Ensor-384 + Raspberry Pi

Sequence of preparing a Raspberry Pi 3B+ card to work with the "Ensor-384" audio recorder card

1.- Elements necessary to start

- Ensor-384 card
- Raspberry Pi 3 B+
- Original Raspberry Pi power supply
- Original good 8GB micro SD card + USB recorder for micro SD card
- USB Flash Memory of 64GB or larger
- A computer with an internet connection with Windows or Linux OS
- The ensor384.zip file downloaded from the GitHub

2.- Install the Raspberry Pi OS Lite (64) operating system without a desktop

Use Raspberry Pi Imager by configuring the username and password,

enabling ssh server

user: ensor

password: ******** any

Recommended to use the Putty program to send commands via ssh to the Raspberry Pi

3.- Connect the Ensor-384 card to the Raspberry pi and it, in turn, connect it to the Internet using an Ethernet cable. Turn on the system

4.- update operating system

```
sudo apt update
sudo apt upgrade
```

5.- run sudo raspi-config

enable autologin with console 1 - S5 - B2

enable ssh 3 - I1

enable i2c 3 - I5

expand filesystem 6 - A1

```
sudo reboot
```

6.- Connect a USB flash memory, with a capacity greater than or equal to 64GB, to the raspi-3b+ card and check its label.

The USB flash memory will be formatted later in exFAT

```
sudo fdisk -l
        -> /dev/sda1
```

The USB flash memory will be used to store the recording configuration file “ensor.conf” and the audio recordings

1 hour of audio recording 2 stereo channels at 192000 m/s and 32 bits occupy 5.4 GB

1 hour of audio recording 2 stereo channels at 384000 m/s and 32 bits occupy 10.8 GB

7.- Mount the USB flash memory at system startup

The USB flash memory will be mounted in the following path

/media/ensor/Ensor384

create the ensor directory and ensor/Ensor384 in /media

```
sudo mkdir -p /media/ensor/Ensor384
```

add the following line in the /etc/fstab file

```
sudo nano /etc/fstab
```

```
/dev/sda1          /media/ensor/Ensor384            auto     auto,user,rw,umask=000,nofail,x-system.device-timeout=10        0           0
```

test that the changes in /etc/fstab are correct

```
sudo mount -a
sudo systemctl daemon-reload


sudo reboot
```

to check that the USB flash memory is mounted correctly

8.- The USB flash memory will be formatted in exFAT

This operation can be easily done in Windows,

or on Linux, as below

```
sudo mkfs.exfat -n TAG /dev/sdXn
```

in our case

sudo fdisk -l to see /dev/sdXn

```
sudo mkfs.exfat -n Ensor384 /dev/sda1
```

9.- File ensor384.zip

The compressed file ensor384.zip will be downloaded from the Internet, which includes the Ensor directory and the audio configuration file ensor.conf

Unzip ensor384.zip and copy the ensor.conf file and the Ensor directory to the USB flash memory

Connect the USB flash memory to the Raspberry Pi and turn on

On the USB flash memory, there will be the recording configuration file:

ensor.conf

and the Ensor directory, which will be copied to:

```
/home/ensor/
cp -dr /media/ensor/Ensor384/Ensor /home/ensor/
```

Once the Ensor directory has been copied, if desired, it can be deleted from the USB flash memory

```
rm -dr /media/ensor/Ensor384/Ensor
```

To record files larger than 4GB.

Install exfat-fuse on raspi

```
sudo apt install exfat-fuse
```

10.- Copy the compiled dts file of the "Ensor-384" audio recording card to /boot/firmware/overlays

The .dts file is located in the /home/ensor/Ensor/dts/ directory

```
/home/ensor/Ensor/dts/tlv320adcx140-overlay.dts
```

We will have to compile it with the following command:

```
sudo dtc -@ -H epapr -O dtb -o tlv320adcx140-overlay.dtbo -Wno-unit_address_vs_reg tlv320adcx140-overlay.dts
```

and the compiled file tlv320adcx140-overlay.dtbo copy it to /boot/firmware/overlays/

```
sudo cp /home/ensor/Ensor/dts/tlv320adcx140-overlay.dtbo /boot/firmware/overlays/
```

11.- modify the /boot/firmware/config.txt file

```
sudo nano /boot/firmware/config.txt
    dtparam=i2c=on
```

```
dtparam=i2s=on

gpio=4=op,dh

dtoverlay=tlv320adcx140-overlay


#disable the default audio cards

#dtparam=audio=on


#The following cannot be disabled because the screen does not #work when booting again

dtoverlay=vc4-kms-v3d


#disable wifi and buetooth to avoid noise and consumption


dtoverlay=disable-wifi

dtoverlay=disable-bt
```

12.- boot the system again and check that the ADCX140 recording card has been installed

```
arecord -l
```

check that card is 0 and device is 0

```
card 0: ADCX140 [ADCX140], device 0:
```

which are the necessary parameters for the recording program

```
/home/ensor/Ensor/eea.py
```

13.- Check that the card records audio at a sampling rate of 192000 m/s and 32 bits

The recording program is located in /home/ensor/Ensor/eea.py

and the recording configuration file, ensor.conf, is located on the USB flash memory

If the USB flash memory does not have the ensor.conf configuration file recorded, the eea.py program creates one by default with the data shown below and the program will be executed with that data.

By default, the system does not record until "record=yes" is indicated.

Recording parameters can be modified with any text editor

```
nano /media/ensor/Ensor384/ensor.conf

[DEFAULT]

record = no       #enable recording or not <yes, no>

                  #By default, it does not record

date = 01/12/2024

filename = test_192Ks_32bits


[sound.format]

sampling = 192000

bits = 32


[record.time]

total_time = 10 #total recording time in #seconds

file_time = 10 #since the files are very large, #it is interesting to split them into several

delay = 0 #wait time, in seconds, from #start, before starting to record

index = 0
```

A directory structure will be created depending on the current date and the file names will depend on the current hour and minute ending with the file number.

modify the configuration file to record

```
nano /media/ensor/Ensor384/ensor.conf

[DEFAULT]

record =  yes      #enable recording or not <yes, no>

. . .              #by default, it does not record
```

And try to record

```
sudo /home/ensor/Ensor/eea.py
```

It is necessary to use sudo to modify the system date.

the program has a delay of 10 seconds to give time for the USB flash memory to mount correctly when we automatically record when starting the system.

14.- Record when the system starts

we will use the rc-local.service service

we will modify the /etc/rc.local file

```
sudo nano /etc/rc.local
```

insert the following line

```
/home/ensor/Ensor/recorder.sh
```

so that the recorder.sh script is executed when the system starts

15.- Script program /home/ensor/Ensor/recorder.sh

```
nano /home/ensor/Ensor/recorder.sh
```

```
#!/bin/sh
sudo /home/ensor/Ensor/eea.py
sudo shutdown -h now
```

what this script does is start recording according to the ensor.conf configuration and when it finishes, it shuts down the system so that it does not consume energy.

16.- Backup the card

widows: Win32DiskImager

linux: dd if=/dev/sda of=/image/path

from raspi: sudo dd if=/dev/mmcblk0 of=/media/ensor/Ensor384/raspbian.img bs=1M status=progress

17.- Prepare Raspberry Pi to record at 384000 samples/s and 32 bits

The system will be prepared with everything necessary to compile the kernel

```
sudo apt install git bc bison flex libssl-dev make
```

The current kernel source files will be downloaded

```
git clone --depth=1 https://github.com/raspberrypi/linux
```

18.- The "menuconfig" tool requires ncurses

```
sudo apt install libncurses5-dev libncursesw5-dev
```

19.- We will use spdif-receiver as the digital audio interface

Since spdif-receiver supports a maximum sampling rate of 192000 samples/s,

we will have to modify spdif-receiver to support 384000 samples/s

the file is located in

```
/home/ensor/linux/
```

```
nano spdif_receiver.c.
```

where it says 192000 put 384000 instead and compile kernel

```
#define STUB_RATES SNDRV_PCM_RATE_8000_384000
#define STUB_FORMATS (SNDRV_PCM_FMTBIT_S16_LE | \
SNDRV_PCM_FMTBIT_S20_3LE | \
SNDRV_PCM_FMTBIT_S24_LE | \
SNDRV_PCM_FMTBIT_S32_LE | \
SNDRV_PCM_FMTBIT_IEC958_SUBFRAME_LE)
```

20.- Compile kernel 6.6.44-v8+ Raspberry Pi OS Lite (64) for Raspberry Pi 3b+

```
cd linux
KERNEL=kernel8
make bcm2711_defconfig


make menuconfig


make -j4 Image.gz modules dtbs
```

```
sudo make -j4 modules_install


sudo cp /boot/firmware/ $KERNEL.img /boot/firmware/$KERNEL-backup.img
sudo cp arch/arm64/boot/Image.gz /boot/firmware/$KERNEL.img
sudo cp arch/arm64/boot/dts/broadcom/*.dtb /boot/firmware/
sudo cp arch/arm64/boot/dts/overlays/*.dtb* /boot/firmware/overlays/
sudo cp arch/arm64/boot/dts/overlays/README /boot/firmware/overlays/


sudo reboot
```

21.- Check that the card records audio at a sampling rate of 384000 samples per second and 32 bits

The recording program is located in /home/ensor /Ensor/eea.py

Edit the configuration file to record at 384000 m/s and 32bits for 10 seconds

```
nano /media/ensor/Ensor384/ensor.conf
[DEFAULT]
record = yes      # enable recording or not < yes, no >
                  #by default, does not record
date = 01/12/2024
file_name = test_384Ks_32bits




[sound.format]
sampling = 384000
bits = 32


[record.time]
totaltime = 10   #total recording time in #seconds
fileTime = 10    #since the files are very large, #it is better to divide them into
several
```

```
delay = 0		#waiting time, in seconds, from the start, before starting to record

index = 0
```

try recording

```
sudo /home/ensor/Ensor/eea.py
```

If the system records perfectly, perform another recording test, but this time automatically from the start of the system. To do this, simply leave the configuration file as it is and reboot the system

```
sudo reboot
```

When the system boots, it will begin recording as indicated in the configuration file.

As the recording configuration file, ensor.conf, is on the USB flash memory, it can be modified with any text editor in both Windows and Linux

In this simple way, automatic recordings can be made.

You will simply set the configuration that is needed at any given time

22.- Backup the card

widows: Win32DiskImager

linux: dd if=/dev/sda of=/image/path

from raspi: sudo dd if=/dev/mmcblk0 of=/media/ensor/Ensor384/raspbian_64_6_6_42.img bs=1M status=progress