dd Raw block copier

START HERE

lsblk
Find the right device

dd if=/dev/zero of=*<file>* bs=1M count=100
Make a 100 MiB file

dd if=*<in>* of=*<out>* bs=4M status=progress
Copy with progress

dd if=*<in>* of=*<out>* count=1
Copy one block only

dd if=*<iso>* of=/dev/*<usb>* bs=4M conv=fsync
Write an image to USB

kill -USR1 $(pidof dd)
Show progress now

dd if=/dev/*<disk>* of=*<img>* bs=1M
Disk to image file

sync
Flush writes to disk

if= input file or device → **read ibs** → **dd** conv= converts → **write obs** → **of=** output file or device

CONCEPTS

Danger
dd overwrites whatever of= points at, with no prompt. A wrong device letter destroys that disk. Check lsblk twice before pressing Enter.

Operands are NAME=VALUE with no dashes and no spaces around the equals sign. Without if= it reads stdin, without of= it writes stdout.

dd copies raw bytes and knows nothing about filesystems. It is just a block copier, which is why it fits images and devices.

BASICS

dd if=*<in>* of=*<out>*
Copy a file or device

bs=*<size>*
Block size for read and write

count=*<n>*
Copy only n input blocks

skip=*<n>*
Skip n blocks of input first

seek=*<n>*
Skip n blocks of output first

status=progress
Show running transfer stats

status=none
Print only errors

bs=4M
Good default for disks and USB

ibs=*<size>* obs=*<size>*
Separate read and write sizes

count=*<n>*B
Count bytes instead of blocks

skip=*<n>*B
Skip bytes instead of blocks

kill -USR1 $(pidof dd)
Print stats once, keep copying

watch -n1 kill -USR1 $(pidof dd)
Poll stats on a dd already running

Size suffixes: K=1024, M=1024*1024, G=1024^3. kB, MB, GB are powers of 1000. Plain bs=1M means MiB.

WRITING AND FLUSHING

conv=fsync
Flush data and metadata at the end

WRITING AND FLUSHING (cont.)

conv=fdatasync
Flush data only at the end

oflag=sync
Synchronous writes, slow and safe

oflag=direct
Bypass the page cache on write

iflag=direct
Bypass the page cache on read

conv=notrunc
Do not truncate the output file

conv=excl
Fail if the output exists

conv=nocreat
Fail if the output is missing

oflag=append conv=notrunc
Append to the output

iflag=fullblock
Fill whole blocks when reading pipes

iflag=nocache oflag=nocache
Drop cached pages after use

Without conv=fsync dd can report done while data still sits in the cache. Wait for the prompt, then run sync before unplugging a USB stick.

CONVERSIONS AND ERRORS

conv=noerror,sync
Continue on read errors, pad with zeros

conv=sparse
Skip all-zero blocks, make a sparse file

noerror alone shifts data after a bad block. Always pair it with sync so offsets stay aligned. For failing disks prefer ddrescue.

DISKS AND IMAGES

lsblk -o NAME,SIZE,MODEL,TRAN,MOUNTPOINT
Identify the target device

dd if=*<iso>* of=/dev/*<usb>* bs=4M conv=fsync status=progress
Write an ISO to a USB stick

dd if=/dev/*<disk>* of=*<disk.img>* bs=4M status=progress
Back up a whole disk

DISKS AND IMAGES (cont.)

dd if=*<disk.img>* of=/dev/*<disk>* bs=4M conv=fsync
Restore a disk image

dd if=/dev/*<disk>* bs=4M | gzip > *<disk.img.gz>*
Compressed disk image

gunzip -c *<disk.img.gz>* | dd of=/dev/*<disk>* bs=4M
Restore compressed image

dd if=/dev/*<disk>* of=*<mbr.bin>* bs=512 count=1
Save the first sector (MBR)

dd if=*<mbr.bin>* of=/dev/*<disk>* bs=446 count=1
Restore boot code, keep partitions

dd if=/dev/*<part>* of=*<part.img>* bs=4M
Image a single partition

dd if=/dev/*<disk>* bs=4M | ssh *<host>* 'dd of=*<disk.img>*'
Image a disk over SSH

dd if=*<disk.img>* bs=4M | sha256sum
Hash the image to verify

cmp *<iso>* /dev/*<usb>* -n $(stat -c %s *<iso>*)
Verify what was written

Unmount the target first (umount /dev/<usb>*). Use the whole device (/dev/sdb), not a partition (/dev/sdb1), for bootable ISOs.

WIPING AND TEST FILES

dd if=/dev/zero of=*<file>* bs=1M count=*<n>*
File of n MiB zeros

dd if=/dev/zero of=*<file>* bs=1 count=0 seek=*<size>*
Sparse file, no data written

dd if=/dev/urandom of=*<file>* bs=1M count=*<n>*
File of random bytes

dd if=/dev/zero of=/dev/*<disk>* bs=4M status=progress
Zero a whole disk

dd if=/dev/urandom of=/dev/*<disk>* bs=4M status=progress
Overwrite with random data

dd if=/dev/zero of=/dev/*<disk>* bs=1M count=10
Wipe start of disk (old signatures)

WIPING AND TEST FILES (cont.)

truncate -s *<size>* *<file>*
Resize or create a sparse file

SSDs remap blocks, so overwriting is not a secure erase. Use blkdiscard or the drive's secure erase instead.

BENCHMARKING AND EXTRACTION

dd if=/dev/zero of=*<file>* bs=1G count=1 oflag=direct
Rough write speed

dd if=*<file>* of=/dev/null bs=1M
Rough read speed

echo 3 | sudo tee /proc/sys/vm/drop_caches
Drop caches before a read test

dd if=*<file>* of=/dev/null bs=1M iflag=direct
Read speed without the cache

dd if=*<in>* of=*<out>* bs=1 skip=*<off>* count=*<len>*
Cut a byte range from a file

dd if=*<in>* of=*<out>* bs=1 seek=*<off>* conv=notrunc
Patch bytes at an offset

Tests with /dev/zero can mislead on compressing or deduplicating storage. Use fio for serious benchmarks.

RECIPES

Make a bootable USB stick

```
lsblk -o NAME,SIZE,MODEL,TRAN
sudo umount /dev/sdX*
sudo dd if=distro.iso of=/dev/sdX bs=4M conv=fsync status=progress
sync
```

Back up and verify a disk

```
sudo dd if=/dev/sdX bs=4M status=progress | gzip > disk.img.gz
# restore later
gunzip -c disk.img.gz | sudo dd of=/dev/sdY bs=4M conv=fsync
```

Clone a failing disk as far as possible

```
sudo dd if=/dev/sdX of=rescue.img bs=64K conv=noerror,sync status=progress
# smaller bs loses less per bad block
```

Back up and restore the MBR

```
sudo dd if=/dev/sdX of=mbr.bin bs=512 count=1
sudo dd if=mbr.bin of=/dev/sdX bs=512 count=1
```