DASD device driver

S/390's disk devices (DASDs) are managed by Linux via the DASD device driver. It is valid for all types of DASDs and represents them to Linux as block devices, namely "dd". Currently the DASD driver uses a single major number (254) and 4 minor numbers per volume (1 for the physical volume and 3 for partitions). With respect to partitions see below. Thus you may have up to 64 DASD devices in your system.

The kernel parameter 'dasd=from-to,...' may be issued arbitrary times in the kernel's parameter line or not at all. The 'from' and 'to' parameters are to be given in hexadecimal notation without a leading 0x.
If you supply kernel parameters the different instances are processed in order of appearance and a minor number is reserved for any device covered by the supplied range up to 64 volumes. Additional DASDs are ignored. If you do not supply the 'dasd=' kernel parameter at all, the DASD driver registers all supported DASDs of your system to a minor number in ascending order of the subchannel number.

The driver currently supports ECKD-devices and there are stubs for support of the FBA and CKD architectures. For the FBA architecture only some smart data structures are missing to make the support complete.
We performed our testing on 3380 and 3390 type disks of different sizes, under VM and on the bare hardware (LPAR), using internal disks of the multiprise as well as a RAMAC virtual array. Disks exported by an Enterprise Storage Server (Seascape) should work fine as well.

We currently implement one partition per volume, which is the whole volume, skipping the first blocks up to the volume label. These are reserved for IPL records and IBM's volume label to assure accessibility of the DASD from other OSs. In a later stage we will provide support of partitions, maybe VTOC oriented or using a kind of partition table in the label record.

USAGE

-Low-level format (?CKD only)
For using an ECKD-DASD as a Linux harddisk you have to low-level format the tracks by issuing the BLKDASDFORMAT-ioctl on that device. This will erase any data on that volume including IBM volume labels, VTOCs etc. The ioctl may take a 'struct format_data *' or 'NULL' as an argument.

```
typedef struct {
        int start_unit;
        int stop_unit;
        int blksize;
} format_data_t;
```

When a NULL argument is passed to the BLKDASDFORMAT ioctl the whole disk is formatted to a blocksize of 1024 bytes. Otherwise start_unit and stop_unit are the first and last track to be formatted. If stop_unit is -1 it implies that the DASD is formatted from start_unit up to the last track. blksize can be any power of two between 512 and 4096. We recommend no blksize lower than 1024 because the ext2fs uses 1kB blocks anyway and you gain approx. 50% of capacity increasing your blksize from 512 byte to 1kB.

-Make a filesystem
Then you can mk??fs the filesystem of your choice on that volume or
partition. For reasons of sanity you should build your filesystem on
the partition /dev/dd?1 instead of the whole volume. You only lose 3kB
but may be sure that you can reuse your data after introduction of a
real partition table.

BUGS:
- Performance sometimes is rather low because we don't fully exploit clustering

TODO-List:
- Add IBM'S Disk layout to genhd
- Enhance driver to use more than one major number
- Enable usage as a module
- Support Cache fast write and DASD fast write (ECKD)