Installing Linux on a Chromebook.

N.B. Before commencing I highly recommend visiting the MrChromebox.tech website https://mrchromebox.tech/#home Without the information on that site I would not have been able to compile this guide. Thanks also to “Needs to Note” whose experiences of installing Mint on a Chromebook started me off on this! These instructions relate to using UEFI Full Rom firmware as RW_LEGACY firmware is no longer supported by MrChromebox. See Notes below.

Step 1 - Create Recovery and Linux ISO Bootable USBs

1. Install the Chromebook Recovery Utility on your Chromebook if it is not already on it.

2. Follow the instructions to create a recovery USB. Once UEFI Full ROM has been installed this cannot be used but it will be needed if it is decided to revert to ChromeOS before UEFI Full ROM is installed.

3. Download the appropriate Linux distro ISO which you wish to install.

4. Write (or "burn") the ISO to a USB flash drive 8GB or larger or an SD card.

5. Verify the ISO using the instructions relevant to the distro.

Step 2 - Enable Chromebook Developer Mode

WARNING: Enabling Developer Mode will erase all user data in ChromeOS. If you have locally-stored data, back it up first. Cloud data will not be lost.

1, With device shut down, press Esc + F3 (refresh) + Power to boot into Recovery Mode. You will see the Recovery Mode boot screen, informing you that "ChromeOS is damaged or missing" although it is not.

2. Press [Ctrl+D] to enable Developer Mode, then follow confirmation steps (usually just pressing enter).

Your machine will reboot to a white screen which says "OS verification is OFF". This is Developer Mode.

3. Press [Ctrl+D] to boot ChromeOS in Developer Mode

4. Configure WiFi if necessary, and log in to Chrome. Guest account is fine.

Step 3 - Update Firmware

You can install/update your firmware from the ChromeOS terminal, or from a running Linux system. These instructions describe the process from the ChromeOS terminal. Reading the information at https://mrchromebox.tech/#fwscript is strongly recommended before updating firmware. See the Notes section below regarding RW_LEGACY and UEFI Full Rom options.

1. Press [Ctrl+Alt+T] to get a ChromeOS terminal ("crosh") window

2. At the prompt, enter **shell**

3. At the chronos@localhost / $ prompt, run the following:

    sudo crossystem dev_boot_legacy=1

4. At the chronos@localhost / $ prompt, run MrChromebox's Firmware Utility Script

    cd; curl -LO https://mrchromebox.tech/firmware-util.sh && sudo bash firmware-util.sh

5. Then follow the on-screen instructions to install your chosen firmware type. Note that some options will be grayed out if write protection has not been disabled. See Notes below.

Step 5 - Install Linux OS

1. Insert the Linux installation USB and reboot.

2. Press Esc at the UEFI (BIOS) screen and select the USB device.

3. Wait for your chosen Linux OS to boot up. Once in the live environment you can play around with the OS, or you can install it using the “Install” icon on the desktop and following the on screen instructions.

4. The Chromebook will subsequently boot straight to Linux but there are various UEFI (BIOS) options if ESC is pressed immediately the machine is switched on.

Notes

1. RW_LEGACY and UEFI Full Rom options.

The RW_LEGACY option carried no risk of bricking the Chromebook and allowed Chrome OS and Linux to be dual booted but it is no longer supported by MrChromebox and the option is grayed out. Installing UEFI Full ROM carries a small risk of bricking the Chromebook and write protection has to be disabled by removing the write protection screw from the Chromebook before updating.

Dual booting of Chrome and Linux is not possible with UEFI. The RW_LEGACY option was available the first time I updated a Chromebook and I used it as I didn't want to remove write protection or risk bricking the Chromebook. When RW_LEGACY stopped being supported I successfully used UEFI Full ROM after removing the write protection screw and it was very straightforward indeed. It is, in many ways, the better option as the Chromebook boots up much more quickly than in RW_LEGACY mode and is much more like a "regular" laptop.

Removing the write protection screw is not difficult but it does involve identifying where the screw is in the Chromebook and also removing the back of the Chromebook. MrChromebox very helpfully gives details of where to find the screw in his "Supported Devices" section. Please don't be daunted by the process as the outcome is well worth it.

2. Problems with Sound

I found that after the installation Mint 20.3 was complete the sound would not work. This problem seems to relate to Chromebooks only as it did not occur on any of the laptops or desktops I have installed Mint on. There was no problem with Mint 21.1 either but see below about video playback problem with 21.1.

On checking the "sound" menu I saw that only a dummy output was shown. There were no internal speakers showing. I solved the issue by reverting to an older kernel. To do this press Esc when the UEFI (BIOS) transitions to disk or, for legacy systems, the right shift key as soon as the Chromebook is switched on. A grub menu will be seen and "Advanced Settings" should be selected. Older kernels are listed and, working backwards, each one should be selected in turn and the computer rebooted until the one is found which fixes the sound issue. Do not use any which show "Recovery". I found that I only had to select the previous kernel and the sound worked fine. I could not work out how to make this kernel the default at boot so I simply deleted the current active kernel. I found it then booted with the older version

as default and the sound worked fine. As I say, sound was only a problem with 20.3.

3. Function Keys

After installing Mint none of the function keys were allocated. I found that they could be allocated to various functions by using “Keyboard” settings in the “All Applications” menu. Although I could assign mute, volume up/down and various others to the relevant keys there is no option for allocating brightness settings but brightness can be easily changed by simply clicking on the battery icon and using the slider.

4. Video Playback Problems

Video playback was not a problem with Mint 20.3 or Mint 21.0 but with Mint 21.1 I found that video playback froze after a couple of minutes.

The problem was solved by adding the **snd_sof.sof_debug=1** kernel parameter as below.

1. Temporarily add kernel parameter

Kernel parameters tend usually to be not needed and adding one is often only for testing purposes. In this case it's best to just add it temporarily for one boot directly from the Grub menu which is obtained on a UEFI system by tapping the Esc key at the point that the boot transitions from UEFI (BIOS) to disk or on a Legacy system holding down the right shift key at that time.

Once in the grub menu highlight the relevant Linux boot entry, use “**e”** to edit it and add the suggested kernel parameter on the "linux" line after where it normally says "quiet splash".

linux /boot/vmlinuz-5.8.0-53-generic
root=UUID=12345678-1234-5678-9abc-123456789abc ro quiet splash
snd_sof.sof_debug=1

Then press F10 to boot. Once booted you can verify from **cat /proc/cmdline** that the parameter was added correctly.. If this works ok after testing it can be made permanent as below.

2. “Permanently” add a kernel parameter

“Permanently” in the sense of not needing to add the kernel parameter manually

at each boot. Although called “permanent” it can quite easily be undone if needed. The parameter is added to **/etc/default/grub** by running the following in a terminal:

sudo nano /etc/default/grub

This will open up the Nano file editor. Add **snd_sof.sof_debug=1** after quiet splash in GRUB_CMDLINE to give the following:

GRUB_CMDLINE_LINUX_DEFAULT="quiet splash snd_sof.sof_debug=1"

Save (write) the file, run **sudo update-grub** and reboot. Once booted you can verify from **cat /proc/cmdline** that the parameter was added correctly.

5. **MX Linux Installation**

Although I had no problem at all installing Linux Mint I did have problems with MX Linux. Attempts to boot from a USB failed with a message that the USB was corrupted. I know this not to be the case as I verified the download and have used it to install MX Linux on machines other than Chromebooks. I suspect it is a problem relating to Chromebooks and I have not been able to resolve it yet.