SDPrep

PicoCalc SD/USB Formatter

Author: Dr. Eric Oliver Flores Toro
Version: 1.0
Date: November 2025
Platform: Pop!_OS 22.04 LTS (GNOME / Wayland)
Language: C / GTK 4 / JSON-GLib
License: MIT-style (open technical use)

Abstract

SDPrep is a professional, GUI-driven storage preparation utility designed to format and partition SD cards and USB storage for use with the **Clockwork Pi PicoCalc** and other embedded systems. Unlike generic formatters, SDPrep enforces device-type awareness, prevents system-disk damage, and produces a **precisely defined two-partition layout** compatible with firmware update and dual-image workflows. The tool merges Linux block-device analytics with a simple GTK 4 interface that uses a single click to perform the full operation, while maintaining absolute safety boundaries.

Design Philosophy

1. **Human-Safe Automation** – A single click should automate complex partitioning without risking user data.
2. **Hardware Awareness** – Only *removable* devices (USB or MMC) are eligible. SATA/NVMe drives and system roots are excluded by design.
3. **Transparency** – Every command is visible in a live log pane; the user always knows what SDPrep is executing.
4. **Cross-Platform Consistency** – Results are byte-identical to the official `partition_usb_32mb.sh` from Clockwork Pi's PicoCalc repository but implemented as a robust C application with GUI feedback.
5. **Recoverability** – No hidden writes, no low-level image overwrites. The FAT32 volume remains readable by any host OS.

Functional Overview

Stage	Operation	Tools Used	Outcome
1	Device Enumeration	`lsblk -J -O` + JSON-GLib	Detect all block devices, filter removable USB/MMC
2	Root-Disk Exclusion	`df /` → `lsblk -no PKNAME`	Prevents system drive selection
3	Unmount &	`sync`, `umount`	Ensures clean state

Stage	Operation	Tools Used	Outcome
	Sync		
4	Signature Wipe	`wipefs -a`	Removes old FS and partition data
5	Table Creation	`parted -s mklabel msdos`	New MBR partition table
6	Partitioning	`parted -s mkpart primary fat32 1MiB <end-32MiB>`; then second 32 MiB tail	Two partitions: main + reserved
7	FAT32 Formatting	`mkfs.fat -F32 -I -n <Label>`	Creates main volume
8	Kernel Sync	`partprobe`, `udevadm settle`	Updates /dev nodes
9	Verification	`fdisk -l`, `lsblk -fo`	Final layout confirmation

Resulting layout:

```
/dev/sdX1  → FAT32  (User data / firmware)
/dev/sdX2  → 32 MiB unformatted tail (reserved for FUZIX or boot image)
```

GUI Architecture

Layout

- **Top Row:** Device selector (`GtkComboBoxText`) + “Refresh” button
- **Second Row:** Volume label entry (`GtkEntry`, default `PICO_DATA`)
- **Checkbox:** Safety confirmation (“I confirm this removable device …”).
- **Progress Bar:** Live pulse during operations.
- **Scrollable Log:** Real-time stdout/stderr from subprocesses.
- **Action Buttons:**
  - **Click to Format**
  - **Abort** (runs `g_subprocess_force_exit`)
  - **Quick (Close)**

Event Model

GTK 4 dropped `gtk_main()`; SDPrep uses a **GLib main loop** (`g_main_loop_run`) for true asynchronous responsiveness.
A small timer callback pulses the progress bar while the active GSubprocess runs.

Core Implementation

Device Detection

```
lsblk -J -O -b
```

is parsed with **JSON-GLib**, producing structured objects containing `kname`, `path`, `tran`, `rm`, `model`, and `size`.
The filter ensures:

```
(rm == 1) AND (tran == "usb" OR subsystems contains "mmc")
AND (device != root parent)
AND (size >= 64 MiB)
```

This eliminates internal SSDs, NVMe drives, and mounted system roots.

Safe Execution Model

All system calls use `g_subprocess_newv()`. No shell expansion is permitted; arguments are passed as true argv arrays to avoid injection.
Outputs are streamed live into the GTK TextView.

FAT Label Sanitization

```
isalnum(c) ? toupper(c) : '_'
```

ensures a valid 11-character FAT32 volume label compliant with Microsoft FAT specifications.

Abort Handling

When the user clicks **Abort**, the current subprocess is forcibly terminated using `g_subprocess_force_exit()`.
The GUI re-enables immediately and logs the interruption.

Safety and Reliability

1. **Root Privilege Gate:** Non-root users are elevated via `pkexec` before any block device access.
2. **Dynamic Blacklist:** The parent disk of `/` is queried in real time; it is never listed.
3. **Removable Validation:** Both `rm=1` and transport USB/MMC are required.
4. **Read-Only Progress:** Until the confirmation checkbox is checked, all destructive actions remain disabled.
5. **Command Echo:** Each shell command is echoed in the log, forming a complete audit trail.
6. **Abort Recovery:** No persistent process continues after user cancellation.
7. **Partition Boundary Checks:** The reserved tail size (32 MiB) is enforced; smaller media abort gracefully.

User Workflow

1. Launch **SDPrep** (`sudo ./sdprep` or via launcher).
2. Choose the removable device (e.g. `/dev/sdb — SanDisk Ultra USB • Removable`).

3. Enter a volume label (default `PICO_DATA` or `MICROPYTHON`).
4. Tick the confirmation box.
5. Click **“Click to Format.”**
6. Watch progress and logs update in real time.
7. Upon completion, the GUI displays “Completed ✔️” and shows the final `fdisk` and `lsblk` output.

Optional post-step:

```
sudo dd if=fuzix.img of=/dev/sdX2 bs=4M status=progress
sync
```

to flash a secondary system image.

Build and Integration

Dependencies

```
libgtk-4-dev
libjson-glib-dev
build-essential
pkg-config
parted
dosfstools
util-linux
udev
```

Compile

```
gcc -O2 -Wall -Wextra -o sdprep sdprep.c \
    $(pkg-config --cflags --libs gtk4 json-glib-1.0)
```

Optional Install

```
sudo install -m755 sdprep /usr/local/bin/
```

Add `sdprep.desktop` under `~/.local/share/applications/` for Pop!_OS launcher integration.

Validation Example

A correct PicoCalc-ready SD card should show:

```
Disk /dev/sdb: 57.95 GiB
/dev/sdb1  57.9 G  W95 FAT32 (LBA)  Label=MICROPYTHON
/dev/sdb2  32 M   Linux (reserved)
```

This layout is byte-identical to the reference `partition_usb_32mb.sh` output published by Clockwork Pi.

Future Enhancements

Area	Planned Feature
FUZIX Deployment	“Write Image to Tail” button with file-picker
Auto Mount	Mount FAT32 after format for file copy tasks
Progress Refinement	Visual percentage based on `mkfs.fat` output
Log Export	Save session log to `~/Documents/sdprep_log.txt`
Theming	Dark/light GTK mode toggle
Firmware Integration	Hook SDPrep into PicoCalc firmware manager

Conclusion

SDPrep turns a potentially destructive low-level procedure into a deterministic, auditable, and user-friendly process. By merging strong metrology-style safety constraints with a responsive modern GTK interface, it provides both engineers and everyday users a guaranteed-safe way to create PicoCalc-compatible storage media. The resulting design serves as a reusable blueprint for any **embedded system formatter** where correctness, transparency, and human safety outweigh raw speed.