

RMPrepUSB

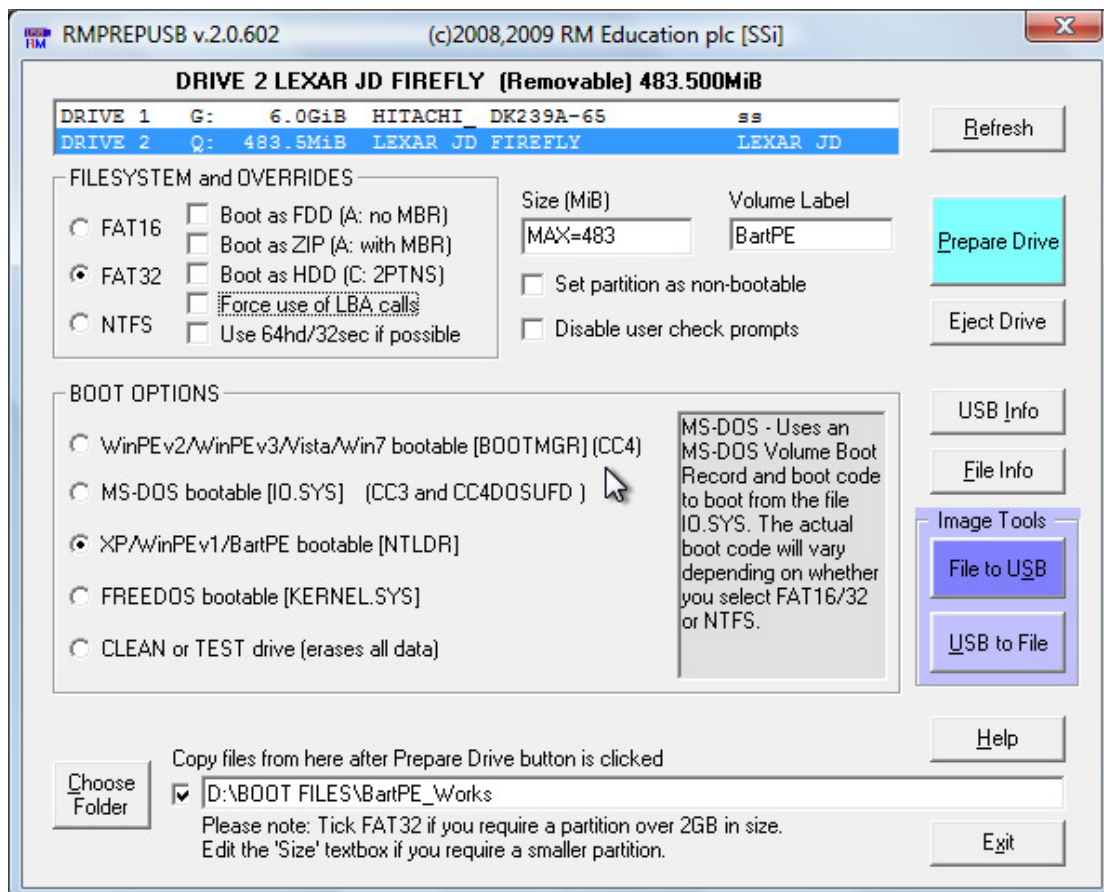
Introduction

RMPrepUSB is a Windows utility that can be used on any USB storage device, e.g. USB Flash drive (UFD) or USB hard disk. It can be used to partition, format, write a Master Boot Record, partition table and/or Volume Boot Record (sometimes called a Partition Boot Record) and operating system boot code to a USB drive.

You do not need to install it, just ensure that the four files (RMPrepUSB.exe, RMPartUSB.exe, RMPrepUSB.pdf and MSVBVM60.DLL) are present in the same folder (it is recommended you do not use a folder on the Desktop if you have a Windows Vista system). For Vista and Windows 7 systems, you should also include the two .manifest files.

RMPrepUSB and RMPartUSB are free (Freeware) for private use only. For commercial use and licensing please contact support@rm.com. Distribution, sale or use in a commercial solution is forbidden without permission from RM Education plc.

When you click on one of the Prepare Drive, Image Tool or Info buttons, RMPrepUSB will call the Windows command line utility RMPartUSB and the command line that is used will be shown to the user. If you prefer, you can use RMPartUSB in your scripts or batch files rather than use RMPrepUSB. Type RMPartUSB from a Windows command line to see full instructions on how to use the utility.



Explanation of Override options

The most complex part of RMPrepUSB is in selecting which overrides to use. The reason that there are so many choices is that different BIOSes behave in different ways. A USB flash drive that is formatted using the HDD (2PTNS) option and containing MS-DOS boot file may boot as an A: drive on one system, but boot as a C: drive on another system. If you change the BIOS Setup menu options on the same system, it might then boot as an A: drive. On yet a third system, the same USB flash drive might not boot at all.

In addition, some BIOSes will not boot from a UFD at all if the physical drive size is greater than 512MB – or – some BIOSes may not boot from a UFD if the volume size is over (say) 1.3GB.

If you wish to boot from a UFD or USB hard drive, you may need to experiment with the settings below. You may find that one group of settings will work for one system and a different group of settings will work for another system. If using UFDs, always try a 512MB USB Flash drive first before you move on to larger UFDs – some BIOSes will only work with smaller UFDs.

Boot as FDD (A: no MBR) (CC3 Build Disk)

This option will format the drive with the chosen filesystem format option (FAT16\FAT32\NTFS) but the USB drive will have no partition table. The first sector of the USB drive will typically contain the operating system boot code. This option is

usually used for an MS-DOS or FreeDOS drive that you wish to boot as a large floppy drive (i.e. they will boot to the A: prompt).

You can also select the 64hd/32sec option (recommended). If you de-select this option then the volume boot record will use 255 heads and 63 sectors per track.

Some BIOSes may not support this type of booting.

Boot as ZIP (A: with MBR) (CC3 Build Disk)

This option creates a Master Boot Record and partition table in the first sector of the drive. The Volume Boot Record code that is written to the drive will be suitable for booting MS-DOS or FreeDOS as a floppy drive. In addition, a drive geometry of 64hd\32sectors will be used (if possible).

If you wish to boot the USB drive as a hard drive then untick this option or tick the 64hd\32sec option.

‘Force use of LBA calls’ can be used with this override, but ticking the 64hd/32sec override will de-select the ZIP option.

Some BIOSes may not support this type of booting.

Boot as HDD (C: 2PTNS) (CC4 Build disk)

This option simply adds a second hidden partition entry to the partition table. Some BIOSes will treat a USB drive as a hard disk if it sees more than one partition table, because the ‘specification’ of a ZIP drive (super-floppy) is that it must only have one partition. If you want to boot a system from a USB drive as a hard disk, set this option and untick the ZIP option. You can also try either the ‘Force LBA’ override or the 64hd\32sec override with this option.

Force use of LBA calls

This option sets the Cylinder/Head/Sector values in the partition table to their maximum value of 1023 (3FE) cylinders, 255 heads (FE) and 63 sectors – even if the partition size is under 8GB. If BIOSes read a drive partition table and see that the CHS value is the maximum, then the BIOS will use a sector translation of 255 heads and 63 sectors per track. In addition, partition and volume boot record boot code will use LBA Extended Int 13h BIOS calls if it determines that the CHS values are set to the maximum. This also may help to successfully boot an operating system from a USB drive.

This option cannot be used with the 64hd\32sec option as they are mutually exclusive.

Use 64hd/32sec if possible

Instead of creating a partition table using the default drive geometry of 255 heads and 63 sectors per track, this option will use 64 heads and 32 sectors per track which is the most compatible setting for USB-ZIP (large-floppy) booting. If the partition is too large to use 64\32 geometry, then 128\32 will be used instead and then 255/32 and then 255/63.

This option is recommended for ZIP (large floppy) booting and FAT16.

Recommended settings

FreeDOS and MS-DOS boot as A: (CC3)

Size=500 Boot.as.FDD 64hd\32sec FAT16

Size=500 Boot.as.ZIP 64hd\32sec FAT16

If either of these work, try using MAX size and/or FAT32

Ensure BIOS boot option for USB is set to 'Removable Drive' and not 'Fixed Disk'

WinPE v2/v3 (CC4)

Size=MAX Boot.as.HDD.(2PTNS) ForceLBA FAT32 WinPE

BartPE, FreeDOS and MSDOS boot as C:

Size=500 Boot.as.HDD.(2PTNS) ForceLBA FAT16

Size=MAX Boot.as.HDD.(2PTNS) ForceLBA FAT32

Ensure BIOS boot option for USB is set to 'Fixed Disk' and not 'Removable Drive'

Try also various combinations of these options.

Tip for UFD booting as a hard drive:

If you have a Lexar or Netac USB Flash drive, obtain the utility BootIt.exe (try a Google search). This utility has a 'Flip Removable Bit' button – click on this and then unplug the UFD, then plug it in again – if it has worked, your UFD will be classified as a 'Fixed Disk' rather than 'Removable Media'. The utility only works with certain types of UFD controller chip. Many BIOSes will always boot as C: (hard disk) if the device identifies itself as a 'Fixed Disk' rather than 'Removable Media'.

Explanation of how BIOSes boot from a USB device

Most BIOSes require the user to enable USB booting and set the boot order so that a USB device will boot before the internal hard disk (although some BIOSes have an F10 or F12 key function which will allow you to choose a boot device from a menu). In addition you can often set the USB speed to USB 1.1 speeds (Hi speed) or USB 2.0 speeds (Full speed). If you are having difficulty booting from a USB drive, it is recommended to use a rear USB port, set the BIOS to USB 1.1 speeds and change the boot order so that the USB drive is first in the menu.

Before you enter the BIOS menu, switch off the system, insert the USB drive and then switch on the system – then press the F1 or F2 or Esc or DEL key (depending on your particular system) to enter the BIOS Setup menu.

All BIOSes behave differently, however disk storage devices are always accessed via Int 13h calls to the BIOS. There are two types of ways to access a disk device via the BIOS – 'floppy' or 'hard disk' read or writes. The BIOS must determine whether to allow access to the USB drive that it detects as either a 'floppy' (which is accessed with Int 13h DL=00h) or a 'hard disk' (which is accessed with Int 13h DL=80h). The BIOS has to determine which of these two access methods (DL=0 or DL=80h) to 'map' the USB drive to.

Once a BIOS determines how to map the USB device (either as a floppy device or a hard disk device), it will read one sector from the USB device (512 bytes) into memory at address 07C0:0000h and then jump to that code. The last two bytes of the sector must end in the two bytes '55h' and 'AAh' or it is not considered to be valid boot code. When the CPU starts to execute the code that has just been copied into memory at 07C0:0000h, the BIOS has also pre-set the DL CPU register to the correct device number (either 00h or 80h). The boot code can use this value in DL to load more sectors into memory from the USB device.

There are three 'types' of USB drive and the BIOS must try to determine which of the three different ways to access the USB drive is to be used:

1. USB-FDD (floppy disk device)

The BIOS maps the device to 'floppy' Int 13h DL=0. A USB device may be detected as a USB-FDD device if it has a device name similar to 'TEAC floppy drive' and a capacity of 1.44MB, or possibly if it has no valid partition table. The device does not need to be formatted as 1.44MB, you can have a 1GB FAT16 UFD with no MBR.

2. USB-HDD (hard disk device)

A BIOS maps the device to 'hard disk' Int 13h DL=80h. A USB device may be detected as a USB-HDD device if it has a partition table and has more than one partition entry in that partition table. If only one entry exists the BIOS could treat the device as a USB-ZIP drive or a USB-HDD drive. Sometimes a BIOS menu setting can be changed to alter this behaviour.

3. USB-ZIP (pseudo-super-floppy)

A BIOS maps the device to 'floppy' Int 13h DL=0. A USB device may be detected as a USB-ZIP device if the drive has a partition table containing only one entry. The BIOS may also require the device to be under a certain size (e.g. 512MB or less) or have values in the partition table of 64hds\32secs. In addition, when a BIOS boots from a device as a USB-ZIP device it does NOT load the first sector MBR code (LBA 0) into memory – instead the BIOS loads the Volume Boot Record code directly into memory and jumps to it by reading the single entry in the partition table to find the start position of the partition (a USB-ZIP drive can thus only contain one partition entry). The BIOS then maps the USB drive to the Int 13h DL=0 call BUT it will always add the partition start address to any access. Thus a BIOS Int 13h DL=0 call to read cylinder 0, head 0 sector 1 (LBA 0) will actually read the first sector of the partition. The real sector 1 (LBA address 0) cannot be accessed at all. For instance, say the USB-ZIP drive has a partition table with a single entry. The partition table entry indicates that the partition starts at sector 32, then the BIOS would load sector 32 into memory and start to execute the code there. Any attempt to read CHS 0\0\1 (LBA 0) via a floppy Int 13h DL=0 call (or Extended Int 13h call) from that point onwards, will actually read sector 32. If the boot code attempts to read CHS 0\1\1 (LBA 32) it will actually read the sector located at CHS 0\2\1 (LBA 64) as the BIOS will always add the partition start address on to any access. In this way, the boot code located at the start of the partition is identical to that found on a floppy disk which has no MBR or partition table.

Some BIOSes do not support this type of booting and may always boot a USB-ZIP device as a hard disk (i.e. map the USB device to respond to Int 13h DL=80h calls). The difference between a USB-ZIP device and a USB-HDD device is very small, both have a master boot record and a valid partition table, however a USB-ZIP device should contain boot code that expects to be booting from a 'floppy drive' (DL=0) whereas a USB-HDD device should contain boot code that expects to be booting from a 'hard disk drive' (DL=80h). Many modern BIOSes contain a special menu option to change this behavior.

Frequently Asked Questions

Q1 – What happens if I don't tick any of the Override options (as in the screenshot above)?

A1 – A partition table will be created using a drive geometry of 255 heads and 63 sectors per track. Only one partition table entry will be made. The volume boot record will be created with hard disk compatible parameters.

Q2 – What code is used for the Master Boot Record (MBR) ?

A2 – The code used is identical to that used by Vista. It includes BitLocker support so that if you install Vista or Windows 7 onto a USB hard disk and then enable BitLocker, it should (!!!) work. The MBR will boot both 'floppy' USB drives and 'hard disk' USB drives.

Q3 – What code is used for the Volume Boot Record?

A3 – The VBR code will vary depending on what filesystem and format options you have selected, as follows:

FAT16 + MSDOS – MSDOS5.0 identifier, looks for IO.SYS, MSDOS.SYS (or WINBOOT.SYS). Possible Error Messages: **Error!**

FAT16 + MSDOS + ZIP – MSWIN4.1 identifier, looks for IO.SYS, MSDOS.SYS (or WINBOOT.SYS). Possible Error Messages: **Invalid system disk, Disk I/O error, Replace the disk, and then press any key.**

FAT16 + FreeDOS – FRDOS4.1 which boots KERNEL.SYS (if ZIP option then floppy disk parameters are used) – Poss. Error Messages: **Error**

FAT16 + BOOTMGR – uses MSDOS5.0 identifier and boot code and hard disk parameters. Possible Error Messages: **Remove disks or other media, Disk error, Press any key to restart.**

FAT16 + NTLDR – uses MSDOS5.0 identifier and boot code with hard disk parameters. Possible Error Messages: **NTLDR is missing, Disk error, Press any key to restart.**

FAT32 + MSDOS – MSWIN4.1 identifier and boot code which looks for IO.SYS and MSDOS.SYS (or WINBOOT.SYS). If ZIP option is used then uses hard disk parameters. Poss. Error Messages: **Invalid system disk, Disk I/O error, Replace the disk and then press any key**

FAT32 + FreeDOS – FREEDOS which boots KERNEL.SYS. If ZIP option is used then uses hard disk parameters. Possible Error Messages: **Loading FreeDOS, No KERNEL.SYS**

FAT32 + NTLDR – uses MSDOS5.0 identifier and boot code and hard disk parameters. Includes 4 byte PATCH at E6-E9 changed to 90h (NOP). Possible Error Messages: **Remove disks or other media, Disk error, Press any key to restart.**

FAT32 + BOOTMGR – MSDOS5.0 identifier. Possible Error Messages:

BOOTMGR is missing, Disk error, Press any key to restart

NTFS + BOOTMGR – Vista NTFS boot code. Possible Error Messages: **A disk read error occurred, BOOTMGR is missing, BOOTMGR is compressed, Press Ctrl+Alt+Del to restart**

NTFS + NTLDR – XP boot code with 4 byte PATCH at D9-DCh changed to 90h (NOP). Possible Error Messages: **A disk read error occurred, NTLDR is missing, NTLDR is compressed, Press Ctrl+Alt+Del to restart.**

Q4 – How do I copy a Vista or Win7 OS install DVD onto a UFD and make it bootable?

A4 – Select the BOOTMGR option + HDD (2PTNs). Browse to the OS DVD and set the DVD drive letter as the file copy path. Tick the ‘Copy’ checkbox. Click Prepare Drive.

Q5 – How do I copy an XP OS install CD onto a UFD and make it bootable?

A5 – There is no simple way to do this and make it bootable. Please visit www.boot-land.net for more information.

Q6 – Why does my bootable BartPE\WinPE v2 (Vista) UFD boot on one system but not on another?

A6 – These operating systems expect be present on a ‘hard disk’ type of device which is accessed via Int 13h DL=80h. The BIOS that does not boot successfully is probably treating the UFD as a USB-FDD or USB-ZIP device and this ‘mapping’ the UFD to Int13h DL=00h and treating the device as a floppy device. It may be possible to change a setting in the BIOS setup menu (e.g. Emulate USB device as ‘Fixed Disk’) so that the UFD is treated as a hard disk. You should also try setting the HDD (2PTNs) option and the ForceLBA option. Also try changing the filesystem (e.g. use NTFS instead of FAT16). Some BIOSes will treat UFDs below a certain physical size (not partition size) as a USB-ZIP device, so try a 4GB or larger UFD. Also, try copying the boot files to the drive before the rest of the files. The best way to do this is to copy these files from a separate folder containing just the boot files to the root of the USB drive (e.g. ntldr, nt detect.com, boot.ini for XP, or IO.SYS, MSDOS.SYS and COMMAND.COM for MS-DOS or KERNEL.SYS for FreeDOS) and copy the other system files afterwards manually.

Q7 – Why does my MS-DOS bootable UFD boot as drive C: on some systems?

A7 – This will be due to the BIOS mapping the UFD as a hard drive (accessible via Int 13h DL=80h) rather than as a floppy device. Try the Boot as FDD and Boot as ZIP override options.

Q7 – I can boot from a FAT16 2GB UFD formatted as FAT16 but not from a 3GB or bigger UFD – why not?

A7 – MS-DOS cannot access FAT16 partitions over 2GB (Size=2000MB in RMPrepUSB/RMPartUSB). Set the partition size to 2000 or use FAT32.

Q8 – My PC seems to treat UFDs as a floppy device, how can I boot to WinPE v2?

A8 – There are ways to work around this. Please visit www.boot-land.net for solutions. A typical way is to use Grub or Grub4DOS as a 2nd stage boot loader which in turn will load the operating system. For instance, you can boot to a WinPE v2 iso file using Grub4DOS.

Q9 – Where can I ask a question about USB booting or ask for help?

A9 – www.boot-land.net – post specific questions about RMPrepUSB in the Project Forge – RMPrepUSB topic, for other questions choose the most appropriate forum and start a new topic. If you are an RM customer, please contact RM Support.

Q10 – Can I re-distribute RMPrepUSB or use it within my company?

A10 – You can, but only with the permission of RM Education plc (UK). This is because although RMPrepUSB and RMPartUSB are FreeWare and made freely available for private use, it must not be used for commercial purposes or sold without permission. Any re-distribution must make this clear and contain some sort of ReadMe file or other notice containing this information as well as this pdf file. Commercial enquiries should be directed to support@rm.com. RMPrepUSB and RMPartUSB is the intellectual property of RM Education plc (www.rm.com). RM may ask you to remove the software or take legal action against you if these utilities are used for commercial purposes without permission having first been obtained (and after all, it's only polite ☺). RM customers can use RMPrepUSB on or with RM products.

Q11 – Does RMPrepUSB/RMPartUSB work under WinPE v2?

A11 – Mostly! You will need to add the WMI and scripting and XML components to WinPE. Most functions will work except perhaps the folder browse button in RMPrepUSB – you will have to enter the path by typing it in manually.

Q12 – How can I use RMPrepUSB in my script/batch files?

A12 – You cannot, but you can use RMPartUSB. Type 'RMPartUSB' in a Windows command console window for help. RMPartUSB uses a command line and returns an errorlevel after completion. You can also use the LIST and FIND commands to find the drive number of a USB device. Tip: RMPrepUSB shows the exact command line that it uses to run RMPartUSB.

Q13 – Sometimes after running RMPartUSB, RMPrepUSB returns a large negative error number – why?

A13 – This has been observed to occur when using Windows 2000 or when terminating the RMPartUSB command console window before RMPartUSB has itself terminated. For instance, if you click the USBInfo button in RMPrepUSB and then terminate the shell windows that is displayed by RMPartUSB, RMPrepUSB may report this error message. The reason is unknown.

Q14 – I used RMPrepUSB to format my UFD and make it bootable but there are no files on it and it won't boot?

A14 – RMPrepUSB does not contain or copy any boot files unless you provide them. Microsoft boot files are part of the operating system and cannot be distributed. You can obtain them if you are licensed to do so, or use Freeware like FreeDOS.

Q15 – I have set the USB partition size text box to xxxMB, how do I set it back to the maximum size?

A15 – Type 'MAX' in the Size text box or click on the Refresh button.

Q16 – RMPrepUSB reports an error when it runs – why?

A16 – Under Vista you must right-click on RMPrepUSB.exe and choose 'Run as Administrator'. To ensure these applications run with Administrator rights, the .manifest files should be present in the same folder.

Q17 – The Prepare button does not seem to work under Vista – nothing happens when it should run RMPartUSB?

A17 – Try creating a folder at C:\RMPrepUSB – do not use a folder in special locations like the Desktop or My Documents as these may have special limited access rights under Vista.