SD Card Partitioning & Formatting

Every storage device has one or more logical partitions and default mount points. A partition is a separate, independent logical storage unit (device). If a storage medium has two partitions, each of them is treated as individual unit on machine and for each of them we need separate device files to perform i/o operations. All connected storage devices and partitions are represented by appropriate **sdxy** dev files in **/dev** directory (sd: **s**csi **d**isk, **x**: unique character assigned to each disk, **y:** partition number; both x and y are in incremental order), such as **/dev/sda**, **/dev/sda1**, **/dev/sda2**, **/dev/sdb, /dev/sdb1, /dev/sdb2** and so on. This information is available through Linux disk partitioning utility **fdisk**. Terms, disk and card will be used interchangeably throughout this document to refer to the microSD card.

Disk Partitioning

Linux has many utilities/tools for disk partitioning. The widely used one is **fdisk**.

Following steps are about managing disk partitions:

- Connect sd card to system.
- Use fdisk to get detailed information about all the storage devices and partitions available in the system (including sd card):

 # fdisk -l

Look for dev files and sizes for card and partitions on it. dev file is needed to perform any of the fdisk operations on the card (partition creation, deletion, updating partition table etc.), as through it only fdisk accesses the

card.

Usually, connected device is represented by the dev file sdb and partitions on it by sdb1, sdb2 and so on. It depends on how many storage devices were already present in the machine when this new device was attached to it, taking next available char for the device. Such as sdb, sdc, sdd and so on.

Note: Before performing any operation on disk, unmount all existing partitions on it.

 # umount /dev/sdb1 **or** *# umount /media/<user>/<partition1-label>*

 # umount /dev/sdb2 **or** *# umount /media/<user>/<partition2-label>*

Now use **fdisk** to manage/create new partitions, for that pass the dev file (dev file for the device, not the ones for partitions) to fdisk as an argument:

 # fdisk /dev/sdb

- Once in fdisk, enter command **"m"** to get a list of all available operations and corresponding commands for them.

- We start off by first deleting all existing partitions on the card and creating a fresh partition table for it, as following:
    - enter **"d"** to delete a partition
    - enter appropriate **partition number** to be deleted
    - repeat above two steps to delete all existing partitions
    - use command **"p"** to print partition table

NOTE: Command "p" can be used anytime to check partition table current status

- Create new partitions
    - to create a new partition enter **"n"**

- for partition type (p: primary, e: extended), **use selected default value** i.e. primary, no need to manually provide it, **just enter**
- similarly, **use selected default values** for *partition number* and *first sector* too.
- for *size* enter value **+256M (Generally 256MB is enough)**.
- repeat all previous steps to create another partition. except, this time around **use selected default values** for size too, to allocate, remaining free (unallocated) space completely to it.
- by this time we will have two partitions created on the device, One of size 256 MB and other with rest of the space on the card. significance and need for these two partitions will be discussed in next section.

- First partition is used for loadable binary/image files and second for root file system. These loadable binary/image files are read by u-boot during boot-up while rootfs is read and mounted by kernel. BootRom Code & Bootloader(u-boot) for embedded boards is compatible with FAT file system. For this reason we need to change the partition system id of first partition to FAT32, we need to set the boot flag for it also, and is done as following:

- To change the system id of partition 1 enter command "**t**"
- To check all the partition system id options available, enter **"L"**
- Enter **"b"** to change the partition 1 id from 83 (Linux) to W95 FAT32
- Set boot flag for partition 1, using command "**a**"
- Write all the changes we have made so far into partition table, for that enter command "**w**".

SD Card Formatting

Formatting a partition means creating the file system on it, to access and manage data on it. Till now we have partitioned the card, chosen the file systems but haven't created any on it yet, and for that reason, it's still not accessible. We will use **FAT32** file system for **boot partition** and Linux **ext3** file system for **root partition**.

To format a disk partition in Linux, widely used tool is **mkfs**.

Note: Before formatting the partitions, unmount them.

 # umount /dev/sdb1 or *# umount /media/<user>/<partition1-label>*

 # umount /dev/sdb2 or *# umount /media/<user>/<partition2-label>*

Format first partition to FAT32 file system:

 # mkfs.vfat -F 32 /dev/sdb1 -n BOOT

Format second partition to ext3 file system:

 # mkfs.ext3 /dev/sdb2 -L ROOT

Summary

Partitioning

1. list out all disks and partitions using fdisk utility
   # fdisk -l
2. Unmount all partitions on the card.
   # umount /dev/sdb1
   # umount /dev/sdb2
3. manipulate partition table on the disk using fdisk
   # fdisk /dev/sdb
4. delete all existing partitions using command **"d"**
5. create two new partitions using command **"n"**, allocate +256M of space for first partition, allocate remaining space to second partition.
6. change first partition system id to W95 FAT32, using command **"t"** then option **"b"** from partition system id list.
7. set the boot flag for partition 1 using command **"a"**.
8. Update (write) partition table using command **"w"**.

Formatting

9. Unmount all partitions on the card.

   # umount /dev/sdb1

   # umount /dev/sdb2

10. mkfs.vfat -F 32 /dev/sdb1 -n BOOT
11. mkfs.ext3 /dev/sdb2 -L ROOT