Cross-Chapter Handout: SD Card Format Procedure for Tang Primer 25K / Tang Nano 20K + General Embedded Use

958 words · ~4 min read

⚠️ SAFETY BANNER. `umount` BEFORE PHYSICAL REMOVAL ⚠️

A FAT32 card mounted on a Linux host has a kernel write cache that may not have flushed. Pulling the card without `umount` first can corrupt the filesystem, even if no application appears to be writing.

Before you physically remove an SD card from the Linux host:

```
sudo umount /dev/sdb1   # or whatever your device path is
```

Wait for the umount to return (it forces a cache flush). THEN physically remove. If `umount` says "target is busy," do not pull. Find what's using it (`sudo lsof /mnt/sd1gb`) and stop it first.

After this procedure formats the card and the verify step mounts it, the card stays MOUNTED until you `umount` explicitly. Don't pull without unmounting.

See Step 9.

Scope. Operational procedure for wiping + formatting SD cards (and microSD cards) for use with Tang Primer 25K's TF Card slot (canonical Phase-1 student baseline per A11 + D187), Tang Nano 20K's TF Card slot (advanced-track alternative), and other embedded targets in the Virtus Academy curriculum. Linux host. Sudo required.

First captured. 2026-05-01 firing #62 (session `virtus-academy-20260501-0728`) responding to Jon's directive at ~07:30 ET to wipe + format a 1 GB SD card for Tang Nano testing and document the procedure. Banner added firing #67 after card was physically removed while still mounted (no observed corruption (card was empty + just-formatted) but the gap was real).

When to use FAT32 vs other filesystems

Filesystem	When	Notes
FAT32	**Default for embedded. Use this.**	Universal embedded-firmware support; max file 4 GiB (rarely a constraint for FPGA / microcontroller use); single MBR partition spanning whole disk.
FAT16	Legacy / cards <2 GiB only	Some old SD-card readers / firmware libraries default to FAT16; 2 GiB cap; avoid unless needed.
exFAT	Cards >32 GiB OR file sizes >4 GiB	Tang stock examples (Primer 25K + Nano 20K) don't need this; some SDIO firmware libraries don't support exFAT.
ext4	NEVER for cards used in embedded targets	Embedded firmware almost never has ext4 driver. Reserve for Linux-host-only cards (e.g. Ultra96 boot media).

Tang Primer 25K + Tang Nano 20K rule of thumb: FAT32 + MBR + single primary partition spanning whole disk. Volume label optional (12-char limit on MBR-FAT).

When to use MBR vs GPT partition table

Table	When
MBR (msdos)	**Default for embedded. Use this.** Cards ≤ 2 TB. Universal firmware support.
GPT	Cards > 2 TB only. Most embedded firmware will NOT boot/read GPT.

Linux host procedure (canonical)

Tested on Ubuntu 22.04 (redbook). Adapt sudo prefix as needed; the canonical machine in this project is **redbook** (passwordless sudo per `feedback_unblock_via_docker_or_alt_host.md`).

Step 0: Identify the device

```
lsblk -f
```

Look for the SD card by **size** (e.g. `968M`, `7.4G`, etc.) and absence of important data. **Triple-check you're not about to wipe `/dev/sda` (your boot drive).** A USB SD-card reader typically shows up as `/dev/sdb`, `/dev/sdc`, etc. Note the device path (e.g. `/dev/sdb`) and the existing partition path (`/dev/sdb1`).

Step 1: Unmount any existing partition

```
mount | grep sdb        # check whether anything is mounted
sudo umount /dev/sdb1   # or whatever partition shows mounted
```

If `not mounted`, skip to Step 2.

Step 2: Wipe partition table + filesystem signatures

```
sudo wipefs -a /dev/sdb
```

This removes existing filesystem-magic bytes so the kernel + tools see a fresh device.

Step 3: Create fresh MBR partition table

```
sudo parted -s /dev/sdb mklabel msdos
```

The `msdos` label = MBR. The `-s` flag = scripted (no prompts).

Step 4: Create single FAT32-typed primary partition

```
sudo parted -s /dev/sdb mkpart primary fat32 1MiB 100%
```

`1MiB` start aligns the partition to the standard MBR boundary (most cards' erase blocks align here). `100%` extends to end of disk.

Step 5: Refresh kernel partition table

```
sudo partprobe /dev/sdb
sleep 1
```

Forces the kernel to re-read the partition table so `/dev/sdb1` becomes valid.

Step 6: Install dosfstools (one-time on host if missing)

```
which mkfs.fat || sudo apt-get install -y dosfstools
```

`mkfs.fat` (alias `mkfs.vfat`) lives in the `dosfstools` package. Most desktop Linux installs lack this by default; install once.

Step 7: Format as FAT32 with optional label

```
sudo mkfs.fat -F 32 -n VIRTUS /dev/sdb1
```

- `-F 32` = force FAT32 (default may pick FAT16 on small cards).
- `-n LABEL` = volume label, max **11 characters**, uppercase ASCII only. Optional.
  - This project's convention: `VIRTUS` for cards prepped for Virtus Academy use.
  - `BOOT` for cards holding bootloader images.
  - `DATA` for cards holding lab data.

Step 8: Verify

```
lsblk -f /dev/sdb
sudo blkid /dev/sdb1
sudo mount /dev/sdb1 /mnt/sd1gb   # or any mount point
df -h /mnt/sd1gb
ls -la /mnt/sd1gb/                # should be empty (no '.fseventsd' / 'System Volume Information' residue)
```

Expected output of `lsblk -f`:

```
NAME   FSTYPE FSVER LABEL  UUID         FSAVAIL FSUSE% MOUNTPOINTS
sdb
└─sdb1 vfat   FAT32 VIRTUS XXXX-XXXX     XXXM    0%   /mnt/sd1gb
```

Expected output of `blkid`:

```
/dev/sdb1: LABEL_FATBOOT="VIRTUS" LABEL="VIRTUS" UUID="..." BLOCK_SIZE="512"
TYPE="vfat" PARTUUID="..."
```

Step 9: Eject for physical removal

```
sudo umount /mnt/sd1gb
sudo eject /dev/sdb              # spins down + powers off if reader supports it
```

Card now safe to physically remove and insert into Tang Nano TF slot.

Worked example, 1 GB SD card prep on redbook (2026-05-01 firing #62)

```
$ ssh redbook
$ lsblk -f
[…] sdb1 vfat FAT16 968M 0% /mnt/sd1gb          # FAT16, has Mac+Win residue
$ sudo umount /mnt/sd1gb
$ sudo wipefs -a /dev/sdb
/dev/sdb: 2 bytes were erased at offset 0x000001fe (dos): 55 aa
$ sudo parted -s /dev/sdb mklabel msdos
$ sudo parted -s /dev/sdb mkpart primary fat32 1MiB 100%
$ sudo partprobe /dev/sdb
$ sudo apt-get install -y dosfstools
$ sudo mkfs.fat -F 32 -n VIRTUS /dev/sdb1
mkfs.fat 4.2 (2021-01-31)
$ sudo mount /dev/sdb1 /mnt/sd1gb
$ df -h /mnt/sd1gb
/dev/sdb1       966M  4.0K  966M   1% /mnt/sd1gb
$ sudo blkid /dev/sdb1
/dev/sdb1: LABEL_FATBOOT="VIRTUS" LABEL="VIRTUS" UUID="016E-C7C8" BLOCK_SIZE="512"
TYPE="vfat" PARTUUID="7be7f289-01"
```

Card is now ready for insertion into Tang Primer 25K (canonical Phase-1 baseline) or Tang Nano 20K (advanced-track) TF slot.

Tang-target specifics (Primer 25K canonical Phase-1; Nano 20K advanced-track)

- **TF Card slot** = microSD on the underside of the board (per Sipeed schematic, both targets). Same FAT32 procedure applies; just use a microSD reader instead of a full-size SD reader.
- **Sipeed canonical examples** that use the TF slot are typically HDMI image-display demos. They read raw RGB bytes or `.bin` files from FAT32. See `github.com/sipeed/TangPrimer-25K-example/` and `github.com/sipeed/TangNano-20K-example/hdmi/` for examples.
- **SDIO bus** on both Tang targets talks to the slot; HDL can use Sipeed's stock `tf_card.v` controller (works on both targets) or roll a SPI-mode SD driver against the same pins. SPI mode is slower but simpler to teach (CSA-101 / CSA-201 prefer SPI mode for the lab arc that builds an SD driver from first principles).

Pedagogical anchor

This procedure is canonical for the **CSA-101 / CSA-201 / VCA-HW-101** lab archaeology when students prep their own SD/TF cards. Three teachable units:

1. **MBR vs GPT**. Students see why MBR is the right default for embedded.
2. **FAT32 vs FAT16 vs exFAT vs ext4**. Students learn the trade-offs and why FAT32 dominates embedded.
3. **`wipefs` + `parted` + `mkfs.fat` toolchain**. Students add these to their Toolchain Diary alongside `openFPGALoader` / `yosys` / `nextpnr` / `gowin_pack`.

Architecture Comparison Sidebar candidate: **SD card filesystem support across boot environments** (Tang-family bare-metal HDL / Ultra96 PYNQ Linux / Raspberry Pi Linux / x86 Windows / Apple macOS). Students see the same FAT32 partition table interpreted differently by each.

Cross-references

- **Sipeed wiki:** `https://wiki.sipeed.com/hardware/en/tang/tang-nano-20k/nano-20k.html` (TF Card slot listed in spec table).
- **Sipeed canonical TF examples:** `github.com/sipeed/TangNano-20K-example/hdmi/`.
- **Memory:** `feedback_unblock_via_docker_or_alt_host.md` (use redbook for sudo-blocked operations).
- **Cross-chapter handout neighbors:** `cross-chapter-sim-toolchain-quick-ref.md` (sim toolchain), `cross-chapter-synth-error-quick-ref.md` (synthesis error catalog).

Captured 2026-05-01 firing #62 by virtus-academy-20260501-0728. Worked example tested on redbook 2026-05-01 ~07:35 ET. Volume `VIRTUS` UUID `016E-C7C8` ready for Tang Nano TF-slot insertion.
