dd

Copie de fichier et conversions

OPTIONS

if=FILE lit FILE au lieu de l'entrée standard

of=FILE Écrit dans FILE au lieu de la sortie standard. par défaut tronque aux octets '0'

ibs=BYTES Définis la taille de block en entrée (défaut 512 octets)

obs=BYTES Définis la taille de block en sortie (défaut 512 octets)

bs=BYTES Définit la taille de block en entrée et en sortie. ça force dd à lire et écrire par block.

cbs=BYTES définit la taille de block de conversion. En convertissant des enregistrements de longueur variable à des longueurs fixes (conv=block) ou inversé (conv=unblock), utilise BYTES comme longueur d'enregistrement.

skip=BLOCKS saute BLOCKS dans le fichier d'entée avant de copier

seek=BLOCKS saute BLOCKS dans le fichier de sortie avant de copier

count=BLOCKS copie BLOCKS dans le fichier d'entrée.

status=noxfer n'affiche pas le taux de transfert et les statistiques de volume quand dd se termine.

conv=CONVERSION[,CONVERSION]... Convertit le fichier comme spécifié par CONVERSION :

ascii Convertit EBCDIC vers ASCII.

ebcdic Convertit ASCII vers EBCDIC

ibm Convertit ASCII vers EBCDIC alternatif

block pour chaque ligne dans l'entrée, sort cbs octets, remplaçant un newline par un espace et en complétant avec des espaces si nécessaire.

unblock remplace les espaces à la fin dans chaque block d'entrée de taille cbs par un newline.

lcase Change les lettres majuscules en minuscules

ucase Change les lettres minuscules en majuscules

swab inverse chaque pair d'octets en entrée.

noerror Continue même en cas d'erreurs

nocreat ne crée pas de fichier de sorite, il doit déjà exister.

excl échoue si le fichier de sorite existe déjà.

notrunc Ne tronque pas le fichier de sortie

sync complète chaque block d'entrée à la taille de ibs, en complétant avec des '0', ou des espace si block ou unblock est spécifié.

fdatasync Synchronise les données en sortie juste avant de finir. Cela force une écrite physique.

iflag=FLAG[,FLAG]... Accède au fichier d'entrée en utilisant les flags spécifiés.

oflag=FLAG[,FLAG]... Accède au fichier de sortie en utilisant les flags spécifiés.

append Écrit en mode ajout.

cio Utilise le mode I/O courant pour les données. effectue effectue un I/O direct, outrepassant les requis POSIX.

direct Utilise I/O direct pour les données, annulant le buffer cache.

directory Échoue à moins que le fichier soit un répertoire. bcp d'OS ne permettent pas I/O sur un répertoire.

dsync Utilise I/O synchronisé pour les données. Pour le ficheir de sortie, çà force l'écriture physique pour chaque écriture.

sync Utilise I/O synchronisé pour les données et les métadonnées.

nonblock Utilise I/O non-blocking

noatime Ne met pas à jours le atime des fichiers.

nofollow ne suit pas les liens symboliques

nolinks échoue si le fichier à de multiples liens dures.

binary Utilise I/O binaire, pour les plate-formes distinguant les fichier binaire des fichiers texte.

text Utilise I/O text.

fullblock accumule les blocks complet en entrée.

BYTES et **BLOCKS** peuvent s'exprimer avec un multiplicateur : **b** vaut 512, **c** vaut 1, **w** vaut 2, ou n'importe quel autre multiplicateur.

Envoyer un signal **INFO** (ou **USR1** pour les système ne supportant pas INFO) à un processus **dd** en cour le force à afficher des statisques d'I/O sur l'erreur standard, puis reprend la copie.

Exemples

copie en block de 512 octets entre un disque et une cassette, mais saute le label de 4 KiB au début du disque :
disk=/dev/rdsk/c0t1d0s2
tape=/dev/rmt/0
(dd bs=4k skip=1 count=0 && dd bs=512k) <$disk >$tape
(dd bs=4k seek=1 count=0 && dd bs=512k) <$tape >$disk
Copier une partition de disque dur sur un autre disque dur
dd if=/dev/sda2 of=/dev/sdb2 bs=4096 conv=notrunc,noerror
Cloner un disque dur en entier
dd if=/dev/sda of=/dev/sdb conv=notrunc,noerror
Copier un grand disque sur un autre disque plus petit. La seule différence entre une grande partition et une petite partition, hormis la taille, est la table de partition. Si vous copiez sda vers sdb, un disque entier avec une seule partition, sdb étant plus petit que sda, alors vous devez faire :
dd if=/dev/sda skip=2 of=/dev/sdb seek=2 bs=4k conv=noerror
Réaliser l'image ISO d'un CD
dd if=/dev/hdc of=/home/sam/moncd.iso bs=2048 conv=notrunc
Créer une clé USB bootable
dd if=/home/uubu/insert.iso of=/dev/sdb ibs=4b obs=1b conv=notrunc,noerror
Copier seulement le MBR d'un disque dur
dd if=/dev/sda of=/home/sam/MBR.image bs=446 count=1
Ecrire par dessus toute la place libre d'une partition
dd if=/dev/urandom > fichieroccupanttoutlespacelibre

Astuces

Pour voir la mémoire vive
dd if=/proc/kcore | hexdump -C | less
Quels systèmes de fichiers sont installés
dd if=/proc/filesystems | hexdump -C | less
Tous les modules chargés
dd if=/proc/kallsyms | hexdump -C | less
Table des interruptions
dd if=/proc/interrupts | hexdump -C | less
Depuis combien de temps fonctionne le système
dd if=/proc/uptime | hexdump -C | less
Partitions et tailles en Ko
dd if=/proc/partitions | hexdump -C | less
Etat de la mémoire
dd if=/proc/meminfo | hexdump -C | less

Créer un disque de sauvegarde (dcfldd est un programme qui ajoute aux fonctionnalités de dd l'affichage d'une barre de progression)
dcfldd if=/dev/sda of=/dev/sdb bs=4096 conv=notrunc,noerror
pour réaliser un backup de root puis :
dd if=/home/sam/root.img of=/dev/sda2 (root) bs=4096 conv=notrunc,noerror
Créer un lecteur virtuel
dd if=/dev/zero of=/dev/ram7 bs=1k count=16384
puis :
mkfs.ext3 -m0 /dev/ram7 4096
Copier la mémoire RAM dans un fichier
dd if=/dev/mem of=/home/sam/mem.bin bs=1024
Lire le BIOS
dd if=/dev/mem bs=1k skip=768 count=256 2>/dev/null | strings -n 8

Utilitaires dérivés

ddrescue permet de récupérer des secteurs défectueux

sdd Utile quand la taille des blocs d'entrée est différente de celle des blocs de sortie, et réussira dans des cas où dd échoue