How to Create a Multi Boot SD card or SSD out of 2 existing OSes using PINN

procount edited this page on 10 Jan, 2021 · 46 revisions

Blind Freddy edited this page to include instructions for using Raspberry Pi to make multi-boot disk with choice of Twister OS or CrowPi2 OS for use with a CrowPi2 laptop. June11, 2021

My thanks to procount (and lurch) for all their wonderful work.

NOTE: Procedures offered here for using Pi will only work with 2 partition OSes. Raspbian based seem to work fine, but others are untested. It has to do with BSD tar and GNU tar and their implementations;

How to Create a Multi-Boot SD card or SSD out of 2 existing OSes using PINN - specifically for CrowPi2, but for others with suitable modifications

Introduction

These instructions explain how to create a multi-boot SD card or SSD for the Crowpi2 that will enable you to select which OS will be used on boot. The 2 (or more) OSes are ones that you have previously installed to 2 (or more) separate SD cards and customised to an extent that you don't want to lose those customisations. They will also include a mod to /boot/config.txt on all OSes, to prevent Crowpi2 from shutting down after 30 seconds. ie add line gpio=0=op,dl to end of /boot/config.txt

These instructions can be used to multi-boot more than 2 OSes, or different OSes than the example ones given here. Just adapt the instructions as necessary.

In any of the following instructions, any line beginning with a $ indicates a line that should be typed in by the user (not including the $ symbol, which represents the command line prompt)

Pre-Requisites

- 2 existing uSD cards containing 2 OSes (e.g. Twister and Crowpi2)
- 1 blank uSD card or SSD large enough to hold both OSes as a multi-boot device. Final objective
- 1 USB memory stick (or uSD card with USB card reader) to hold install versions of required OSes
- a USB uSD card reader

- a PC running a version of desktop Linux (Ubuntu assumed). *It is possible to use a Raspberry Pi running Raspbian, but due to it's lack of memory, it is advisable to omit the '-9' option from any xz commands, or use gzip instead of xz.* I used Pi 4 8GB with -9 option and Crowpi2 took about 6 hours

Overall Process

The overall process that I will cover will have the following general outline. Each step will be documented along the way.

- Backup the existing OSes to PINN required format
- Make the backups suitable for installation by PINN. ie download and modify support files
- Copy the custom OSes to a new SD card to make new local repository.
- Make a PINN bootable uSD card or SSD
- Use PINN to install the OS backups to the multi-boot card or SSD

Step 1 - Backup existing OSes

The SD cards containing the existing customised OSes need to be backed up in a particluar way in order for PINN to install them. PINN prefers a backup file per partition, which could be a compressed image file, or a compressed tar file.

An image file is a bit-by-bit backup of the whole of a partition, including any unused space, so it can be a lot bigger than is necessary. Compressing the image afterwards can remove a lot of this unused space to make the file smaller, but it nevertheless requires an SD card at least as big as the original partition size to restore the files to. To avoid this, a common practice is to first resize the partition and its file system to be as small as possible before taking the image. In this way, less of the unused space is recorded in the image file and it can be restored to a smaller SD card. However, it also requires that after restoration, the file system should be expanded to fill the remaining space of the partition, otherwise it will not be possible to add much more data or install anymore programs to the partition. Because of these complications of first reducing and then expanding the file systems, this method will not be considered here.

An alternative approach is to create an archive of the files on each partition as a tar file and then compress it. Using this method, only the existing files will be backed up, and not any empty space, so it should be smaller than a corresponding image file. On restoration, the tar files can be decompressed to a preformatted partition of any size at least as large as the files themselves. No file system expansion would be necessary because the partition would have already been formatted to its full capacity.

This second method is documented here.

ASIDE: A word about OS partition labels and filenames

PINN uses several JSON files to describe each OS that it will install. Each OS has a partitions.json file which contains information about how many partitions there are, what their label should be, how to format them, and where the installation image for each partition can be found.

Most OSes comprise 2 partitions, a small FAT32 partition called 'boot' that holds the startup, firmware and kernel files and an EXT4 partition called 'root' that holds the root file system, (although there are exceptions). All the partition labels on a drive must be unique, so when PINN installs multiple OSes on the same device, it will make any duplicate labels unique by appending a number where necessary. Nevertheless, it is still not easy to identify which partitions belong to a particular OS when they are all called: 'boot', 'boot1', 'boot2',... 'root', 'root1', 'root2'... etc. So OSes converted for PINN often have their partition labels renamed to link them to the OS name, e.g. 'bootcp' & 'rootcp' for Crowpi2.

When installing an OS, PINN looks in partitions.json for the label name of each partition and searches for a corresponding archive file with the same basename. The extension can be any one of the following: (.tar.lzo, .tar.gz, .tar.bz2, .tar.zip, .tar.xz or .tar) according to the compression used. *Raw image files (not discussed further in this tutorial) can have any one of the following extensions: (.img.lzo, .img.gz, .img.bz2, .img.zip, .img.xz, .lzo, .gz, .bz2, .zip, .xz)* .

This tutorial assumes the partition labels are 'boot' and 'root'. However, if the partitions.json file for the OS you are converting indicates different label names from these, please substitute those for 'boot' and 'root' accordingly in the following instructions.

We are creating bootcp, rootcp for Crowpi2 and boottwstr and roottwstr for Twister

Backup Twister

First we will backup the Twister OS.

We will start on our PC and fire up Linux or your trusty Pi. (Pi4 8GB preferred) We will be doing a lot of this procedure by the command line, so if you have booted into a desktop environment, launch your normal terminal program (Terminal, LXTerminal or similar). This should drop you into your home directory (/home/<username> or just '~' for short)

1. Create a folder called `os/twister` in your home folder to store your backup in.
   $ mkdir -p ~/os/twister
2. Insert your uSD card that has your customised Twister OS on it into the USB reader and insert it into a free USB slot of the PC. You need to note what device Linux has assigned it (e.g. `/dev/sdb`) and where it has been mounted
   (e.g. `/media/<username>/boot` or `/root` etc).
   Some Linux OSes may automount the partitions on this SD card, so in this case you can type `mount` to identify the device and where it's partitions have been mounted and skip the next step. For this tutorial, we will assume the SD card is referenced as /dev/sdb with /dev/sdb1 mounted at `/media/<username>/boot` and /dev/sdb2 mounted

at `/media/<username>/root`. If yours is mounted at a different location, please replace appropriately in the following steps.

3. If it has not been auto-mounted, you will need to manually mount the partitions on the device.
   $ dmesg
   Look for the recent logs at the end of the dmesg output to identify the name of the most recently added device. You then need to create a mountpoint for each partition on the device. We will assume the device has been assigned `/dev/sdb` and has 2 partitions (`/dev/sdb1` and `/dev/sdb2`) named boot and root

```
$ sudo mkdir /media/<username>/boot
$ sudo mkdir /media/<username>/root
$ sudo mount /dev/sdb1 /media/<username>/boot
$ sudo mount /dev/sdb2 /media/<username>/root
```

4. Now we will take a copy of our SD card and store it in 2 archive files.

```
If using Linux desktop eg Ubuntu

$ cd /media/<username>/boot
$ sudo bsdtar --numeric-owner --format gnutar -cpvf ~/os/twister/boot.tar .
  (note the dot on the end)
$ cd /media/<username>/root
$ sudo find . -type s -exec rm {} \;
$ sudo bsdtar --numeric-owner --format gnutar --one-file-system -cpf ~/os/twister/root.tar .
  (note the dot on the end)

If using Raspberry Pi

$ cd /media/<username>/boot
$ sudo tar --numeric-owner -cpvf ~/os/twister/boot.tar . (note the dot on the end)
$ cd /media/<username>/root
$ sudo find . -type s -exec rm {} \;
$ sudo tar --numeric-owner --one-file-system -cpf ~/os/twister/root.tar .
  (note the dot on the end)

Then with either Linux or Pi

$ cd ~/os/twister
$ xz -9 -e boot.tar
$ xz -9 -e root.tar
note that this last process can take up to 6 hours on a Pi
```

Note the command `sudo find . -type s -exec rm {} \;` The purpose of this command is to remove any socket files on the image which cannot be backed up by bsdtar and can cause restoration problems.

You should now have 2 files in `~/os/twister` called `boot.tar.xz` and `root.tar.xz`

5. We need to make a note of how much space the OS takes up. This information is used later to modify .json file

```
$ sudo du -BK -s /media/<username>/boot | cut -d"K" -f1 >boot.size
$ sudo du -BK -s /media/<username>/root | cut -d"K" -f1 >root.size
```

```
rename boot and root files such as they have unique names for later

$mv boot.tar.xz boottwstr.tar.xz
$mv root.tar.xz roottwstr.tar.xz
```

6. Finally we will unmount the Twister OS SD card

```
$ umount /media/<username>/boot
$ umount /media/<username>/root
```

and then you may eject and remove the Twister OS SD card.

Backup Crowpi2

Now we will backup the other OS, Crowpi2, which is just a repeat of the operation to backup our Twister OS.

1. Create a folder called `os/crowpi2` in your home folder to store your backup in.
   $ mkdir -p ~/os/crowpi2
2. Insert your uSD card that has your customised Crowpi2 OS on it into the USB reader and insert it into a free USB slot of the PC. You need to note what device Linux has assigned it (e.g. `/dev/sdb`) and where it has been mounted
   (e.g. `/media/<username>/boot` or `/root` etc). Some Linux OSes may automount the partitions on this SD card, so in this case you can type `mount` to identify the device and where it's partitions have been mounted and skip the next step. For this tutorial, we will assume the SD card is referenced as /dev/sdb with /dev/sdb1 mounted at `/media/<username>/boot` and /dev/sdb2 mounted at `/media/<username>/root`. If yours is mounted at a different location, please replace appropriately in the following steps.
3. If it has not been auto-mounted, you will need to manually mount the partitions on the device.
   $ dmesg
   Look for the recent logs at the end of the dmesg output to identify the name of the most recently added device. You then need to create a mountpoint for each partition on the device. We will assume the device has been assigned `/dev/sdb` and has 2 partitions (`/dev/sdb1` and `/dev/sdb2`) named boot and root
   (we will assume the `/media/<username>/boot` and`/media/<username>/root` folders still exist from backing up Twister)

```
$ sudo mount /dev/sdb1 /media/<username>/boot
$ sudo mount /dev/sdb2 /media/<username>/root
```

4. Now we will take a copy of our SD card and store it in 2 archive files.

```
Again, for linux desktop system

$ cd /media/<username>/boot
$ sudo bsdtar --numeric-owner --format gnutar -cpvf ~/os/crowpi2/boot.tar .
$ cd /media/<username>/root
$ sudo find . -type s -exec rm {} \;
$ sudo bsdtar --numeric-owner --format gnutar --one-file-system -cpf ~/os/crowpi2/root.tar .

Or for Raspberry Pi system

$ cd /media/<username>/boot
$ sudo tar --numeric-owner -cpvf ~/os/crowpi2/boot.tar .
$ cd /media/<username>/root
$ sudo find . -type s -exec rm {} \;
$ sudo tar --numeric-owner --one-file-system -cpf ~/os/crowpi2/root.tar .


Now for either host system

$ cd ~/os/crowpi2
$ xz -9 -e boot.tar
$ xz -9 -e root.tar
```

You should now have 2 files in `~/os/crowpi2` called `boot.tar.xz` and `root.tar.xz`

```
Rename them to unique file names for Pinn installation later

mv boot.tar.xz bootcp.tar.xz
mv root.tar.xz rootcp.tar.xz
```

5. We need to make a note of how much space the OS takes up.

```
$ sudo du -BK -s /media/<username>/boot | cut -d"K" -f1 >boot.size
$ sudo du -BK -s /media/<username>/root | cut -d"K" -f1 >root.size
```

6. Finally we will unmount the Crowpi2 OS SD card

```
$ umount /media/<username>/boot
$ umount /media/<username>/root
```

and then you may eject and remove the Crowpi2 OS SD card.

Step 2 - Add OS Meta Data

Now we need to add some meta data files that describe these OSes to PINN so it knows how to install them. We shall download some existing similar meta files for these OSes and then adapt them as necessary.

First, Twister: Pinn already has an older version of Twister in their online repository. Get a copy of them from PINN - Browse /os/twister at SourceForge.net

```
$ cd ~/os/twister
$ wget -N "https://sourceforge.net/projects/pinn/files/os/twister/os.json"
$ wget -N "https://sourceforge.net/projects/pinn/files/os/twister/twister.png"
$ wget -N "https://sourceforge.net/projects/pinn/files/os/twister/partitions.json"
$ wget -N "https://sourceforge.net/projects/pinn/files/os/twister/marketing.tar"
$ wget -N "https://sourceforge.net/projects/pinn/files/os/twister/partition_setup.sh"
```

In partitions.json, we need to edit root and boot file names to match the files we have created. viz boottwstr and roottwstr.

We also need to modify partitions.json and edit the values for "partition_size_nominal" and "uncompressed_tarball_size" for each of the boot and root partitions, so 4 values in total.

First in the boot partition, replace the value of "uncompressed_tarball_size" with the value produced from:
$ expr `cat boot.size` / 1024 + 1
Whatever this value is, add 100 to it and use it to replace the value of "partition_size_nominal"

Next for the root partition, replace the value of "uncompressed_tarball_size" with the value produced from:
$ expr `cat root.size` / 1024 + 1
Whatever this value is, add 500 to it and use it to replace the value of "partition_size_nominal"
(This value of 500 is the minimum additional free space required for the OS to execute. Updating/Upgrading the OS using apt-get etc. may require additional space for it to complete successfully)

Note the spaces and punctuation of backticks in the above expressions which are not quotes or apostrophes.

Later versions of PINN support optional checksums for security. If you don't want to use these, you should delete any sha256sum lines from os.json and partitions.json, making sure to respect the JSON file syntax by possibly removing the trailing comma on the previous line. NOTE THIS WELL

If you do want to add checksums, you can checksum partition_setup.sh by issuing `sha256sum partition_setup.sh` and copying the resultant hash to the sha256sum line in os.json. Repeat this for each of the tar.xz files adn copy the resultant hash file to the appropriate sha256sum in partitions.json.

Then Crowpi2:

As Crowpi2 is based on Raspbian, we will get files suitable for modification based on Raspbian

```
$ cd ~/os/crowpi2

$wget -N "http://downloads.raspberrypi.org/raspbian/os.json"
$wget -N "http://downloads.raspberrypi.org/raspbian/Raspbian.png"
$wget -N "http://downloads.raspberrypi.org/raspbian/partitions.json"
$wget -N "http://downloads.raspberrypi.org/raspbian/marketing.tar"
$wget -N "http://downloads.raspberrypi.org/raspbian/partition_setup.sh"

Note that raspbian.png is not the correct image. It could be renamed crowpi2.png or could be
replaced with something more suitable.  Any 40 x 40 image will do.
```

As before, we need to modify partitions.json and change names to bootcp and rootcp as appropriate.

Then edit the values for "partition_size_nominal" and "uncompressed_tarball_size" for each of the boot and root partitions, so 4 values in total.

First in the boot partition, replace the value of "uncompressed_tarball_size" with the value produced from:

$ expr `cat boot.size` / 1024 + 1
Whatever this value is, add 100 to it and use it to replace the value of "partition_size_nominal"

Next for the root partition, replace the value of "uncompressed_tarball_size" with the value produced from:
$ expr `cat root.size` / 1024 + 1
Whatever this value is, add 500 to it and use it to replace the value of "partition_size_nominal". (This value of 500 is the minimum additional free space required for the OS to execute. Updating/Upgrading the OS using apt-get etc. may require additional space for it to complete successfully).

Note the spaces and punctuation of backticks in the above expressions which are not quotes or apostrophes.

See the above section relating to checksums if you want to update or remove them.

If not using checksums, remove all references in partitions.json and os.json, remembering to remove preceding comma if necessary.

Project Space

If space needs to be reserved on final disk, download all "ProjectSpace" files from sourceforge and place in folder ~/os/projectspace.

Then, 3 folders under ~/os can be installed as required, and it will be possible to replace projectspace with another OS at some later date without re-writing entire disk.

Step 3 - Copy the custom OSes to a new SD card.

Having prepared the custom OSes, they now need to be put on a USB stick, or an SD card that can be inserted into a USB SD Card reader. If any of the files in the ~/os/<osname> are greater than 4GB in size (the root.tar.xz file is likely to be the largest), then this memory device must be formatted as ext4 and you must use PINN v2.4.2i which has been adapted to support ext4 formatted installation devices. If the installation files are all smaller than 4GB, then the memory device may be formatted as FAT32 and any version of PINN can be used.

So, format your memory device to the appropriate above format and make sure it is large enough to hold all of the files in the ~/os/ folder.

Now copy the ~/os/ folder and all sub folders to your memory device. Assuming your memory device is mounted at `/media/<username>/usb` you can use the following command to copy them.
$ cp -r ~/os/ /media/<username>/usb

Step 4 - Make a PINN Bootable SD card or SSD

This step prepares the final SD card or SSD where the custom OSes will be installed for final use.

So choose one that is large enough to store all the OSes, so at least as large as the original SD cards added together.

Format your SD card or SSD as FAT32

For **Windows** users, we recommend formatting your SD card using the SD Association's Formatting Tool, which can be downloaded from https://www.sdcard.org/downloads/formatter_4/ You will need to set "FORMAT SIZE ADJUSTMENT" option to "ON" in the "Options" menu to ensure that the entire SD card volume is formatted - not just a single partition. For more detailed and beginner-friendly formatting instructions, please refer to http://www.raspberrypi.org/quick-start-guide

The SD Association's Formatting Tool is also available for **Mac** users although the default OSX Disk Utility is also capable of formatting the entire disk (select the SD card volume and choose "Erase" with "MS-DOS" format).

For **Linux** users we recommend `gparted` (or the command line version `parted`). (Update: Norman Dunbar has written up the following formatting instructions for Linux users: http://qdosmsq.dunbar-it.co.uk/blog/2013/06/NOOBS-for-raspberry-pi/)

Blind Freddy note: For large disks, formatting FAT32 creates a small problem. SD formatter will not format large disk as FAT32. It insists on Exfat. However, if SD card has previously been used for Pinn, gparted does not format correctly. So, use SD formatter in windows (or Mac), then gparted in linux to change partition to FAT32. You can also take this opportunity to make second partition Exfat for freedom of use, if spare unused space is available.

Copy the PINN files to your SD card or SSD.

- Download **pinn-lite.zip** from sourceforge
- Extract the files from this zip file onto the SD card or SSD. (Windows built-in zip features may have trouble with this file. If so, use another program such as 7zip.) Please note that in some cases it may extract the files into a folder, if this is the case then please copy across the files from inside the folder rather than the folder itself.
- Note that /boot/config.txt must have "gpio=0=op,dl" line added prior to next step, otherwise CrowPi2 will shut down after 30 seconds

Step 5 - Install the custom backups

Now that all the SD cards are prepared we can move over to the Raspberry Pi and install the OSes.

1. Insert the PINN bootable SD card from step 4 into the SD card slot of the RPi

2. Insert the memory device with your prepared custom OSes on it from step 3 into one of the USB ports of the RPi
3. Since you will be installing the custom OSes from a local device, there is no need to use the internet, and in fact PINN may choose a more recent version of the same OS from the internet instead of your custom OS, so it is best to ensure the RPi is not plugged into an ethernet cable and do not configure the wifi.
4. Boot your RPi. On first boot, PINN will reformat the SD card and when it is finished it will display your two custom OSes.
5. Select both OSes and install them to your SD card or SSD. This may take a while.
6. When installation is complete, accept the dialog box and the RPi will reboot.
7. After rebooting, you will have a boot selection dialog where you can select which of your custom OS you want to boot into.

End of file