

# Multiboot USB drive

From ArchWiki

A multiboot USB flash drive allows booting multiple ISO files from a single device. The ISO files can be copied to the device and booted directly without unpacking them first. There are multiple methods available, but they may not work for all ISO images.

## Related articles

---

[GRUB](#)

[Syslinux](#)

[Archiso](#)

## Contents

- 1 Using GRUB and loopback devices
  - 1.1 Preparation
  - 1.2 Installing GRUB
    - 1.2.1 Simple installation
    - 1.2.2 Hybrid UEFI GPT + BIOS GPT/MBR boot
  - 1.3 Configuring GRUB
    - 1.3.1 Using a template
    - 1.3.2 Manual configuration
  - 1.4 Boot entries
    - 1.4.1 Arch Linux monthly release
    - 1.4.2 archboot
  - 1.5 Boot entries for other distributions
    - 1.5.1 Alpine Linux
    - 1.5.2 CentOS
      - 1.5.2.1 Stock installation medium
      - 1.5.2.2 Desktop live medium
    - 1.5.3 Clonezilla Live
    - 1.5.4 Debian
    - 1.5.5 Elementary OS
    - 1.5.6 Fedora
      - 1.5.6.1 Stock installation medium
      - 1.5.6.2 Workstation live medium
    - 1.5.7 Gentoo
    - 1.5.8 GParted Live
    - 1.5.9 Kali Linux
    - 1.5.10 Knoppix
    - 1.5.11 Linux Mint
    - 1.5.12 openSUSE
      - 1.5.12.1 Stock installation medium

- 1.5.12.2 Desktop Live medium
  - 1.5.13 Parabola GNU/Linux-libre
  - 1.5.14 Sabayon
  - 1.5.15 Slackware Linux
  - 1.5.16 SystemRescueCD
  - 1.5.17 Slitaz
  - 1.5.18 Slax
  - 1.5.19 Spinrite
  - 1.5.20 Tails
  - 1.5.21 Ubuntu
  - 1.5.22 Xubuntu (32 bit)
- 2 Chainloading Windows
- 3 Using Syslinux and memdisk
  - 3.1 Preparation
  - 3.2 Install the memdisk module
  - 3.3 Configuration
  - 3.4 Caveat for 32-bit systems
- 4 See also

## Using GRUB and loopback devices

Advantages:

- only a single partition required
- all ISO files are found in one directory
- adding and removing ISO files is simple

Disadvantages:

- not all ISO images are compatible
- the original boot menu for the ISO file is not shown
- it can be difficult to find a working boot entry

### Preparation

Create at least one partition and a filesystem supported by GRUB on the USB drive. See Partitioning and File systems#Create a file system. Choose the size based on the total size of the ISO files that you want to store on the drive, and plan for extra space for the bootloader.

### Installing GRUB

#### Simple installation

Mount the filesystem located on the USB drive:

```
# mount /dev/sdXY /mnt
```

Create the directory /boot:

```
# mkdir /mnt/boot
```

Install GRUB on the USB drive:

```
# grub-install --target=i386-pc --recheck --boot-directory=/mnt/boot /dev/sdX
```

In case you want to boot ISOs in UEFI mode, you have to install grub for the UEFI target:

```
# grub-install --target x86_64-efi --efi-directory /mnt --boot-directory=/mnt/boot --removable
```

For UEFI, the partition has to be the first one in an MBR partition table and formatted with FAT32.

### Hybrid UEFI GPT + BIOS GPT/MBR boot

This configuration is useful for creating an universal USB key, bootable everywhere. First of all you must create a GPT partition table on your device. You need at least 3 partitions:

1. A BIOS boot partition (type EF02)
2. An EFI System partition (type EF00 with a FAT32 filesystem)
3. Your data partition (use a filesystem supported by GRUB)

The BIOS boot partition must be sized 1 MB, while the EFI System partition can be at least as small as 50 MB. The data partition can take up the rest of the space of your drive.

Next you must create a hybrid MBR partition table, as setting the boot flag on the protective MBR partition might not be enough.

Hybrid MBR partition table creation example using gdisk:

```
# gdisk /dev/sdX
Command (? for help): r
Recovery/transformation command (? for help): h
Type from one to three GPT partition numbers, separated by spaces, to be added to the hybrid MBR, in sequence
Place EFI GPT (0xEE) partition first in MBR (good for GRUB)? (Y/N): N
```

```

Creating entry for GPT partition #1 (MBR partition #2)
Enter an MBR hex code (default EF):
Set the bootable flag? (Y/N): N

Creating entry for GPT partition #2 (MBR partition #3)
Enter an MBR hex code (default EF):
Set the bootable flag? (Y/N): N

Creating entry for GPT partition #3 (MBR partition #4)
Enter an MBR hex code (default 83):
Set the bootable flag? (Y/N): Y

Recovery/transformation command (? for help): x
Expert command (? for help): h
Expert command (? for help): w

Final checks complete. About to write GPT data. THIS WILL OVERWRITE EXISTING
PARTITIONS!!

Do you want to proceed? (Y/N): Y

```

You can now install GRUB to support both EFI + GPT and BIOS + GPT/MBR. The GRUB configuration (--boot-directory) can be kept in the same place.

First, you need to mount the EFI System partition and the data partition of your USB drive. Then, you can install GRUB for EFI with:

```
# grub-install --target=x86_64-efi --efi-directory=/EFI_MOUNTPOINT --boot-directory=/DATA_MOUNTPOINT/boot
```

And for BIOS with:

```
# grub-install --target=i386-pc --boot-directory=/DATA_MOUNTPOINT/boot --recheck /dev/sdX
```

As an additional fallback, you can also install GRUB on your MBR-bootable data partition:

```
# grub-install --target=i386-pc --boot-directory=/DATA_MOUNTPOINT/boot --recheck /dev/sdX
```

## Configuring GRUB

### Using a template

There are some git projects which provide some pre-existing GRUB configuration files, and a nice generic `grub.cfg` which can be used to load the other boot entries on demand, showing them only if the specified ISO files - or folders containing them - are present on the drive.

Multiboot USB: <https://github.com/aguslr/multibootusb>

GLIM (GRUB2 Live ISO Multiboot): <https://github.com/thias/glim>

## Manual configuration

For the purpose of multiboot USB drive it is easier to edit `grub.cfg` by hand instead of generating it. Alternatively, make the following changes in `/etc/grub.d/40_custom` or `/mnt/boot/grub/custom.cfg` and generate `/mnt/boot/grub/grub.cfg` using `grub-mkconfig`.

As it is recommend to use a persistent name instead of `/dev/sdxY` to identify the partition on the USB drive where the image files are located, define a variable for convenience to hold the value. If the ISO images are on the same partition as GRUB, use the following to read the UUID at boot time:

```
/mnt/boot/grub/grub.cfg
# path to the partition holding ISO images (using UUID)
probe -u $root --set=rootuuid
set imgdevpath="/dev/disk/by-uuid/$rootuuid"
```

Or specify the UUID explicitly:

```
/mnt/boot/grub/grub.cfg
# path to the partition holding ISO images (using UUID)
set imgdevpath="/dev/disk/by-uuid/UUID_value"
```

Alternatively, use the device label instead of UUID:

```
/mnt/boot/grub/grub.cfg
# path to the partition holding ISO images (using labels)
set imgdevpath="/dev/disk/by-label/label_value"
```

The necessary UUID or label can be found using `lsblk -f`. Do not use the same label as the Arch ISO for the USB device, otherwise the boot process will fail.

To complete the configuration, a boot entry for each ISO image has to be added below this header, see the next section for examples.

## Boot entries

It is assumed that the ISO images are stored in the `boot/iso/` directory on the same filesystem where GRUB is installed. Otherwise it would be necessary to prefix the path to ISO file with device identification when using the `loopback` command, for example `loopback loop (hd1,2)$isofile`. As this identification of devices is not

persistent, it is not used in the examples in this section.

One can use persistent block device naming like so. Replace the UUID according to your ISO filesystem UUID.

```
# define globally (i.e outside any menuentry)
insmod search_fs_uuid
search --no-floppy --set=isopart --fs-uuid 123-456
# later use inside each menuentry instead
loopback loop ($isopart)$isofile
```

**Tip:** For a list of kernel parameters, see <https://www.kernel.org/doc/Documentation/admin-guide/kernel-parameters.rst> and <https://www.kernel.org/doc/Documentation/admin-guide/kernel-parameters.txt> (still incomplete)

## Arch Linux monthly release

Also see [archiso](#).

```
menuentry '[loopback]archlinux-2017.04.01-x86_64.iso' {
    set isofile='/boot/iso/archlinux-2017.04.01-x86_64.iso'
    loopback loop $isofile
    linux (loop)/arch/boot/x86_64/vmlinuz archisodevice=/dev/loop0 img_dev=$imgdevpath img_loop=$isof
    initrd (loop)/arch/boot/x86_64/archiso.img
}
```

**Note:** As of [archiso v23](#) (monthly release 2015.10.01), the parameter `archisodevice=/dev/loop0` is no longer necessary when boot using GRUB and loopback devices.

## archboot

Also see [archboot](#).

```
menuentry '[loopback]archlinux-2014.11-1-archboot' {
    set isofile='/boot/iso/archlinux-2014.11-1-archboot.iso'
    loopback loop $isofile
    linux (loop)/boot/vmlinuz_x86_64 iso_loop_dev=$imgdevpath iso_loop_path=$isofile
    initrd (loop)/boot/initramfs_x86_64.img
}
```

## Boot entries for other distributions

### Alpine Linux

**Tip:** If you want to boot into a 32-bit system, replace `x86_64` with `x86`.

```
menuentry '[loopback]alpine x86_64' {
    set isofile='/boot/iso/alpine-extended-3.6.0-x86_64.iso'
    loopback loop $isofile
    set root=loop
    linux /boot/vmlinuz-hardened modloop=/boot/modloop-grsec modules=loop,squashfs,sd-mod,usb-storage
    initrd /boot/initramfs-hardened
}
```

## CentOS

### Stock installation medium

```
menuentry "[loopback]CentOS-7.0-1406-x86_64-DVD" {
    set isofile='/boot/iso/CentOS-7.0-1406-x86_64-DVD.iso'
    loopback loop $isofile
    linux (loop)/isolinux/vmlinuz noeject inst.stage2=hd:/dev/sdb2:$isofile
    initrd (loop)/isolinux/initrd.img
}
```

### Desktop live medium

```
menuentry '[loopback]CentOS-7.0-1406-x86_64-GnomeLive' {
    set isofile='/boot/iso/CentOS-7.0-1406-x86_64-GnomeLive.iso'
    loopback loop $isofile
    linux (loop)/isolinux/vmlinuz0 root=live:CDLABEL=CentOS-7-live-GNOME-x86_64 iso-scan/filename=$is
    initrd (loop)/isolinux/initrd0.img
}
```

## Clonezilla Live

**Tip:** Since 2014.01.05[1] (<https://projects.archlinux.org/archiso.git/commit/?id=5cd02c704046cdb6974f6b10f0cac366eeebec0e>), the Arch Linux monthly release contains clonezilla.

```
menuentry "[loopback]clonezilla-live-20170220-yakkety-amd64" {
    set isofile="/boot/iso/clonezilla-live-20170220-yakkety-amd64.iso"
    loopback loop $isofile
    linux (loop)/live/vmlinuz boot=live union=overlay username=user config components quiet noswap no
    initrd (loop)/live/initrd.img
}
```

## Debian

```
menuentry '[loopback]debian-live-8.8.0-amd64-gnome-desktop' {
    set isofile='/boot/iso/debian-live-8.8.0-amd64-gnome-desktop.iso'
}
```

```

loopback loop (hd1,2)$isofile
linux (loop)/live/vmlinuz boot=live config fromiso=/dev/sdb2$isofile
initrd (loop)/live/initrd.img
}

```

## Elementary OS

```

menuentry '[loopback]elementaryos-freya-amd64.20150411' {
    set isofile='/boot/iso/elementaryos-freya-amd64.20150411.iso'
    loopback loop $isofile
    linux (loop)/casper/vmlinuz boot=casper iso-scan/filename=$isofile locale=en_US.UTF-8
    initrd (loop)/casper/initrd.lz
}

```

## Fedora

### Stock installation medium

```

menuentry '[loopback]Fedora-Workstation-netinst-x86_64-24-1.2' {
    set isofile='/boot/iso/Fedora-Workstation-netinst-x86_64-24-1.2.iso'
    loopback loop $isofile
    linux (loop)/isolinux/vmlinuz inst.stage2=hd:LABEL=Fedora-WS-dvd-x86_64-24 iso-scan/filename=$isofile
    initrd (loop)/isolinux/initrd.img
}

```

### Workstation live medium

```

menuentry '[loopback]Fedora-Workstation-Live-x86_64-24-1.2' {
    set isofile='/boot/iso/Fedora-Workstation-Live-x86_64-24-1.2.iso'
    loopback loop $isofile
    linux (loop)/isolinux/vmlinuz root=live:CDLABEL=Fedora-WS-Live-24-1-2 iso-scan/filename=$isofile
    initrd (loop)/isolinux/initrd.img
}

```

## Gentoo

```

menuentry "[loopback]livedvd-amd64-multilib-20160514" {
    set isofile="/boot/iso/livedvd-amd64-multilib-20160514.iso"
    loopback loop $isofile
    linux (loop)/isolinux/gentoo root=/dev/ram0 init=/linuxrc aufs looptype=squashfs loop=/image.squashfs
    initrd (loop)/isolinux/gentoo.xz
}

```

## GParted Live

```

menuentry "[loopback]gparted-live-0.28.1-1-amd64" {
    set isofile="/boot/iso/gparted-live-0.28.1-1-amd64.iso"
}

```

```

loopback loop $isofile
linux (loop)/live/vmlinuz boot=live union=overlay username=user config components quiet noswap no
initrd (loop)/live/initrd.img
}

```

Customize your resolution and language to skip the setup questions and boot into X. This uses swap - auto-detected.

```

menuentry "[loopback]gparted-live-0.28.1-1-amd64 EN_US 1920x1200 SWAP" {
    set isofile="/boot/iso/gparted-live-0.28.1-1-amd64.iso"
    loopback loop $isofile
    linux (loop)/live/vmlinuz boot=live union=overlay username=user config components quiet swapon no
    initrd (loop)/live/initrd.img
}

```

## Kali Linux

```

menuentry "[loopback]kali-linux-1.0.7-amd64" {
    set isofile='/boot/iso/kali-linux-1.0.7-amd64.iso'
    loopback loop $isofile
    linux (loop)/live/vmlinuz boot=live findiso=$isofile noconfig=sudo username=root hostname=kali
    initrd (loop)/live/initrd.img
}

```

## Knoppix

```

menuentry "[loopback]KNOPPIX_V7.4.2DVD-2014-09-28-EN" {
    set isofile="/boot/iso/KNOPPIX_V7.4.2DVD-2014-09-28-EN.iso"
    loopback loop $isofile
    linux (loop)/boot/isolinux/linux bootfrom=/dev/sda2$isofile acpi=off keyboard=us language=us lang
    initrd (loop)/boot/isolinux/minirt.gz
}

```

## Linux Mint

```

menuentry "Linux Mint 17.2 Cinnamon LTS RC (x64)" {
    set iso=/boot/iso/linuxmint-17.2-cinnamon-64bit.iso
    loopback loop $iso
    linux (loop)/casper/vmlinuz boot=casper iso-scan/filename=$iso noeject noprompt
    initrd (loop)/casper/initrd.lz
}

```

## openSUSE

### Stock installation medium

```

menuentry '[loopback]openSUSE-13.1-DVD-x86_64' {
    set isofile='/boot/iso/openSUSE-13.1-DVD-x86_64.iso'
}

```

```
loopback loop $isofile
linux (loop)/boot/x86_64/loader/linux install=hd:$isofile
initrd (loop)/boot/x86_64/loader/initrd
}
```

## Desktop Live medium

```
menuentry '[loopback]openSUSE-13.1-KDE-Live-x86_64' {
    set isofile='/boot/iso/openSUSE-13.1-KDE-Live-x86_64.iso'
    loopback loop $isofile
    linux (loop)/boot/x86_64/loader/linux isofrom_device=$imgdevpath isofrom_system=$isofile LANG=en
    initrd (loop)/boot/x86_64/loader/initrd
}
```

## Parabola GNU/Linux-libre

**Tip:** If you want to boot into a 32-bit system, replace `x86_64` with `i686`.

```
menuentry '[loopback]parabola-2015.07.01-dual.iso' {
    set isofile='/boot/iso/parabola-2015.07.01-dual.iso'
    loopback loop $isofile
    linux (loop)/parabola/boot/x86_64/vmlinuz parabolaisolabel=PARA_201507 img_dev=$imgdevpath img_lo
    initrd (loop)/parabola/boot/x86_64/parabolaiso.img
}
```

**Tip:** The label string after `parabolaisolabel=` needs to be edited when a newer release is used.

## Sabayon

```
menuentry '[loopback]Sabayon_Linux_14.05_amd64_KDE' {
    set isofile='/boot/iso/Sabayon_Linux_14.05_amd64_KDE.iso'
    loopback loop $isofile
    linux (loop)/boot/sabayon root=/dev/ram0 aufs cdroot locale=en_US loop=/livecd.squashfs looptype=
    initrd (loop)/boot/sabayon.igz
}
```

## Slackware Linux

```
menuentry '[loopback]slackware64-14.1-install-dvd' {
    set isofile='/boot/iso/slackware64-14.1-install-dvd.iso'
    loopback loop $isofile
    linux (loop)/kernels/huge.s/bzImage printk.time=0
    initrd (loop)/isolinux/initrd.img
}
```

## SystemRescueCD

**Note:** Replace 64 with 32 if you want to boot into a 32-bit system.

```
menuentry '[loopback]systemrescuecd-x86-4.5.2' {
    set isofile='/boot/iso/systemrescuecd-x86-4.5.2.iso'
    loopback loop $isofile
    linux (loop)/isolinux/rescue64 isoloop=$isofile
    initrd (loop)/isolinux/initram.igz
}
```

## Slitaz

This image needs to be extracted to the directory given in `dir`.

```
menuentry 'slitaz-4.0 core' {
    set dir='/live/slitaz-4.0'
    set root=(hd0,msdos3)
    set lang='pt_BR'
    set kmap='br-abnt2'
    linux ($root)/$dir/bzImage lang=$lang kmap=$kmap rw root=/dev/null vga=normal autologin
    initrd ($root)/$dir/rootfs4.gz ($root)/$dir/rootfs3.gz ($root)/$dir/rootfs2.gz ($root)/$dir/rootfs1.gz
}
```

## Slax

This image needs to be extracted to the directory given in `dir`.

```
menuentry 'slax' {
    set dir=/live/slax
    set root=(hd0,msdos3)
    linux $dir/boot/vmlinuz from=$dir vga=normal load_ramdisk=1 prompt_ramdisk=0 printk.time=0 slax.f
    initrd $dir/boot/initrfs.img
}
```

## Spinrite

```
menuentry "Spinrite" {
    set gfxpayload=text
    set isofile="/boot/iso/spinrite.iso"
    set memdisk="/boot/iso/memdisk4.05"
    linux16 (hd1,gpt3)$memdisk iso
    initrd16 (hd1,gpt3)$isofile
}
```

## Tails

```
menuentry "[loopback]tails-i386-1.5.iso" {
```

```
set isofile='/boot/iso/tails-i386-1.5.iso'
loopback loop $isofile
linux (loop)/live/vmlinuz2 boot=live config findiso=${isofile} live-media=removable apparmor=1 securi
initrd (loop)/live/initrd2.img
```

**Warning:** Emergency memory erasure does not seem to work when booting this way.

Remove the `live-media=removable` option if the ISO file is not on removable media.

## Ubuntu

```
menuentry '[loopback]ubuntu-14.04.1-desktop-amd64' {
    set isofile='/boot/iso/ubuntu-14.04.1-desktop-amd64.iso'
    loopback loop $isofile
    linux (loop)/casper/vmlinuz.efi boot=casper iso-scan/filename=$isofile locale=en_US.UTF-8
    initrd (loop)/casper/initrd.lz
```

## Xubuntu (32 bit)

```
menuentry '[loopback]Xubuntu-16.04-desktop-i386' {
    set isofile='/boot/iso/xubuntu-16.04-desktop-i386.iso'
    loopback loop $isofile
    linux (loop)/casper/vmlinuz file=/cdrom/preseed/xubuntu.seed boot=casper iso-scan/filename=$is
    initrd (loop)/casper/initrd.lz
```

# Chainloading Windows

It can be very difficult to loopback a Windows install disc. One simple solution that allows you to install a variety of platforms from a USB drive with a single, unified partition is to start with a working, bootable Windows USB drive, and to replace its bootloader with GRUB.

Before installing GRUB, rename or move the Windows bootloader. It should be the default *.efi* executable - located at `(USB)/efi/boot/bootx64.efi` for a 64-bit system. Install GRUB in its place, and ensure that it is now the default executable.

You can then chainload the renamed Windows bootloader from GRUB, and also configure GRUB to loopback *.iso* files as described above.

```
menuentry '[chain]en_windows_8.1_professional_x64' {
    insmod chain
    chainloader /efi/boot/bootx64.efi.windows
```

# Using Syslinux and memdisk

Using the memdisk (<http://www.syslinux.org/wiki/index.php/MEMDISK>) module, the ISO image is loaded into memory, and its bootloader is loaded. Make sure that the system that will boot this USB drive has sufficient amount of memory for the image file and running operating system.

## Preparation

Make sure that the USB drive is properly partitioned and that there is a partition with file system supported by Syslinux, for example fat32 or ext4. Then install Syslinux to this partition, see Syslinux#Installation<sup>[broken link: invalid section]</sup>.

## Install the memdisk module

The memdisk module was not installed during Syslinux installation, it has to be installed manually. Mount the partition where Syslinux is installed to `/mnt/` and copy the memdisk module to the same directory where Syslinux is installed:

```
# cp /usr/lib/syslinux/bios/memdisk /mnt/boot/syslinux/
```

## Configuration

After copying the ISO files on the USB drive, edit the Syslinux configuration file and create menu entries for the ISO images. The basic entry looks like this:

```
boot/syslinux/syslinux.cfg

LABEL some_label
    LINUX memdisk
    INITRD /path/to/image.iso
    APPEND iso
```

See memdisk on Syslinux wiki (<http://www.syslinux.org/wiki/index.php/MEMDISK>) for more configuration options.

## Caveat for 32-bit systems

When booting a 32-bit system from an image larger than 128MiB, it is necessary to increase the maximum memory usage of vmalloc. This is done by adding `vmalloc=valueM` to the kernel parameters, where *value* is larger than the size of the ISO image in MiB.[2] ([http://www.syslinux.org/wiki/index.php/MEMDISK#-\\_memdiskfind\\_in\\_combination\\_with\\_phram\\_and\\_mtdblock](http://www.syslinux.org/wiki/index.php/MEMDISK#-_memdiskfind_in_combination_with_phram_and_mtdblock))

For example when booting the 32-bit system from the Arch installation ISO (<https://www.archlinux.org/download/>), press the `Tab` key over the `Boot Arch Linux (i686)` entry and add `vmalloc=768M` at the end. Skipping this step will result in the following error during boot:

```
modprobe: ERROR: could not insert 'phram': Input/output error
```

## See also

- GRUB:
  - <https://help.ubuntu.com/community/Grub2/ISOBoot/Examples>
  - <https://help.ubuntu.com/community/Grub2/ISOBoot>
- Syslinux:
  - Boot an ISO image ([http://www.syslinux.org/wiki/index.php?title=Boot\\_an\\_Iso\\_image](http://www.syslinux.org/wiki/index.php?title=Boot_an_Iso_image))

Retrieved from "[https://wiki.archlinux.org/index.php?title=Multiboot\\_USB\\_drive&oldid=479923](https://wiki.archlinux.org/index.php?title=Multiboot_USB_drive&oldid=479923)"

Categories: Boot process | Getting and installing Arch

---

- This page was last modified on 16 June 2017, at 15:11.
- Content is available under GNU Free Documentation License 1.3 or later unless otherwise noted.