PC partitioning and OS2 LVM

Jan van Wijk

Principles of disk-partitioning as used on
most personal computer systems, including the
OS/2 Logical Volume Manager extensions

FSYS - *Software*   *DFSee*

Presentation contents

- Who am I

- Physical disk layout
- Partition-tables
- Primary versus Logical partitions

- OS/2 and eCS Logical Volume Manager

- Examples using DFSee ...

Who am I ?

Jan van Wijk

- Software Engineer, C, Rexx, Assembly
- Founded FSYS Software in 2001
- First OS/2 experience in 1987, developing parts of OS/2 1.0 EE (Query Manager, later DB2)
- Used to be a systems-integration architect at a large bank, 500 servers and 7500 workstations

- Home page: *http://www.dfsee.com*

FSYS - *Software*

DFSee

What is ...

- ## Disk partitioning

    - dividing the available disk space over one or more separate areas called 'partitions' that each have there own filesystem structures

- ## A Filesystem

    - A way to organize data-structures on a disk (partition) to allow storage of file data, and retrieve that data
    - Software (driver) to work with the filesystem

Why use partitions ?

- To keep things separate, like applications and data, operating system, swap-space

- To have different/multiple driveletters (PC)

- To use the full disk-size when the filesystem is limited in size (FAT 2GiB, HPFS 64 GiB)

- To use more than one OS (multi boot)

- Because most operating systems require partitioning data, even with a single partition
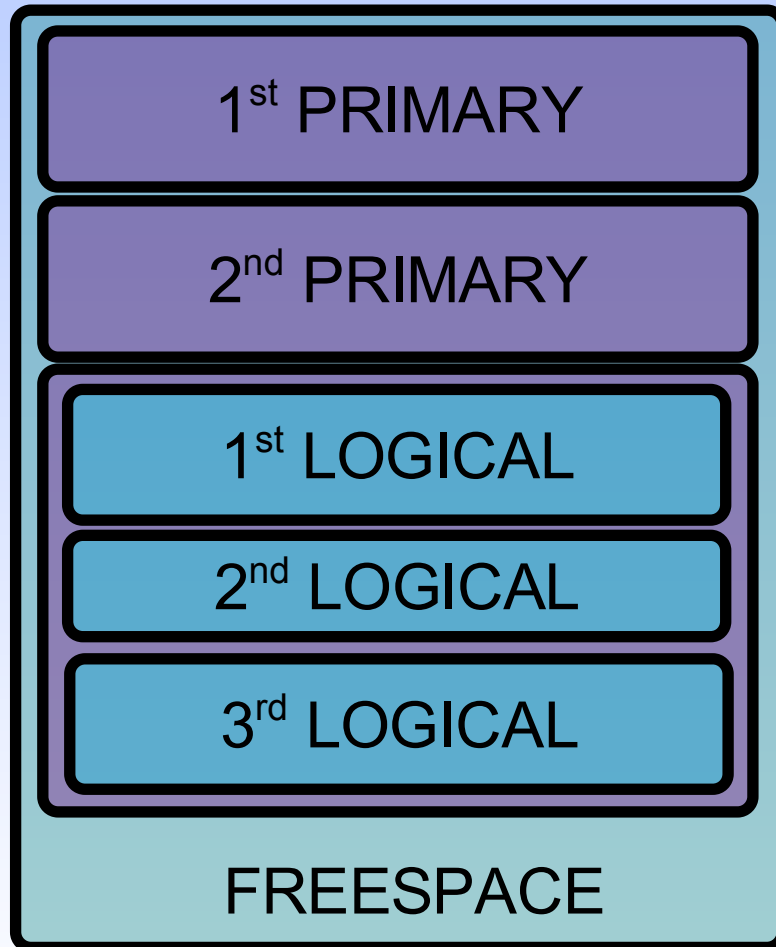
Partitioning methods

- **On PC-systems, in almost all cases the classic partition-tables will be used:**

  - Allow dividing the disk in 4 PRIMARY areas
  - Allow subdividing one of these in LOGICAL areas
  - Include 'bootable' indicator and actual bootcode

- **Other alternatives, PC or others are:**

  - Dynamic disk database, MS Win-2000 and later
  - Macintosh disk partitioning
  - Several Unix and Mainframe methods for partitioning

FSYS - *Software*   © 2007 JvW   DFSee

Classic PC partitioning

- Each disk (partionable media) has an MBR or Master Boot Record containing bootcode and a 4-entry partition table

- The system BIOS will boot using the code in the MBR on 1st (or selectable) disk

- The bootcode in the active partition on that disk will be used to continue booting
This is either an OS, bootmanager or both

FSYS - *Software*

DFSee

Partitioned disk

Partitioned disk

1st PRIMARY	Primary partition
2nd PRIMARY	Primary partition
1st LOGICAL	Extended partition in this case containing 3 Logical partitions.
2nd LOGICAL	
3rd LOGICAL	*EXTENDED CONTAINER*
FREESPACE	

FSYS - *Software*

DFSee

Partition table

Partition Table

Start End	Entry-1
Start End	Entry-2
Start End	Entry-3
Start End	Entry-4

Partition-table entry layout
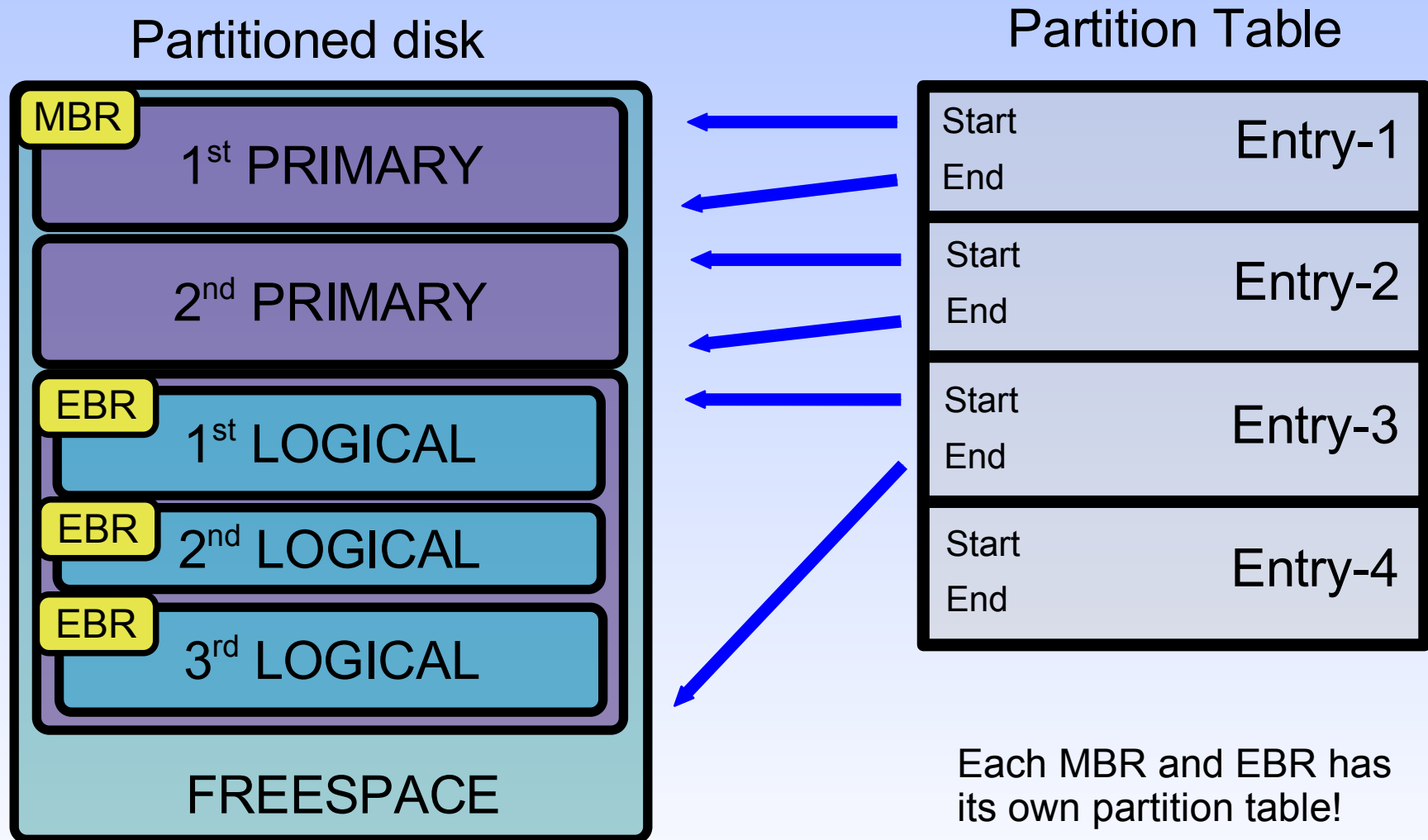
- 1   Flag   Active/bootable indicator
- 3   CHS-start   Start location as CHS
- 1   Type   System-type
- 3   CHS-end   End location as CHS
- 4   LBA-offset   Linear offset (start)
- 4   LBA-size   Size in sectors

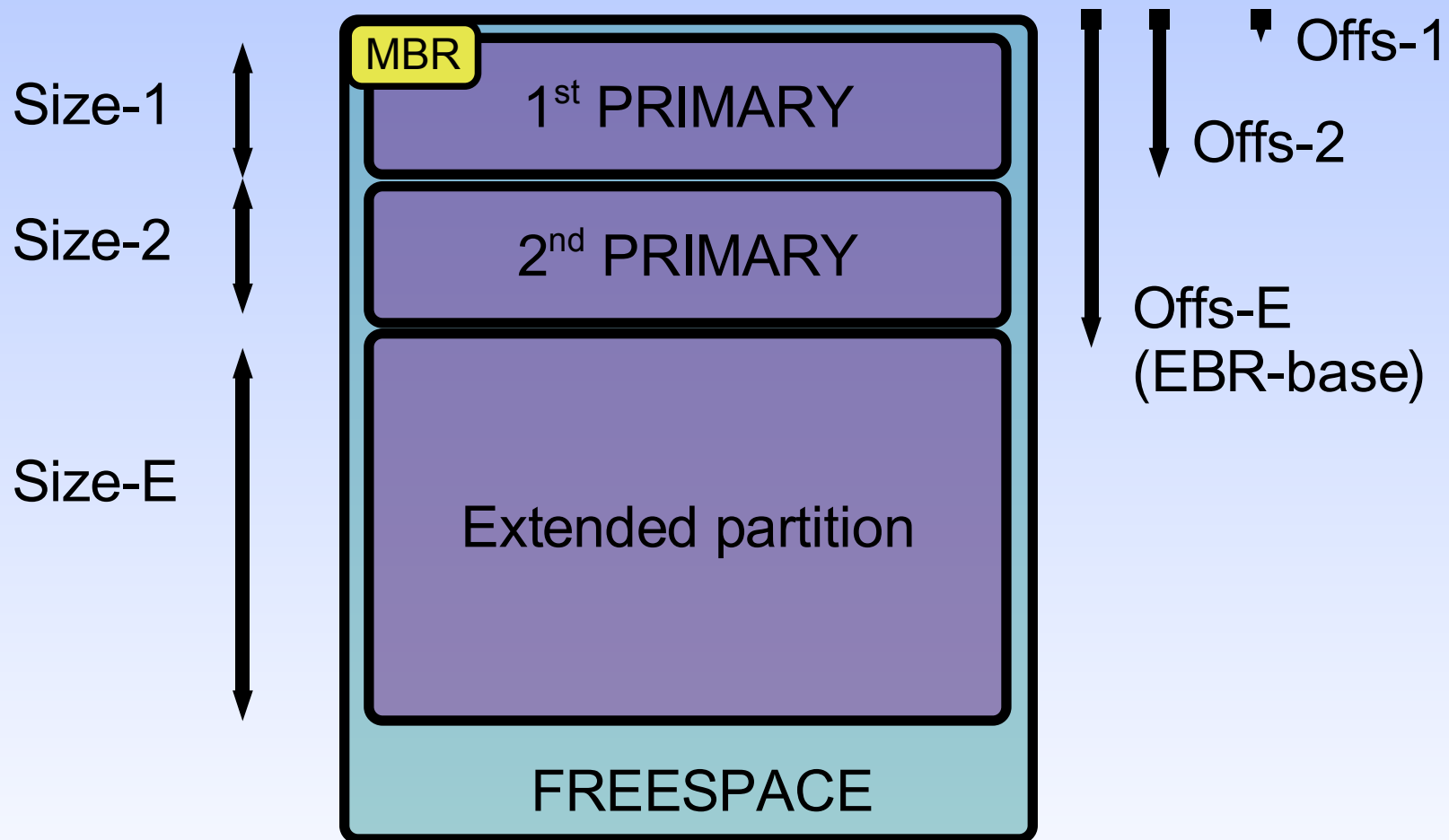There are FOUR such entries in each MBR or EBR, starting at byte offset 0x1BE

FSYS - *Software*

DFSee

PC partitioning properties

- A partition table has a maximum of 4 entries
    - MBR can hold 4 PRI, or 3 PRI and 1 extended
    - EBR usually holds 1 logical, and 1 'next' extended partition, forming an infinite linked list of logicals

- Partitions can be 'hidden' using a type value
    - One primary or multiple visible ones allowed by OS
    - Hiding logicals is usually not supported
    - Real visibility also depends on other OS specific settings, and availability of the right FS drivers

- BIOS boots primaries only (on some disks)
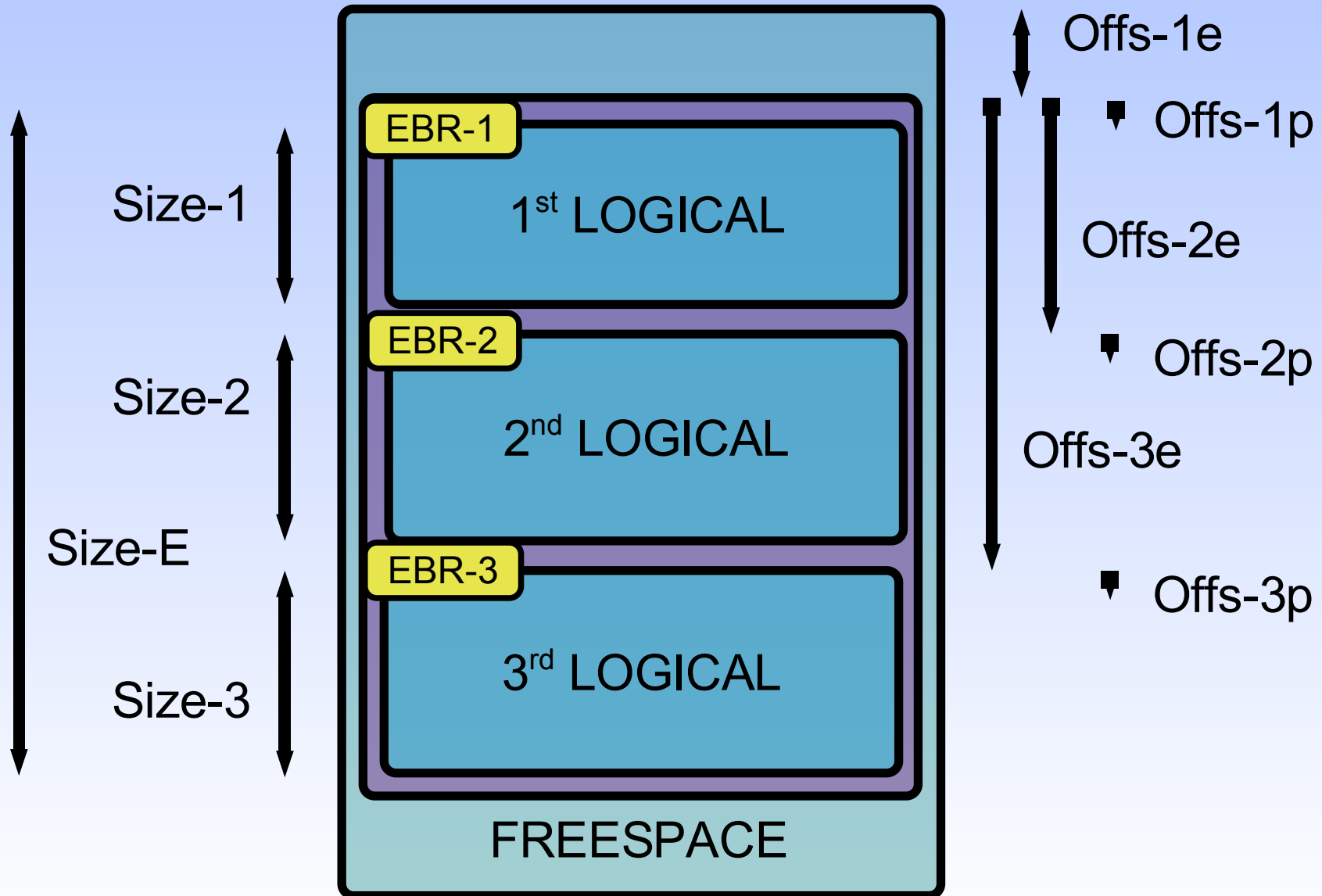   - Bootmanagers might allow boot from logical (OS2)

FSYS - *Software*

DFSee

MBR-EBR placement

Partitioned disk

MBR	1st PRIMARY

2nd PRIMARY

EBR	1st LOGICAL

EBR	2nd LOGICAL

EBR	3rd LOGICAL

FREESPACE

Partition Table

Start	
End	Entry-1

Start	
End	Entry-2

Start	
End	Entry-3

Start	
End	Entry-4

Each MBR and EBR has its own partition table!

MBR partition table

EBR partition tables

MBR/EBR table values

- Offs-1e   Size-E   = Extended container

- Offs-1p   Size-1   = 1st logical partition
- Offs-2e   Size-2   = 2nd extended

- Offs-2p   Size-2   = 2nd logical partition
- Offs-3e   Size-3   = 3rd extended

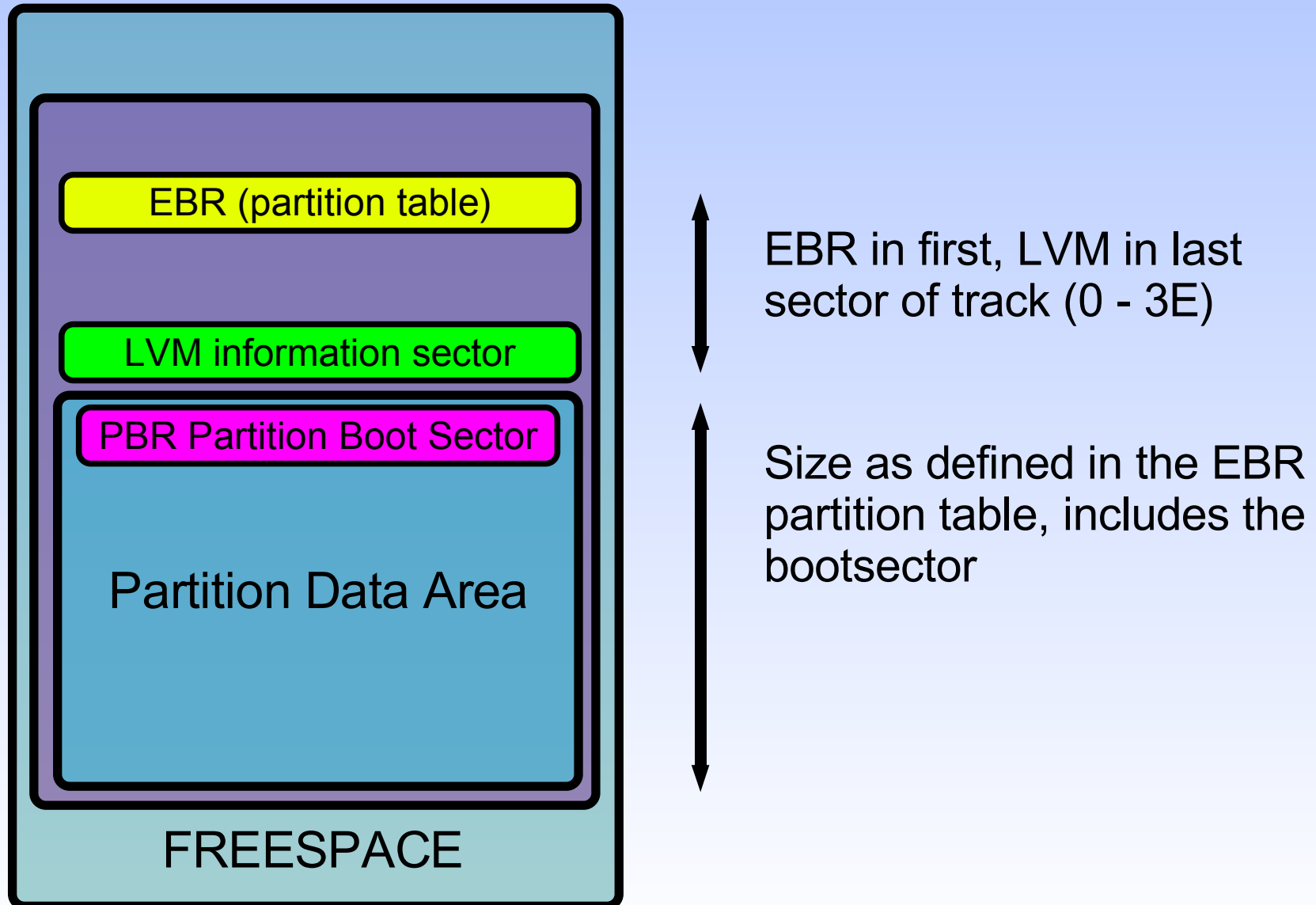- Offs-3p   Size-3   = 3rd logical partition

What does LVM add ?

- The volume concept, where a volume is used to store a complete filesystem

- Volumes can have multiple partitions on any number of disks (disk spanning)

- 20-character descriptive names for disks, partitions and volumes
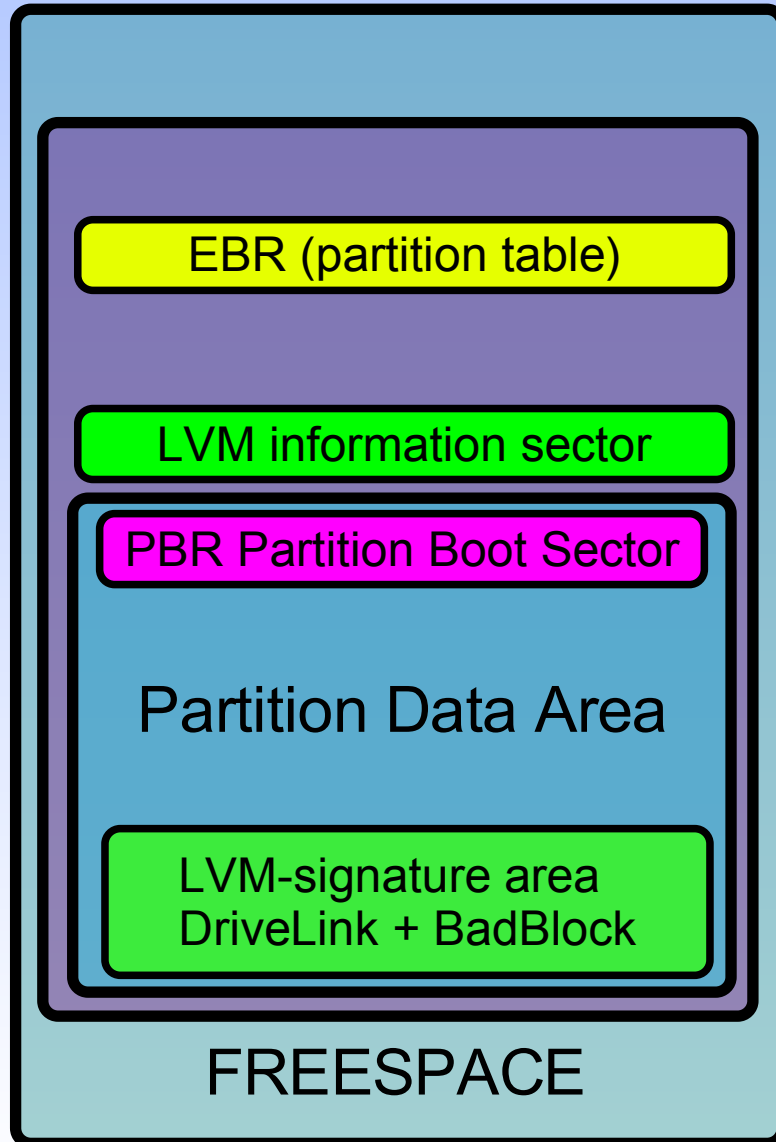
- Free assignment of driveletters

Why use LVM, advantages

- Dynamic expansion of volumes is possible by adding more partitions (OS2: JFS only)

- Can use one large volume on multiple disks

- Driveletter assignments do not change when partitions are added or removed

- LVM can handle bad-block relocation for the filesystem (OS2: JFS only)

FSYS - *Software*

© 2007 JvW

DFSee

LVM compatibility volume

EBR (partition table)

EBR in first, LVM in last sector of track (0 - 3E)

LVM information sector

PBR Partition Boot Sector

Size as defined in the EBR partition table, includes the bootsector

Partition Data Area

FREESPACE

LVM volume, usually JFS

EBR (partition table)

LVM information sector

PBR Partition Boot Sector

Partition Data Area

LVM-signature area
DriveLink + BadBlock

FREESPACE

EBR in first, LVM in last
sector of track (0 - 3E)

Size as defined in the EBR
partition table, includes the
bootsector and LVM cylinder

Nett partition size seen by the
filesystem upto 1 cylinder LESS
than seen in partition table!

LVM-info structure layout

- Used by compatibility and LVM (JFS) volumes
- IBM name: Drive Letter Assignment Table

- Generic info in each DLAT sector:
    - Disk geometry, diskname and disk-id value
    - Boot-disk-id, sector CRC-value and install flags

- Partition specific information (max 4 entries):
    - Start PSN of partition and size in sectors
    - Volume-id and Partition-id values
    - On-BM-menu flag and installable flag
    - Volume and Partitions names   (20-character max)
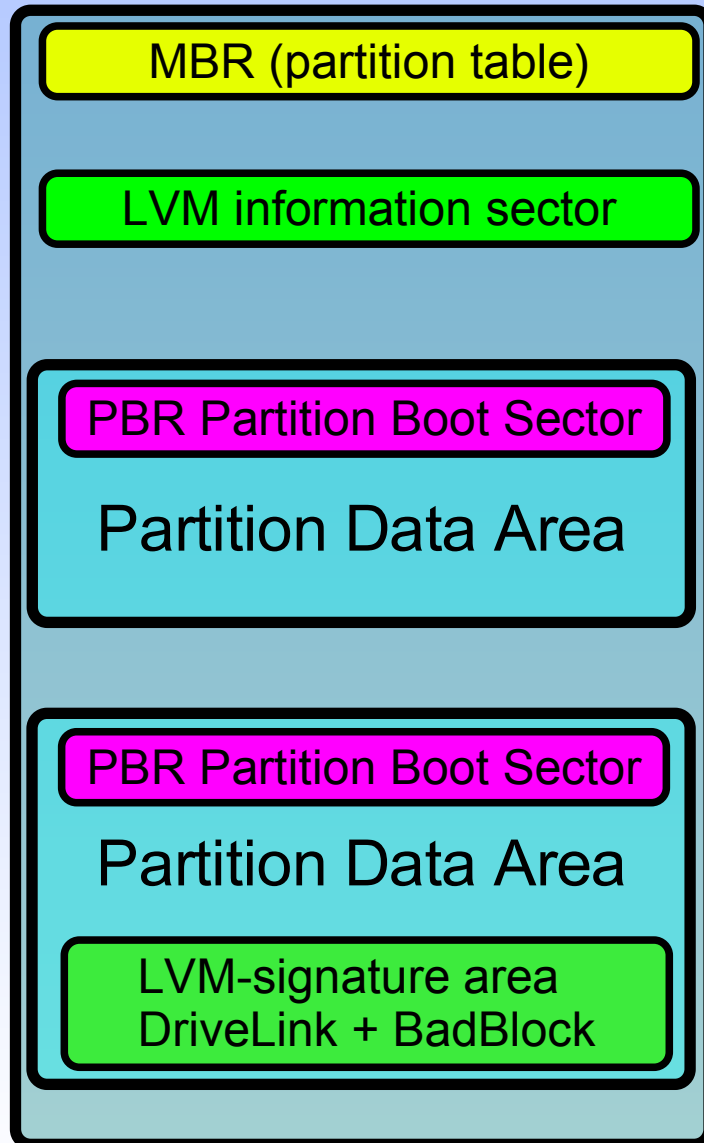   - Assigned driveletter, if any   (zero if hidden)

FSYS - Software

DFSee

LVM-signature structure layout

- ▪ **Used by LVM (JFS) volumes only**

- ▪ **Partition specific information (always 1):**
  - ▪ Start PSN, last PSN and partition size in sectors
  - ▪ Boot-disk-id and sector CRC-value
  - ▪ Volume-id and Partition-id values
  - ▪ Volume and Partitions names   (20-character max)
  - ▪ Assigned driveletter, if any   (zero if hidden)
  - ▪ Fake-EBR sector PSN, describes whole volume
  - ▪ Partition comment-string (not used by LVM yet)
  - ▪ LVM major and minor version number
  - ▪ Disk name, 20 character description
  - ▪ Sequence and aggregate information (unused ?)

FSYS - Software

DFSee

LVM features linked to signature

- The LVMsig sector can link to multiple features which will be located in the same (last) cylinder

- For the Warp 4.5 LVM only two are defined:

  - Bad Block Relocation data (BBR)
    An area of sectors set apart to be replacements for sectors detected as 'bad', managed by LVM

  - Drive Link data (DL)
    Listing all partitions (by partition-id) that are part of the same volume, and also their ordering

FSYS - Software   © 2007 JvW   DFSee

LVM info for primary partitions

MBR (partition table)

LVM information sector

PBR Partition Boot Sector

Partition Data Area

PBR Partition Boot Sector

Partition Data Area

LVM-signature area
DriveLink + BadBlock

MBR in very first sector (0)

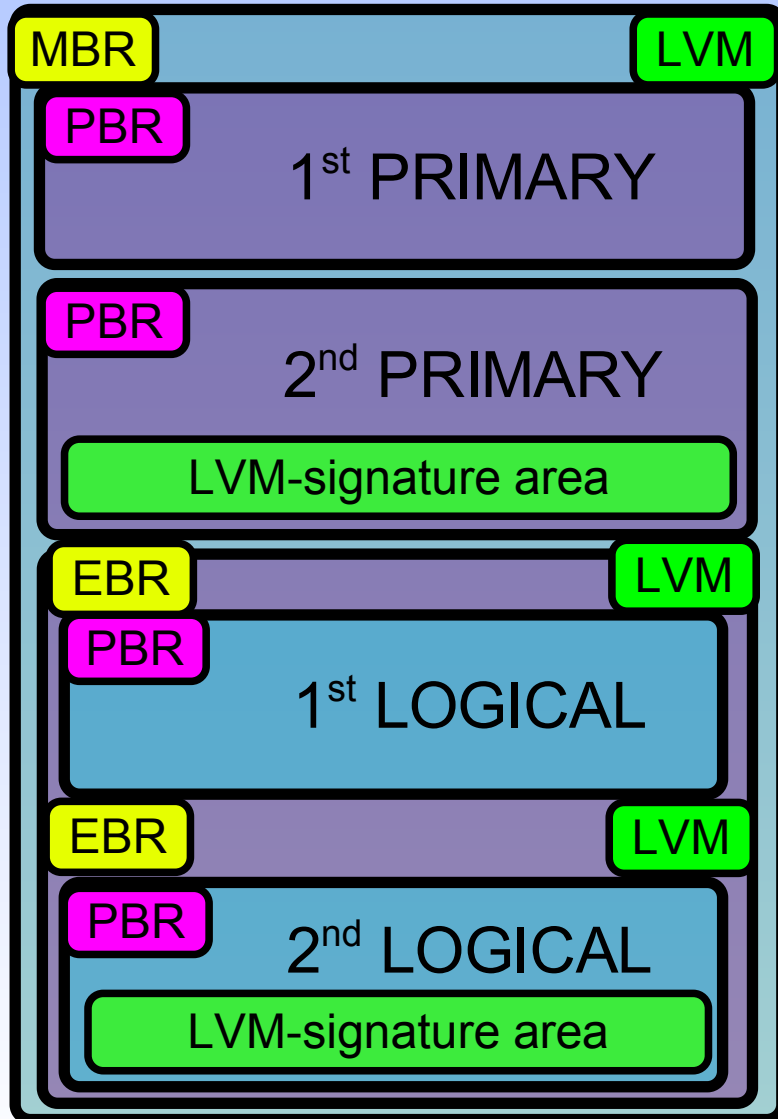LVM info for upto 4 primaries
in last sector 1st track (often 3E)

Primary Compatibility Volume.
Size as defined in MBR table
includes bootsector

Primary LVM (JFS) Volume.
Size as defined in MBR table
includes bootsector AND the
LVM signature area

MBR-EBR-LVM placement

MBR ... LVM	First track, MBR (0) and LVM (3e)
PBR — 1st PRIMARY	Primary Compatibility partition
PBR — 2nd PRIMARY — LVM-signature area	Primary LVM-JFS partition
EBR ... LVM / PBR — 1st LOGICAL	Logical Compatibility partition
EBR ... LVM / PBR — 2nd LOGICAL — LVM-signature area	Logical LVM-JFS partition

FSYS - Software   PC partitioning principles, including OS/2 LVM   © 2007 JvW   DFSee

CHS addressing, background

- CHS values were present in the tables because they are exactly in the format required by the standard programming interface of these days being the 'interrupt 13' calls in DOS/BIOS

- The specific fields are sized to fit in 3 bytes, taking up minimum CPU-register space.

- This choice has led to limits and restrictions in CHS style addressing that still causes problems today (backward compatibility)

FSYS - *Software*

DFSee

CHS value field layout

- hhhhhhhh   CCssssss   cccccccc

- C = Cylinder, 10 bits   range 0 .. 1023
- H = Heads, 8 bits   range 0 .. 254
- S = Sectors, 6 bits   range 1 .. 63

  - Head value 255 normally not used (DOS/BIOS bug)
  - Sector value 0 is illegal, numbering is 1-based
    (cylinder and head are both zero-based)

- CHS values are NOT really used anymore,
  but are often CHECKED by partitioning tools

CHS related limits

- The values and ranges specified in the last slides lead to the dreaded '1024 cylinder' limit

- To maximise the addressable size with this limit, many systems use 'translation' to maximise the number of heads and sectors (255, 63) in order to minimise the number of cylinders leading to a cylinder size of 7.8 MiB, and a disksize that is directly addressable of almost 8 GiB

- Sectors beyond the limit can't be represented by CHS fields, and are replaced by DUMMY values

FSYS - *Software*

DFSee

CHS, dummy standards

- **IBM/DFSee style:**
  - Use maximum valid values for all fields
  - Typical values on 255/63 geo   : C:1023 H:254 S:63

- **PowerQuest (partition magic) style:**
  - Use maximum value for Cylinder only, rest real value
  - Typical start on 255/63 geo: C:1023 H:0   S:1
  - Typical end on 255/63 geo: C:1023 H:254 S:63

- **Microsoft style:**
  - Use maximum (invalid) values for all fields
  - Typical values on 255/63 geo   : C:1023 H:255 S:63
  - This leads to a combined CHS value of 0xFFFFFF

CHS, warnings and errors

- Using dummy values that do NOT conform to the 'dummy style' of your tools will lead to errors or warnings, and sometimes prevent you from making any changes, until corrected
  - DFSee: use 'FIXCHS' with the '-c:style' option

- Using incorrect CHS values WITHIN the limits is always considered an error, and most tools will not allow any changes to be made
- A cause for this is a change in used geometry
  - DFSee: use 'FIXCHS' to update the CHS values, or 'GEO cc hh ss' to change used geometry

Questions ?

FSYS - *Software*   **DFSee**