Install Raspbian on f2fs root

this may be combined with solve raspbian SD card corruption issues with read-only mounted root partition in order to minimize SD card corruption on a raspberry pi or if your raspberry is usually shut down regularily, you might still want to consider this because f2fs was specifically optimized for flash memory storage and supposedly should help to extend the life of such storage devices significantly.

before you begin **disable automount in gnome**. i've seen it more than once in the recent past, that gnome auto-mounted a f2fs partition thinking it was ext4 and then destroyed the contents of the filesystem. so disable it while you do all this.

```
gsettings set org.gnome.desktop.media-handling automount 'false'
```

to re-enable it when you are done you can simply run

```
gsettings set org.gnome.desktop.media-handling automount 'true'
```

the script

i have created a little bash script that does it all.. it takes the path to the raspberry OS image and the path to the sd card block device as parameters and then erases the card, creates partitins, copies the files from the image and finally adjusts what is necessary in order to adopt to the different filesystem and partition uuid's.

for the script to run you need to have the following tools installed on your linux: `kpartx,wipefs,parted,f2fs-tools` besides the usual stuff like bash etc.

here is the script, just run it without any parameters to get the usage info:

raspberry_f2fs.sh

```
#!/bin/bash
if [ "$(whoami)" != "root" ]; then
    echo "this script needs to be run as root in order to modify
partitions etc. please provide your root password for sudo to execute
this script as root"
    sudo -s /bin/bash "$0" "$@"
    exit $?
fi

image=$1
card=$2

if [ -z $1 ]; then
    echo "usage: raspberry_f2fs.sh <image> <sdcard>"
    echo "example: raspberry_f2fs.sh Downloads/raspberryos.img
/dev/sdb"
```

```
    exit 100
fi

if [ ! -f $image ]; then
    echo "ERROR"
    echo "file $image not found,"
    echo "first parameter should be the raspberry os image"
    exit 1
fi

if ! kpartx -l $image  | grep -q "loop.p1 : 0"; then
    echo "ERROR"
    echo "image $image not readable by kpartx or kpartx not found"
    echo "first parameter should be the raspberry os image, and make
sure kpartx is installed"
    exit 2
fi

if [ ! -b $card ]; then
    echo "ERROR"
    echo "$card is not a block device"
    echo "second argument should be a block device"
    exit 3
fi
echo
read -p "I will now ERASE EVERYTHING on $card, do you want to continue!
(yN)" -n 1 -r
echo
if [[ ! $REPLY =~ ^[Yy]$ ]]; then
    echo "aborted"
    exit 101
fi

# ok we are good to go :)

set -e
function clean_up(){
    umount /tmp/sd
    umount /tmp/img
    kpartx -d $image
    rmdir /tmp/{sd,img}
    echo
    echo "aborted"
}
trap clean_up EXIT

echo "make sure the card is unmounted"
set +e
umount $card* 2>/dev/null
set -e
```

```
echo "erase old partitions"
wipefs -af $card
echo "create new partitions"
parted -s $card mklabel msdos
parted -s $card mkpart primary 0% 256MB
parted -s $card mkpart primary 256MB 100%
partprobe $card
sleep 2

echo "format boot partition"
mkfs.vfat ${card}1

echo "format os partition with f2fs"
mkfs.f2fs -f ${card}2

echo "create mountpoints and mount boot partition"
mkdir -p /tmp/{sd,img}
mount ${card}1 /tmp/sd

echo "load image as loopback device and mount boot partition"
out=$(kpartx -av $image)
echo "$out"
loopdev=$(echo "$out" | sed 's/^add map \(loop[^p]\+\)p. .*$/\1/' |
head -1)
mount /dev/mapper/${loopdev}p1 /tmp/img

echo "copy the contents of the boot partition"
rsync -av /tmp/img/ /tmp/sd/

echo "adjust cmdline.txt"
partuuidbase=$(blkid /dev/sdb | sed -e
's/^.*PTUUID="\([^"]*\)".*$/\1/')
sed -i "s/\(PARTUUID=\)[^ ]*\(-02 \)/\1$partuuidbase\2/"
/tmp/sd/cmdline.txt
sed -i 's/init=[^ ]* / /' /tmp/sd/cmdline.txt

echo
echo
echo "I am done with the boot partition, now is the time to configure
your network for headless operation if you want to.."
echo
read -p "do you want to configure a wireless lan? if so, type \"y\" and
i will open a configuration template in nano for you. simply adjust,
quit and save and you are all set, otherwise press any key to continue
without configuring a wlan" -n 1 -r
echo
if [[ $REPLY =~ ^[Yy]$ ]]; then
    cat > /tmp/sd/wpa_supplicant.conf <<EOF
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
```

```
country=«your_ISO-3166-1_two-letter_country_code»

network={
    ssid="«your_SSID»"
    psk="«your_PSK»"
}
EOF
    nano /tmp/sd/wpa_supplicant.conf
fi
echo
read -p "do you want to enable the ssh-server on this raspberry right
from the start? if so, type \"y\" or else press any key to continue
without enabling ssh" -n 1 -r
echo
if [[ $REPLY =~ ^[Yy]$ ]]; then
    touch /tmp/sd/ssh
    ssh=1
else
    ssh=0
fi

echo
echo "we are done with the boot partition"
echo "unmount boot partition and mount root partition of both the card
and the image"
umount /tmp/sd
umount /tmp/img
mount ${card}2 /tmp/sd
mount /dev/mapper/${loopdev}p2 /tmp/img

echo "copy the contents of the root partition to the card, this can
take a few minutes..."
rsync --progress -aAHhvxX /tmp/img/ /tmp/sd/
echo "done copying"

echo "adjust fstab for f2fs"
sed -i "s/\(PARTUUID=\)[^ ]*\(-0[12] \)/\1$partuuidbase\2/"
/tmp/sd/etc/fstab
sed -i 's/ext4/f2fs/' /tmp/sd/etc/fstab

if [ $ssh -eq 1 ]; then
    echo "creating empty authorized_keys files for users root and pi"
    mkdir -p /tmp/sd/root/.ssh /tmp/sd/home/pi/.ssh
    chmod 700 /tmp/sd/root/.ssh /tmp/sd/home/pi/.ssh
    touch {/tmp/sd/root/.ssh,/tmp/sd/home/pi/.ssh}/authorized_keys
    chmod 600 {/tmp/sd/root/.ssh,/tmp/sd/home/pi/.ssh}/authorized_keys
    echo "done, to add your private key, boot the raspberry pi and
login as user pi with password \"raspberry\" to add your public key to
those files. make sure you don't forget to chagne the default
password!"
```

```
fi
echo
echo
echo "done preparing the root partition"
echo
read -p "do you want to configure a fixed ip address? if so, type \"y\"
and i will open the /etc/dhcpcd.conf file in a nano editor for you.
simply adjust, quit and save and you are all set, otherwise press any
key to continue without configuring your network interface" -n 1 -r
echo
if [[ $REPLY =~ ^[Yy]$ ]]; then
    nano /tmp/sd/etc/dhcpcd.conf
fi
echo
read -p "do you want to change the hostname to something meaningful
right now? if so, type \"y\" and i will open the /etc/hostname file in
a nano editor for you. simply adjust, quit and save and you are all
set, otherwise press any key to continue without changing the hostname"
-n 1 -r
echo
if [[ $REPLY =~ ^[Yy]$ ]]; then
    nano /tmp/sd/etc/hostname
fi
echo
echo "okay, we are done, time to clean up! mind you, this can take some
time as we weill have to wait for everything to be written to the SD
card. do not abort!"
umount /tmp/sd
umount /tmp/img
kpartx -d $image
rmdir /tmp/{sd,img}
echo
echo "you are all set, you can now remove the sd card and put it into
your raspberry, boot it up and start using it"
echo "if you want to insert this card into your pc in the future to
modify some things on it, I strongly recommend to disable gnomes
automount temporarily, as it may mess up your f2fs filesystem"
echo "to do that you can simply run this command:"
echo "    gsettings set org.gnome.desktop.media-handling automount
'false'"
echo "to re-enable simply use 'true' as value instead"
```

to do it manually

if you still want to do it manually, here are the steps. you can of course also create the partitions manually, rather than burning the image and booting it on the raspberry to resize the root partition to the full size of the sdcard.

here is a quick step by step howto how i installed raspbian from the minimal image onto an f2fs sd card

1. download image and dd it onto the sd card
2. boot raspberry pi from sd card and expand partition by using `raspi-config`
3. now shut down using the `shutdown`
4. insert sd card again into your pc. from here on, we suppose the sd card is /dev/sdf. in order to find its name on your system, you can use `lsblk`
5. format 2nd partition with f2fs:

```
mkfs.f2fs /dev/sdf2
```

6. mount it

```
mkdir /tmp/sd
mount -t f2fs /dev/sdf2 /tmp/sd/
```

7. mount second partition from image (see also mount_a_single_partition_from_a_dd_disk_image):

```
mkdir /tmp/image
kpartx -av raspbianimage.img
mount /dev/mapper/loop0p2 /tmp/image
```

8. now copy the os to the sd card

```
rsync -aHvx /tmp/image/ /tmp/sd/
```

9. find the partition uuid by using `blkid` and copy the “PARTUUID” number
10. edit fstab:

```
nano /tmp/sd/etc/fstab
```

and replace `ext4` in the root mount line with `f2fs`, also update the `PARTUUID` with the one you found in the previous step.

11. unmount it all again

```
umount /tmp/sd /tmp/image
```

12. mount boot partition and edit boot options:

```
mount /dev/sdf1 /tmp/sd
nano /tmp/sd/cmdline.txt
```

and now replace ext4 with f2fs in here and paste the correct `PARTUUID` as well.

13. cleanup:

```
umount /tmp/sd
rmdir /tmp/sd /tmp/image
```

14. insert the sd card into your raspberry pi and finally start configuring it :)