar os resultados e avisar as partes de código interessadas. Portanto é o IOS FRONT-END quem ganha controle como resultado da STARTIO, recebendo como parâmetros principais os apontadores da IOQ e IOSB que representam este pedido de operação.

Seu trabalho a partir deste ponto consiste em verificar se o dispositivo desejado (endereço de disco) está disponível, ou já encontrá-se em uso por outra operação (seja do mesmo programa ou de outro). Para isto, verifica a existência ou não de uma outra IOQ (operação de IO) já enfileirada a partir do bloco que representa um dispositivo (UCB – UNIT CONTROL BLOCK). Caso haja, a IOQ recebida agora é enfileirada atrás da última IOQ presente, a menos que o controle de WLM IO PRIORITY esteja ativo para este sistema, caso no qual a IOQ será posta em fila na mesma ordem hierárquica que corresponda à prioridade original do ADDRESS SPACE que lhe deu origem (o tempo gasto nesta fila é calculado pelo RMF e apresentado na coluna IOSQ do DASD ACTIVITY REPORT).

Caso nenhuma IOQ esteja presente na UCB, o IOS coloca a que recebeu como primeira da fila e marca a UCB como ATIVA para esta operação. Feito isso, copia campos específicos da IOQ e IOSB para um bloco de controle de hardware, chamado ORB (OPERATION REQUEST BLOCK), apontando o mesmo pelo GPR1, e emite a instrução SSCH (START SUBCHANNEL) para o dispositivo desejado. Neste

ponto a execução de instruções é suspensa no CP no qual o IOS estava executando, aguardando a resposta dos processadores de canal. Estes validam a operação (checando, por exemplo, a existência de uma UCW – UNIT CONTROL WORD, o equivalente à UCB para o CSS, e se o mesmo encontrá-se online), e devolve um CONDITION CODE via bits de PSW indicando que a operação pode prosseguir. Deste ponto em diante, o CONDITION CODE é repassado de volta até o método de acesso, o qual entra em WAIT (SVC01), aguardando que a ECB que criara originalmente seja preenchida com o resultado da operação. Durante todo este tempo, o programa original (emissor do GET) esteve parado, e continuará assim até que o método lhe devolva o controle, o que só acontecerá ao fim do IO. Neste meio tempo, os CPs do sistema irão executar códigos de outros programas (este estado é informado pelo RMF Monitor III sob a forma de IO-WAIT).

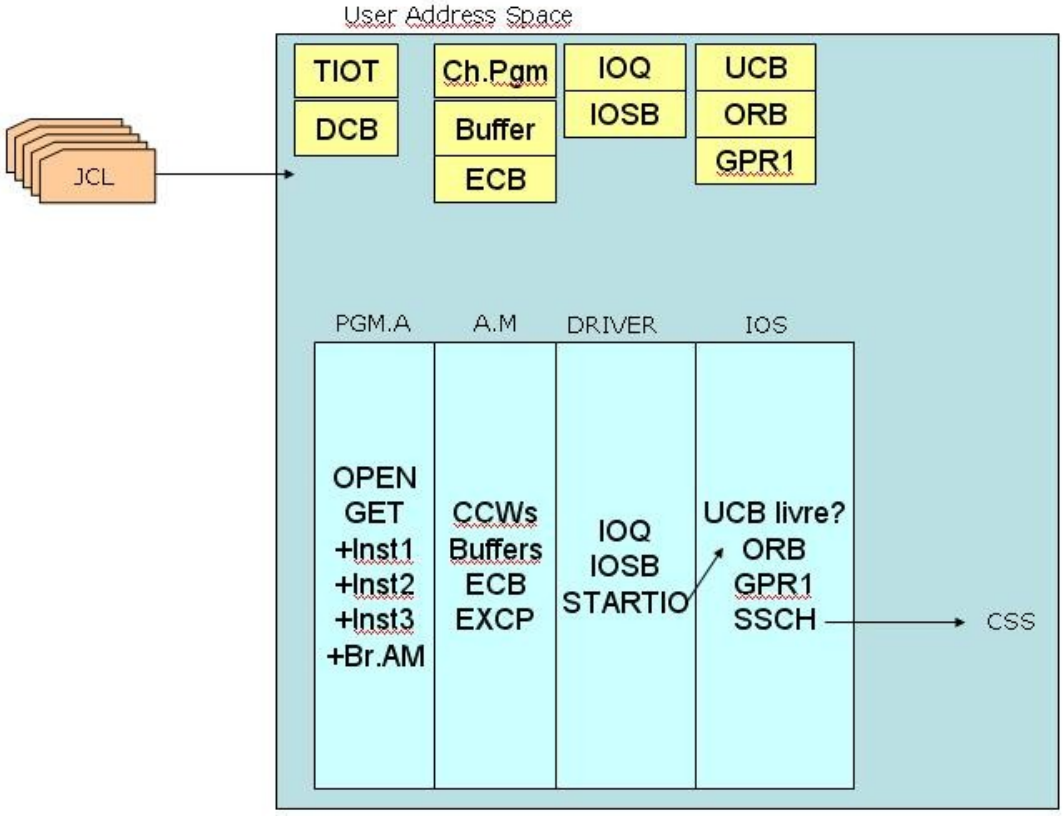


Figura 37 - IOS

Caso o dispositivo esteja indisponível no momento, por ter um ou mais de seus recursos de acesso ocupados por outra operação, este pedido é enfileirado em uma *Initiative Queue* (enfileiramento na unidade lógica que representa o conjunto de caminhos para um dado dispositivo), e aguarda até que o mesmo seja liberado. Neste ponto são gerados os tempos de CHANNEL PATH BUSY, DP BUSY e CU BUSY, dependendo de qual recurso estava ocupado.

Em se liberando este recurso, o pedido de operação é enviado através do caminho de acesso (canal, port de switch, control unit), requisitando o início da operação. Neste ponto, o dispositivo físico em si ainda pode estar ocupado, executando uma operação iniciada previamente por um outro sistema que compartilhe de arquivos no mesmo. Esta situação é informada pela controladora, e o sistema zOS o reporta sob a forma de DEVICE BUSY.

Quando finalmente a operação é iniciada, os dados são transferidos entre o dispositivo e a memória física do sistema e, o tempo gasto nesta transferência informado como CONNECT TME. Às vezes, durante a transferência (CCW CHAIN), há a necessidade de um reposicionamento físico das cabeças de leitura/gravação, quando então há uma desconexão com o sistema, e uma reconexão após a nova posição ter sido atingida. Este tempo é informado como DISCONNECT TME, consideravelmente reduzido caso os dados sejam localizados no CACHE da controladora (ver 4-Estratégias de CACHE).

Todo o transcorrer destas operações são controladas pelos canais e SAPs. Ao término das mesmas (final da CCW CHAIN ou erro), uma interrupção de I/O é gerada, colocando-se as informações do status da transferência em um outro bloco de controle (Interrupt Request Block – IRB), o qual será analisado por códigos do IOS-BACKEND, (IOSVIRBA, por exemplo). Caso as operações tenham transcorrido sem problemas, o método de acesso (AM) será avisado via macro POST, retomando o controle. Em alguns casos, haverá ainda processamentos adicionais, como a divisão dos dados recebidos em registros lógicos, e a transferência do buffer de I/O para a área de dados do cliente. Após isso, o código do programa que pediu os dados retomará controle.

Parte 2 – Bancos de Dados

Grande parte dos serviços críticos prestados atualmente pelos ambientes computacionais à seus usuários finais, traduz-se na forma de operações de entrada e saída, executadas não mais contra arquivos discretos, como descrito acima, mas por um único programa, um Gerenciador de Banco de Dados, relacional ou não, contra suas bases de dados.

Uma vez que o desempenho apresentado ao usuário final é um dos grandes indicadores utilizados na avaliação de uma instalação (um outro seria o cumprimento dos prazos de seu processamento batch), este resultado é de grande criticidade para a mesma.

Entretanto, estes ambientes apresentam algumas diferenças em seu processamento de I/O, as quais, embora sutis, são de grande impacto tanto na análise de seu comportamento, quanto em sua gerência.

Em resumo, poderíamos dizer que as principais diferenças, no que diz respeito à análise, seriam:

- Mais de um I/O por transação
- Uso de índices
- Processamento adicional
- Cache pré-fetch próprio

Vamos revisá-las uma a uma, e entender seus impactos em nossas avaliações.

Mais de um I/O por transação

Um cliente requisita seu extrato de conta na ATM ou Internet. Em resposta, um ambiente transacional dispara uma ou mais transações, algumas das quais voltadas aos Bancos de Dados. Estas transações de DB´s convertem-se em (geralmente), alguns acessos aos arquivos do banco. Algum tempo depois, os

diversos resultados são obtidos, sumarizados, formatados e enviados como resposta. E somente neste ponto termina a espera do usuário final.

O ponto importante a ser ressaltado aqui é que o tempo "sentido" pelo usuário final pode ser "longo", mesmo que cada uma as operações de I/O requisitadas ocorram em prazos bastante aceitáveis, por ser a soma de todas elas, acrescida de processamentos para sumarização e envio, por exemplo.

Uso de Índices

Independentemente de sua estrutura, os bancos de dados utilizam índices ou catálogos, centrais aos produtos, ou inerentes ao desenho de um banco em particular. Estes arquivos, caso apresentem problemas de desempenho, podem influenciar negativamente o resultado de todo um banco que, por si só, estaria desempenhando muito bem. Aqui cabe bem o ditado de "um olho no peixe e outro no gato".

Processamento adicional

Gerenciadores de Bancos de Dados são programas complexos, e apresentam uma infinidade de recursos valiosos para o desenvolvimento das aplicações de que precisamos hoje em dia. Entretanto, é sempre importante manter-se em mente que todos estes recursos (JOINs, VIEWs, STORED QUERIES, para mencionar algumas), implicam em processamentos cujo tempo somá-se ao do acesso normal aos discos. Uma pesquisa (QUERY) mal-escrita é, quase sempre, a maior responsável por problemas de desempenho de um DB, apesar de, na maior parte dos casos, os discos serem os primeiros a sofrer a responsabilidade.

CACHE PRE-FETCH próprio

Via de regra, quando algo vai mal no ambiente transacional ou batch, e que utiliza DBMs, os discos são os primeiros a ser incriminados. Neste momento, devido à criticidade destes ambientes, as empresas responsáveis pelos subsistemas são convocadas para avaliar o que pode estar errado por ali. Utilizando ferramentas próprias ou do sistema (como o RMF, por exemplo), as empresas mostram que seus discos estão com um excelente desempenho, "inclusive, resolvendo a grande maioria das operações EM CACHE", o que, obviamente, as torna bastante rápidas.

Entretanto, quando se trata de operações contra DB´s, estes números são absolutamente inúteis, e a razão é simples: todo DBM mantém uma versão própria de CACHE em memória, para o qual TAMBEM EFETUA PRE-FETCH, exatamente como as controladoras. Operações de PRE-FETCH caracterizam-se como seqüenciais. Daí, logicamente, a MAIORIA das operações, conforme experimentadas pelas controladoras, são resolvidas, rapidamente, em CACHE. Entretanto, essas operações NÃO SÃO as que resolveram a transação do cliente, mas sim as que serviram para carregar o CACHE do DBM e, portanto, para o usuário final, não fizeram qualquer diferença.

A regra referente a este ponto é também bastante simples: ao se analisar problemas de desempenho em volumes que contenham arquivos de bancos de dados, NÃO SE DEVE UTILIZAR as ferramentas das controladoras ou do sistema, MAS SIM os medidores do próprio DBM em relação a CACHE HITS atingidos pelas transações. Na grande maioria das vezes, aquele único I/O, lento e feito no próprio disco, e que disparou o pré-fetch em ambos os ambientes (controladora e DBM), era o único que interessava ao cliente. E daí, se for o caso, corrigir o desempenho desta única operação importante.

É sempre importante lembrar que os DBMs servem, geralmente, a dois ambientes, muito distintos quanto às características de operação. Um, o transacional, caracterizá-se por acesso aleatório a registros pequenos, e que exige mais da arquitetura de cada equipamento (pontos de estrangulamento), e outro, o BATCH (para backup, geração de relatórios, etc), que o processa seqüencialmente, beneficiando-se mais dos CACHE HITS criados por este perfil, tanto no LOCAL CACHE do DBM, quanto nos das controladoras envolvidas.

3 – Novos recursos

Ao longo dos últimos anos, várias melhorias tecnológicas foram acrescentadas ao (já complexo) ambiente de armazenamento. Algumas das mais importantes foram o Parallel Access Volume / Multiple Allegiance (PAV/MA), Channel Subsystem IO Priority Queuing e o Dynamic Channel-path Management (DCM), sendo os dois últimos introduzidos com o conjunto de funcionalidades chamado de IBM/IRD (Intelligent Resource Director), da arquitetura Z.

Parallel Access Volume (PAV) e Multiple Allegiance (MA)

PAV e MA são duas funcionalidades distintas, algumas vezes tomadas por uma única, por serem mutuamente dependentes, e terem sido introduzidas simultaneamente no mercado. PAV implementa funcionalidades no sistema operacional. MA opera no microcódigo da controladora somente. O fato de resolverem problemas semelhantes também contribui para a confusão.

Conforme pode ser visto no item 2 acima (Anatomia de um I/O), o sistema operacional interage com uma unidade de disco (assim como com qualquer outro dispositivo de entrada e saída), através de sua representação em memória, chamada de UCB (Unit Control Block). As UCBs são criadas através do processo de geração de configuração do sistema (Hardware Configuration Definition - HCD), e carregadas em memória em tempo de IPL (Initial Program Load).

O sistema foi desenhado dessa forma porque da época de sua concepção até recentemente (em termos de Mainframes), uma unidade de disco era um dispositivo físico. Um braço atuador só pode se movimentar em uma única direção, para atender a uma única operação de cada vez. Portanto, a representação era satisfatória.

Entretanto, a introdução de subsistemas de armazenamento com tecnologias como RAID e CACHE alteraram isso, pois permitiam que mais de uma operação fosse executada, simultaneamente, contra um único dispositivo. Poderia haver várias trilhas, já pré-carregadas no cache, que poderiam ser lidas simultaneamente por várias operações. Assim como os dados, dispersos fisicamente por mais de um disco (RAID – ver acima), também permitiriam, mesmo que os dados requeridos não estivessem em memória.

Portanto, posteriormente à introdução de tais controladoras, o sistema operacional foi modificado, de forma a tirar vantagem do paralelismo de operações, agora oferecido pelas mesmas.

A alteração em si baseia-se na criação de UCBs adicionais para um mesmo dispositivo lógico (DEVNUM ou VOLSER), de forma que mais de uma operação possa ser iniciada contra o mesmo. A UCB que representa o dispositivo original passa a chamar-se UCB BASE, e suas replicas (cada uma com um DEVNUM diferente, mas apontando para o mesmo VOLSER), passam a se chamar UCB ALIAS, e são utilizadas somente internamente, pelas rotinas do IOS.

Aliada a essa mudança do sistema, as controladoras tiveram de implementar uma tabela de controle, chamada de ALIAS TABLE, para identificar QUAIS ALIAS apontam para QUAIS BASEs. Uma vez dispondo desse suporte, o representante técnico ou o próprio cliente (dependendo do fornecedor do subsistema), faz a identificação de quem aponta a quem (cria a ALIAS TABLE), a qual é carregada durante a inicialização do subsistema. Posteriormente, esta tabela pode ser modificada de forma manual (Static PAV), ou dinamicamente, por sugestão do WLM (Dynamic PAV). A restrição aqui é que caso uma controladora com suporte a PAV atenda a mais de um sistema, dinamicamente, é necessário que estes sistemas (imagens), façam todos parte do mesmo WLMPLEX e tenham WLM DYNAMIC=YES ativado para todos. Não é permitido o compartilhamento de unidades Dyn-PAV entre sistemas sem conexão, ou conhecimento da possibilidade de mudanças na ALIAS TABLE.

Uma vez implementada, a funcionalidade permite que, ao se tentar iniciar uma operação de IO, caso se verificar que a UCB já esta em uso pelo sistema (há uma IOQ enfileirada na mesma, e a indicação de BUSY esta ativa), a operação é re-direcionada a uma das UCBs ALIAS, e iniciada, junto ao CSS (Channel SubSystem), como se o dispositivo estivesse desocupado. As restrições referentes à segurança de dados, como as causadas pela sobreposição de operações de gravação sobre áreas atualmente em leitura, são impostas pela controladora, através do enfileiramento interno de operações concorrentes. Cada fabricante implementa um comprimento de fila (QUEUE-DEPTH) diferente para estes enfileiramentos, os quais, quando excedidos, ainda apresentam DEVICE BUSY ao sistema. Entretanto, essa ocorrência é bastante rara, pois a ordem de grandeza varia de várias dezenas a centenas de operações enfileiradas.

Já a MA (Multiple Allegiance), surge como uma decorrência natural do PAV. Nas implementações antigas, quando uma operação de IO era iniciada contra um dispositivo lógico, a controladora assegurava uma fidelidade única ("single allegiance"), somente à imagem que possuía o CPG (Channel Path Group) através do qual a operação foi recebida, barrando outros sistemas de acessá-lo (DEVICE BUSY). Agora, como é possível receber-se diversos pedidos de operação contra o mesmo dispositivo, nada mais lógico que assumir uma fidelidade múltipla ("multiple allegiance"), recebendo-os a todos.

Atualmente, para a maior parte dos casos que resultariam em DEVICE BUSY, as controladoras aceitam e enfileiram a operação, reportando o tempo gasto nesta fila como DISCONNECT, ao final das mesmas. É o caso, por exemplo, de operações que apresentem conflitos (gravações no mesmo EXTENT sendo atualmente lido ou gravado). Já para situações de RESERVE, a resposta continua sendo DEVICE BUSY.

Channel Subsystem IO Priority Queuing

Prioridades só fazem diferença caso haja enfileiramento. Para esses casos, os sistemas Mainframe já dispõem de algoritmos de prioridade há muito tempo. Uma dada operação de IO já podia ser enfileirada na UCB de acordo com sua ordem de chegada (FIFO), de acordo com a prioridade de DISPATCH da unidade de

trabalho original (IOQ=PRIORITY da IEAIPSxx), ou de acordo com uma prioridade manualmente especificada no PGN de sua task (IOP=xx).

Com o advento do zOS 1.3, os sistemas executando em WLM GOAL MODE passaram a dispor de um modo adicional, que permitia enfileirar as IOQs (pedidos de IO), de acordo com a importância e resultados (PI) de suas SERVICE CLASSES. Ao mesmo tempo (zOS 1.3), os subsistemas de armazenamento passaram a considerar o byte de prioridade da CCW de DEFINE EXTENT (definido nesta versão), enfileirando os pedidos de operação de acordo com ele.

A única parte do caminho seguido por uma operação de IO que ainda não era capaz de respeitar uma ordem de prioridade (operava FIFO), era o próprio Channel Subsystem (CSS). Nele, há uma fila de SSCH (Start Subchannel), para cada SAP (System Assist Processor), na qual os pedidos eram enfileirados e processados de acordo com sua ordem de chegada. Esta também era a ordem mantida nas filas de recursos (CPGs, control units, etc).

Claro que, do ponto de vista de cada LPAR, não haveria problema, pois a prioridade da UCB definiria a ordem de chegada das operações no CSS. Entretanto, o problema surge quando diversas LPARs compartilham do mesmo CSS, como é o caso na maioria dos ambientes atuais. Da forma como era feita anteriormente, uma operação de baixa prioridade, vinda de uma imagem, poderia "plantar-se" à frente de uma outra, de alta prioridade, porem originada em outra partição do mesmo CPC.

Este desvio é corrigido a partir da geração z900, que permite que as mesmas informações de prioridade utilizadas para enfileiramento na UCB de cada imagem, e transmitidas às controladoras via DEFINE EXTENT, sejam também utilizadas quando do enfileiramento no CSS, não importando a imagem de origem.

Comentário adicional

Quanto aos esquemas de prioridade, há um comentário adicional que vale a pena ser feito, e refere-se à unidade de disco (físico) em si.

Todos os subsistemas de armazenamento modernos são equipados com unidades de disco que, já há muito, deixaram de ser simples dispositivos eletromecânicos. Estes contam, hoje em dia, com processadores de interface, buffers locais (ALBs), e implementações de lógicas de controle, entre outras sofisticações.

Um dos algoritmos que estas unidades implementam é conhecido, popularmente, como "algoritmo do elevador", e recebe seu nome do fato de ser diretamente derivado do uso racional de elevadores.

Em poucas palavras, quando um grupo de pessoas entra em um elevador, e cada um pressiona seu andar de destino, este irá parar, seqüencialmente, em cada um deles, começando pelo mais próximo. Não há muitos casos reportados de elevadores que distinguem a importância de seus ocupantes e, por exemplo, vão direto ao último andar para deixar a Presidência e Diretoria, e depois voltando, aleatoriamente, aos outros andares, movimentando-se de acordo com as "patentes" dos presentes. Isso consumiria um tempo muito maior, e dispenderia muito mais recursos, do que simplesmente deixar os ocupantes pela ordem dos andares por onde passa.

Da mesma forma (e pelas mesmas razões), as unidades de disco modernas, ao aceitarem diversos pedidos de gravação e leitura, ordenam o atendimento dos mesmos por sua disposição física no disco, em relação à sua posição atual. É mais rápido e mais eficiente, POREM, não leva em consideração qualquer diferença de importância ou prioridade entre elas.

Dynamic Channel-path Management

A gerência de desempenho dinâmica, permitida pelo WLM, inclui a monitoração do tempo gasto em operações de IO. Caso este seja o maior responsável pela falha no objetivo da SERVICE CLASS, e, dentre seus diversos componentes, o PEND seja o maior valor, o problema pode ser caracterizado como falta de banda ao dispositivo (canais).

Ate o MVS/ESA 4.2 precisaríamos de um IPL para corrigir uma situação deste tipo. Depois, uma carga dinâmica de configuração poderia ser feita, fornecendo canais adicionais para a unidade com problema.

Um dos novos recursos fornecidos pela z900 é a possibilidade de definir um canal (conectado a um SWITCH), como MANAGED. Isso significa que, inicialmente, este canal NÃO SERA atribuído a nenhuma controladora. Mas, caso a situação descrita acima venha a ocorrer, ele poderá ser, automaticamente, fornecido a que apresenta PEND TMEs que afetem o desempenho das SCs do sistema.

Ate o momento, esta funcionalidade suporta somente canais do tipo ESCON e FICON BRIDGE.

IV – Gerência de Armazenamento

1 - Introdução

"O Ministério da Gerência adverte: gerenciar armazenamento pode causar danos ä saúde."

A quantidade e complexidade das soluções de armazenamento cresceram de forma exponencial ao longo dos últimos anos. Então, o que dizer da gerência?

Volumes lógicos, de vários tipos diferentes, são criados em físicos, distribuídos de formas distintas dependendo dos esquemas de proteção e da arquitetura da maquina, Alem disso, uma parcela representativa dos dados que residem em discos magnéticos devem ser copiados, ou por segurança ou para transporte, em cartuchos magnéticos ou via linhas de telecomunicação. Há também um volume considerável de informação que deve ter varias copias ao longo de um curto espaço de tempo, para proteção lógica das informações. Há perdas devido aos esquemas de proteção ou quantidades de espaço mantidas como reserva, perdas pela modelagem lógica, criação de tipos de volumes agregados (como os metavolumes de baixa plataforma). Os mesmos subsistemas são compartilhados por ambientes MainFrame e Open, com características de acesso e necessidades radicalmente diferentes.

Gerenciar? Ultimamente, a maioria das empresas tem empreendido enormes esforços para, pelo menos, tentar entender o que tem, o que se passa ali e o que precisam.

E, claro, os fornecedores, tanto de armazenamento, quanto de ferramentas de software, vêem nessa necessidade uma grande oportunidade de negócios. Por isso, inúmeras soluções vem sendo apresentadas ao longo dos últimos anos. Todas prometendo gerenciar tudo em todos os equipamentos. Ate o momento, nenhuma conseguindo implementar tais promessas na pratica.

A idéia desse capitulo é a de explorar um pouco este território bravio, e talvez, com alguma sorte, lançar algumas migalhas de pão que nos sirvam de guia, um fio de Ariadne que consiga conduzir-nos de volta sãos e salvos.

Creio que a melhor forma de começar seria definir algumas bases para esta proposta. Talvez as definições aqui sugeridas não coincidam com as acadêmicas ou correntemente aceitas, mas nos servirão bastante bem para este trabalho.

A primeira definição importante é a de GERENCIA. Sempre me pareceu algo vago ("... e como vou sair de férias, você gerencia isso pra mim ?? "). Confesso que nunca entendi a expressão.

Por isso, para todos os efeitos desse trabalho, vou assumir a seguinte definição:

Aos que puderem, como eu, considerar uma tal definição como aceitável, surgem, inevitavelmente, algumas implicações que poderíamos definir como corolários.

A primeira implicação óbvia seria a de que NÃO SE GERENCIA NADA antes de os indicadores e seus correspondentes valores e ações terem sido claramente definidos. Há, no universo do armazenamento moderno, milhões de números flutuando em telas de computador, elegantemente impressos em relatórios, os quais representam apenas a ponta do iceberg deste ambiente.

Quantas vezes por dia o operador de fitas levanta-se de sua cadeira? Quantas vezes por segundo a lâmpada de acesso pisca em cada um dos discos? Quantas vezes por minuto o usuário liga para queixar-se a respeito de um problema? Qual a temperatura (média e máxima), debaixo de cada gabinete de armazenamento? Quais critérios devemos empregar para saber ao menos o que nos interessa ou é irrelevante?

Minha sugestão aqui é a que me parece adequar-se melhor ao bom senso. Se vamos instalar um sistema de alarmes, de custo e dificuldade crescentes em função da quantidade de pontos monitorados, será que o primeiro deles seria colocado na casinha do cachorro? A exceção dos inveterados apaixonados por esses animais (entre os quais eu me incluo), é bastante grande a probabilidade do primeiro ser colocado na porta da frente, seguido talvez pela garagem e janelas.

Ou seja, protegemos ou monitoramos primeiro as áreas de maior risco de perda. E, claro, há certas áreas (janelas, por exemplo), que necessitam de mais de um monitor para serem cobertas.

E, é claro, a confiabilidade de cada um dos monitores tem de ser a maior possível, pois eles é que nos avisarão de uma invasão quando não estivermos olhando. Um sensor de baixa confiabilidade pode ser pior que nenhum. Transmite uma falsa sensação de segurança, o que todos sabemos o quão perigoso pode ser.

Um outro ponto a se considerar é que não se gerencia o que não se conhece. Uma "porta dos fundos", desconhecida e, conseqüentemente, desguarnecida, pode por toda a estrutura de proteção a perder.

Portanto, a proposta aqui é a de, uma vez entendida a topologia a ser guarnecida, estabelecer os pontos de proteção (indicadores), selecionando-os depois por sua confiabilidade.

Um último conceito que considero cabível ainda no escopo desta introdução é o de "luzes e medidores". Um mesmo indicador pode "acender" quando um certo valor for atingido, ou mostrar toda a graduação de valores assumidos. É como se ter a escala de um termômetro, para se acompanhar sua variação, ou simplesmente uma lâmpada que se acende quando a temperatura excede um valor máximo de segurança.

2 – Nível de Serviço de Armazenamento

Um primeiro conceito que ocorre facilmente quando se pensa em gerencia de armazenamento é o de ESPAÇO. Ou seja, quanto espaço tenho no total, quanto está livre, quanto ocupado. É natural pensar-se nesses termos, mas a experiência mostra que essa forma de raciocínio acaba sendo uma simplificação excessiva, e de certa forma, uma visão distorcida, do que é, afinal, o recurso de armazenamento.

Já há muito não se pode mais pensar nesse recurso na forma de espaço em NBytes (K, M, G, T, P), pois, caso contrário, 3 chaveiros daqueles "flash-memory" de 2GB em porta USB seriam iguais a 2 X 3390-3. E muitíssimo mais baratos que as unidades que os simulam hoje em dia.

Atualmente, temos de pensar em armazenamento como sendo ÁREA + ATRIBUTOS, os quais abrangem características importantes como DESEMPENHO, NÍVEIS DE SEGURANÇA (número de "backups" tirados, tipo e nível de replicação de dados, tipo de RAID utilizado) e CONECTIVIDADE (ligam-se à canais ESCON, FICON, SCSI, FC-SW, a quantos HOSTS, etc), citando somente algumas. É claro que cada instalação pode ter um conjunto específico de atributos que devem ser considerados, aliados ao primeiro. Um atributo adicional importante é o PERFIL DE USO DE CACHE, dado por características das próprias transações que criarão seus arquivos nesse espaço.

Estes diversos atributos, somados a capacidade em si, tem sido chamados conjuntamente de NÍVEL DE SERVIÇO de armazenamento, e acaba impactando os cálculos de capacidade.

Um bom exemplo disso são as "reservas locais" que podem ser deixadas nos discos físicos, não configurando o máximo de volumes lógicos que seria possível, para o caso de uma necessidade de emergência. Se, durante o transcorrer do processamento diário, os volumes lógicos que foram configurados nesses discos atingirem picos de utilização muito altos (soma > 120 a 150 IOPS por disco físico, por exemplo), o espaço reservado, mesmo estando fisicamente disponível, se utilizado, poderá causar forte impacto sobre o desempenho dos volumes que já estão em funcionamento lá, precisando de uma avaliação mais detalhada para saber se pode ser usado ou não.

Ao longo dos últimos anos, as empresas fornecedoras de discos vem sendo pressionadas por seus clientes, no sentido de disponibilizar soluções cada vez mais baratas. Essa pressão, vem causando um efeito colateral ao qual nem todos ainda deram a devida atenção. Veja a tabela abaixo :

- Linha 1 : Capacidade total do subsistema (fixada em 10 TeraBytes)
- Linha 2 : Capacidade média do maior HD disponível na época
- Linha 3 : Número de HDs necessários para compor a capacidade do subsistema
- Linha 4 : IORate médio sustentável do subsistema, dado o número de HDs presentes (nota 1)
- Linha 5 : IORate por GigaByte

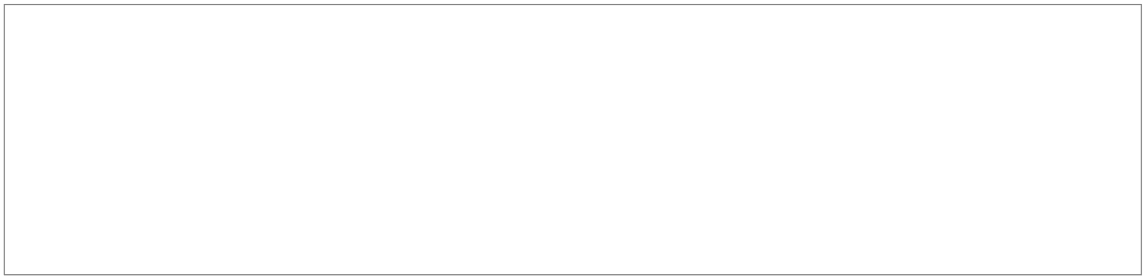
- Linha 6 : US$ médio por GigaByte (nota 2)

  Nota 1: valor médio de 150 IOPS, característico para HDs de 15Krpm

  Nota 2: valor médio entre fornecedores

  * A coluna 2007 assume a disponibilidade de HDs de 1 TeraByte, já divulgados pela mídia especializada, baseados em gravação perpendicular

O aumento de densidade dos HDs ao longo dos últimos anos foi da ordem de mais de 100 vezes, reduzindo a densidade de IO disponível (IOR/GB), na mesma proporção. Enquanto isso, a redução em seu custo foi inferior a 10 vezes. Isso impacta os usuários desse sistema, a medida em que a curva de redução de densidade de IO se aproxima da densidade total necessária para o ambiente. Veja a figura abaixo:



- Linha 1 : Capacidade total, com crescimento de 35% a.a.
- Linha 2 : IORate total necessário à instalação (também crescendo proporcionalmente)
- Linhas 3 a 10 : IODensity disponível à configuração, dependendo do modelo de HD utilizado

A tabela acima mostra que a partir de 2005, apesar de se possuir mais de 8X a capacidade inicial, quando o total de IOR disponível era mais de 30X maior que a necessária, os subsistemas da época só conseguem empatar em nível de serviço. Qualquer um que já tenha tido contato com análise de desempenho vai lembrar que nunca se deve dimensionar um recurso para 100% de sua capacidade. Um outro ponto importante a ser entendido é que a tabela acima assume uma dispersão homogênea de carga, entre todos os HDs disponíveis, o que nunca ocorre num ambiente real de produção. A tabela abaixo mostra os mesmos valores, porem considerando a tradicional regra 8020, na qual 20% dos volumes fazem 80% do total de operações:

Nesse caso, o problema agrava-se mais cedo. E vale a pena notar também que a obsolescência dos primeiros dispositivos, caminha de mãos dadas com a disponibilidade dos novos.

Ainda há muitas instalações que acreditam poder se manter à frente de suas necessidades, simplesmente comprando equipamentos novos e melhores a cada ano. Em muitos casos, isso ainda é verdade, principalmente quando se leva em conta fatores que atenuam os efeitos mostrados acima:

Operações seqüenciais ao cache podem atingir taxas mais altas. A utilização de paralelismos disponíveis nas configurações RAID aumentam as taxas (não por disco físico, mas por volumes lógicos e, conseqüentemente, por arquivos), assim como a taxa de operações de FRONT-END em relação ao BACK-END (um HOST pode requisitar 14 IOs de 4K ao FE, os quais podem ser traduzidos em um único IO de 56K para o BE). Por outro lado, há fatores agravantes, como a proporção 8020 de cada ambiente e suas necessidades de replicação de dados.

É como a conhecida imagem da serpente engolindo seu próprio rabo. O truque consiste em nunca permitir que chegue até a cabeça :) E o nome desse truque é GERÊNCIA POR NÍVEL DE SERVIÇO (!)

3 – Espaço em Camadas

Talvez a melhor maneira de se compreender o "espaço" de armazenamento seja dividindo o mesmo em suas diversas CAMADAS, o que facilita o entendimento de cada uma. Esta proposta baseia-se principalmente nas visões de sistemas operacionais Mainframe IBM, tendo depois os comentários sobre diferenças importantes para ambientes open.

a) CAPACIDADE INSTALADA (ou FÍSICA)

Esta é a primeira, e provavelmente a mais fácil de se entender. É a capacidade TOTAL instalada no ambiente, sendo igual ao número de discos físicos vezes a capacidade de cada um. Seu valor deve ser obtido através de ferramentas do fabricante dos equipamentos, seja diretamente, via intervenção humana, ou através da codificação de APIs, as quais a maioria dos fornecedores já disponibilizou.

b) CAPACIDADE UTILIZADA

Este é o valor formado pela soma das capacidades CONFIGURADA, PERDIDA e/ou RESERVADA LOCALMENTE (veja abaixo). Se subtraído da capacidade INSTALADA, resulta no montante de área ainda disponível diretamente para configuração.

c) CAPACIDADE PERDIDA / RESERVADA LOCALMENTE

O processo de divisão da capacidade instalada em volumes lógicos resulta sempre em alguma perda de espaço. Essa perda deve-se a fatores como áreas de mapeamento e controle no volume físico, bem como a perdas por "encaixe", ou seja, áreas instaladas que não puderam ser configuradas por não ter espaço disponível na granularidade desejada. Um exemplo seria a divisão de um disco de 73 GBs em volumes lógicos 3390-3. Arredondados para 3GB cada, poderíamos definir 24 volumes lógicos (72 GBs), e sobraria 1.

Essa perda está sempre relacionada ao algoritmo utilizado pelo fabricante no mapeamento, no tipo de RAID e opção de volumes lógicos e estimada, em linhas gerais, como estando entre 2 e 5 % (mínima).

Além disso, a própria instalação pode optar por deixar nesses discos físicos uma certa área de reserva, não configurando o máximo de volumes possível. Tal decisão poderia ter por base uma necessidade operacional (aumentar o desempenho dos volumes configurados, por exemplo), ou de negócio (espaço já instalado, e configurável mais rapidamente que o restante). Alguns fabricantes permitem a definição de volumes lógicos, dessa forma, mais diretamente pelo cliente. Outros já exigem processos mais impactantes.

d) CAPACIDADE CONFIGURADA

Simplisticamente, pode ser definida como sendo a soma das capacidades dos volumes lógicos formatados e, como prefiro considerar, associados a portas de conexão em cada controladora. Essa é a capacidade liquida sendo oferecida aos diversos hosts.

e) CAPACIDADE DEFINIDA

Até este ponto, as capacidades eram tratadas como as existentes DENTRO dos subsistemas de armazenamento em si. Já a capacidade definida é calculada nos termos DE CADA HOST que acesse volumes lógicos nesses subsistemas. Em termos de MainFrame IBM (z/OS), tratá-se da definição feita via HCD (Hardware Configuration Definition). As outras plataformas possuem formas distintas de definir essa capacidade ("mounts" em ambientes Unix, por exemplo). De qualquer forma,

é o que será efetivamente utilizado pelo ambiente, pois a capacidade sem definição não é usada por ninguém.

Um ponto importante a se manter em mente aqui é que a SOMA das capacidades DEFINIDAS pode resultar MAIOR que o total CONFIGURADO. E o motivo disso é o compartilhamento. No caso de ambientes MF, é comum a pratica de ter-se uma mesma unidade (lógica), definida para mais de um sistema.

f) CAPACIDADE DISPONÍVEL

Trata-se de um subconjunto da capacidade definida. Uma vez tendo acesso a um determinado grupo de volumes lógicos, uma instalação pode optar por deixar partes destes volumes OFF-LINE, talvez como uma forma de isolamento entre os diversos hosts (o que não seria a maneira mais apropriada de se fazer isso), talvez como uma "reserva rápida", que poderia ser posta em produção imediatamente, caso fosse necessário.

g) CAPACIDADE RESERVADA ("ALLOCADA")

Em os volumes estando definidos e disponíveis, os processos do sistema passam a reservar pedaços dos mesmos para seu uso (DATASET SPACE). Cada pedaço passa a ter um nome (DATASET NAME), um proprietário (ou grupo), e regras associadas de acesso.

É de bom-tom que não se mantenha reservada uma área em disco da qual não se necessite. Este cavalheirismo pode ser conseguido através da opção RLSE do parâmetro SPACE durante sua criação, o que faz com que o espaço reservado, mas não efetivamente preenchido com dados, seja liberado ao final do uso (CLOSE). Entretanto, como indicado acima, isso se constitui numa recomendação de boa conduta, e não uma regra sintática do sistema, podendo, portanto, ser ignorada (um exemplo disso seria o caso da criação de um banco de dados, no qual ainda não se consegue, no momento da criação, estimar a quantidade em bytes que suas tabelas ocuparão, depois de carregadas).

h) CAPACIDADE USADA

Finalmente, aqui, chegamos aos bytes efetivamente gravados nas superfícies dos discos pelas aplicações. Uma exceção a este conceito aplica-se no caso de equipamentos com compressão. Para esses, o espaço usado ainda varia em função da compressibilidade dos dados.

Para baixas plataformas, há alguns fatores adicionais que devem ser levados em conta, como por exemplo os gerenciadores de volume (VOLUME MANAGERs). Estes possibilitam a concatenação de diversos volumes lógicos, conforme entregues pelos subsistemas, em um único volume lógico a ser oferecido à aplicação. Devido a sua flexibilidade, os gerenciadores de volume atuais permitem, por exemplo, que volumes lógicos de subsistemas diferentes sejam concatenados em uma única visão. Isso, em alguns casos, pode gerar perdas por formatação (lógica, não mais física), e resultar em interpretações incorretas de relatórios de desempenho ou capacidade por site (as unidades concatenadas encontram-se em prédios diferentes, interligadas via SAN, por exemplo).

V – Recuperação de Contingências

1 - Introdução

"THE CONTINGENCY PLANNER"

"There once was a job no one wanted,
It was demanding, impersonal, undaunted,
Since no one applied - they took me aside,
"Contingency Planner" they flaunted." (continues...)

Don Edwards - Contingency Planner for Puget Sound Bank, Tacoma - WA - 1997

–

Para se preparar contra contingências, deve-se primeiro compreender o que são. Uma vez entendidas, deve-se então decidir para quais se deseja estar preparado, e a partir dai, definir seus planos.

O ponto mais estranho em relação à sentenças tão simples e óbvias como as acima, é que mesmo após tantos anos de atividades nessa área (o DR Journal, do qual sou assinante há 15 anos, já existe, oficialmente, desde 1987), ainda haja tantas dúvidas e erros conceituais nessa área.

Planejamento de Contingência, seja ela chamada pelo nome que se quiser (Recuperação de Desastres, Continuidade de Negócios, ou a mais atual, Resistência dos Negócios - "Business Resilience), NÃO COMEÇA no IT, mas nas diversas áreas de negócios da empresa. É incrível a quantidade de processos deste tipo que, ainda hoje, são iniciados e liderados por pessoas de tecnologia. Montam esquemas de replicação, remota e local, scripts de inicialização, definição de conectividades, sem fazer a menor idéia dos valores principais envolvidos.

E sempre, sempre que um processo destes começa, a história é a mesma: "Queremos replicação SÍNCRONA, RTO=0, RPO=0, automação COMPLETA do processo". Depois de enormes esforços, quando do recebimento dos primeiros orçamentos, a situação começa a mudar.

Alguns pontos interessantes para consideração

I - Nenhum esquema de recuperação remoto será mais rápido do que os já possíveis no ambiente local, o qual, geralmente, JÁ NÃO É IGUAL A ZERO !!!!

Qual é o RTO (Recovery Time Option), e o RPO (Recovery Point Option), que a instalação possui em seu site local ? Caso caia a força, por exemplo, quanto tempo se leva, localmente, para repor os sistemas, bancos de dados restaurados, controladores de online, etc, no ar novamente, com todos os dados íntegros (RTO local)? Quanto tempo se perde daquilo que estava em processamento (RPO local) ?

Arquivos que estavam em uso devem ser fechados e corrigidos. Fitas que pararam em meio a gravação devem ser recuperadas. Bases de dados devem ser corrigidas. Discos podem precisar de uma verificação de integridade. Transações executando no momento da falha tem de ser canceladas. Sistemas e processos batch tem de ser reiniciados.

Portanto, mesmo com replicações síncronas, os dados, remotamente, estarão (nos discos), iguais aos da instalação original. Faltam os dados em cache, memórias de servidor, etc. Tempos locais típicos variam de 1 a 2 horas.

II – Quanto tempo se leva para DECLARAR CONTINGÊNCIA ?

Algumas contingências são óbvias, como a queda de força na instalação. Mas desse ponto em diante, quanto tempo se leva até que se declare uma situação de contingência? Alguém deve conversar com eletricistas locais, para saber se a falha vai durar muito tempo (ativar a contingência exige sacrifícios, tempo, etc). Normalmente, não se vai desejar ativar o site remoto quando se sabe que a energia volta em menos de meia-hora.

Detectar um estado de contingência, por si só, já leva tempo. A maioria das instalações, após alguns problemas (geralmente), descobre que AUTOMATIZAR a declaração de contingência não é uma boa idéia. Um simples "loop" num servidor qualquer pode ser mal interpretado, e toda a instalação é desativada sem uma boa razão para isso.

III – Do que mais se necessita, além dos discos ?

Pessoas precisam estar disponíveis, mesmo que remotamente, para ativação dos ambientes de contingência. Documentações precisam ser desviadas. Logísticas inteiras tem de ser postas em ação. Acessos aos ambientes remotos, procedimentos de backup. A lista é enorme.

Fim do Documento SSK0151