

# UGLYDATV 0.1

## by F5OEO Evariste

November 2014

### Introduction

This documentation describes a solution to use the Raspberry Pi as a main component of a DVB-S modulator. Two modes are available :

- Output I/Q bitstream compliant ready to feed an external QPSK modulator (IQ mode)
- A direct HF QPSK modulator (Ugly mode)

#### **Particular thanks to :**

Brian G4EWJ for sharing his very efficient code of canal encoding in ARM Assembler

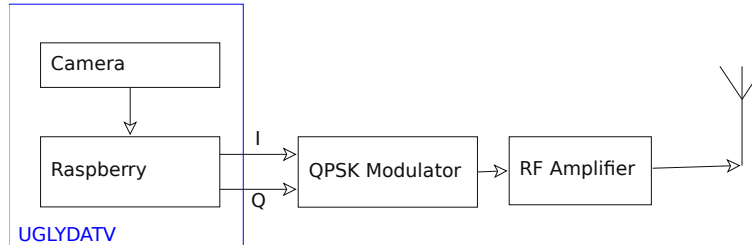
Perceval for his linux emergency support

F4DAY for pionnering the Poor Man DATV

PiFM for idea of using GPIO for transmi

# IQ Mode

## General diagram

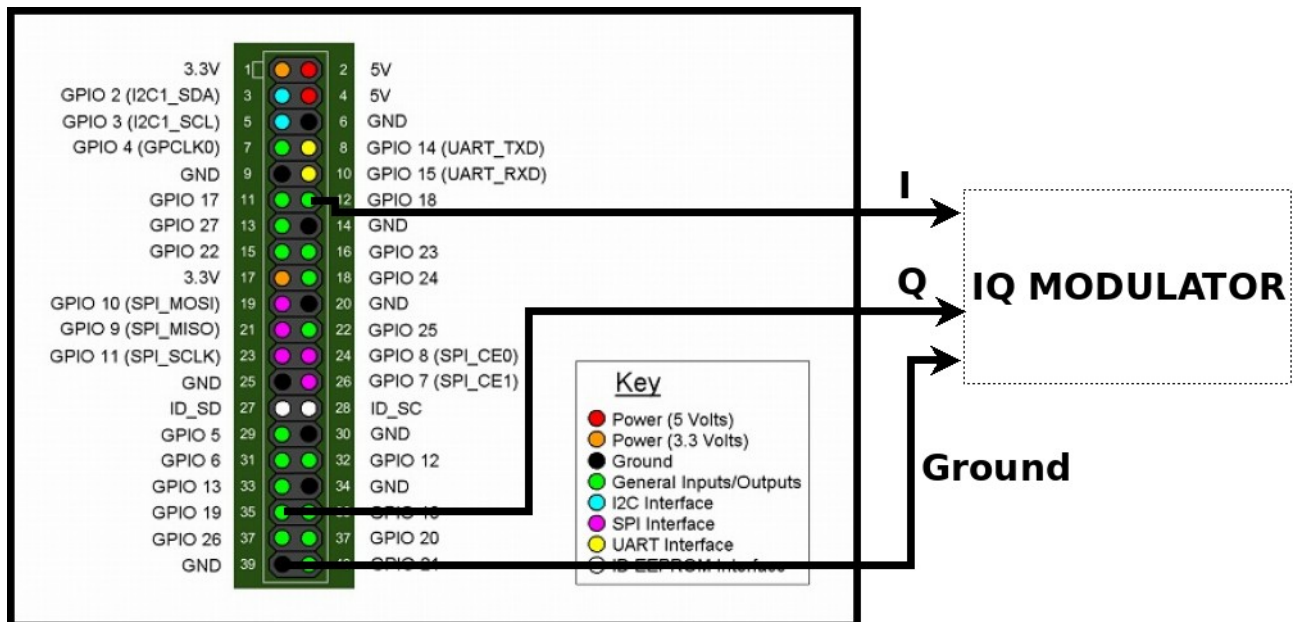


## Hardware required

- [Raspberry Pi Model B+](#) with a Micro SD card of 8 GigaBytes minimum
- [Camera](#)
- External QPSK Modulator and RF amplifier
- RF amplifier and antenna

## Link Raspberry to IQ modulator

To link output IQ of Raspberry to an external modulator :



Take care of pin numbering : PIN #1 is located the closest to MicroSD card on B+

Pin 12 : output I

Pin 35 : output Q

pin 39 : Ground (you could also take any other ground available on the Header)

Output voltage of I and Q is 0-3.3V.

## Software required

A complete system to write on the SD card : <http://f5oeo.fr/UglyDATV0.1.img.zip>

You have to extract the image with (Winzip ou other zip decompressor) and then follow the instructions like : [Image installation instruction](#)

Then insert the SD card in the Raspberry and power on.

For now the software requires a minimum of commands to launch the modulation. In a future release, all will be launch automatically and will not require any action.

In order to communicate with the Raspberry there are 2 methods :

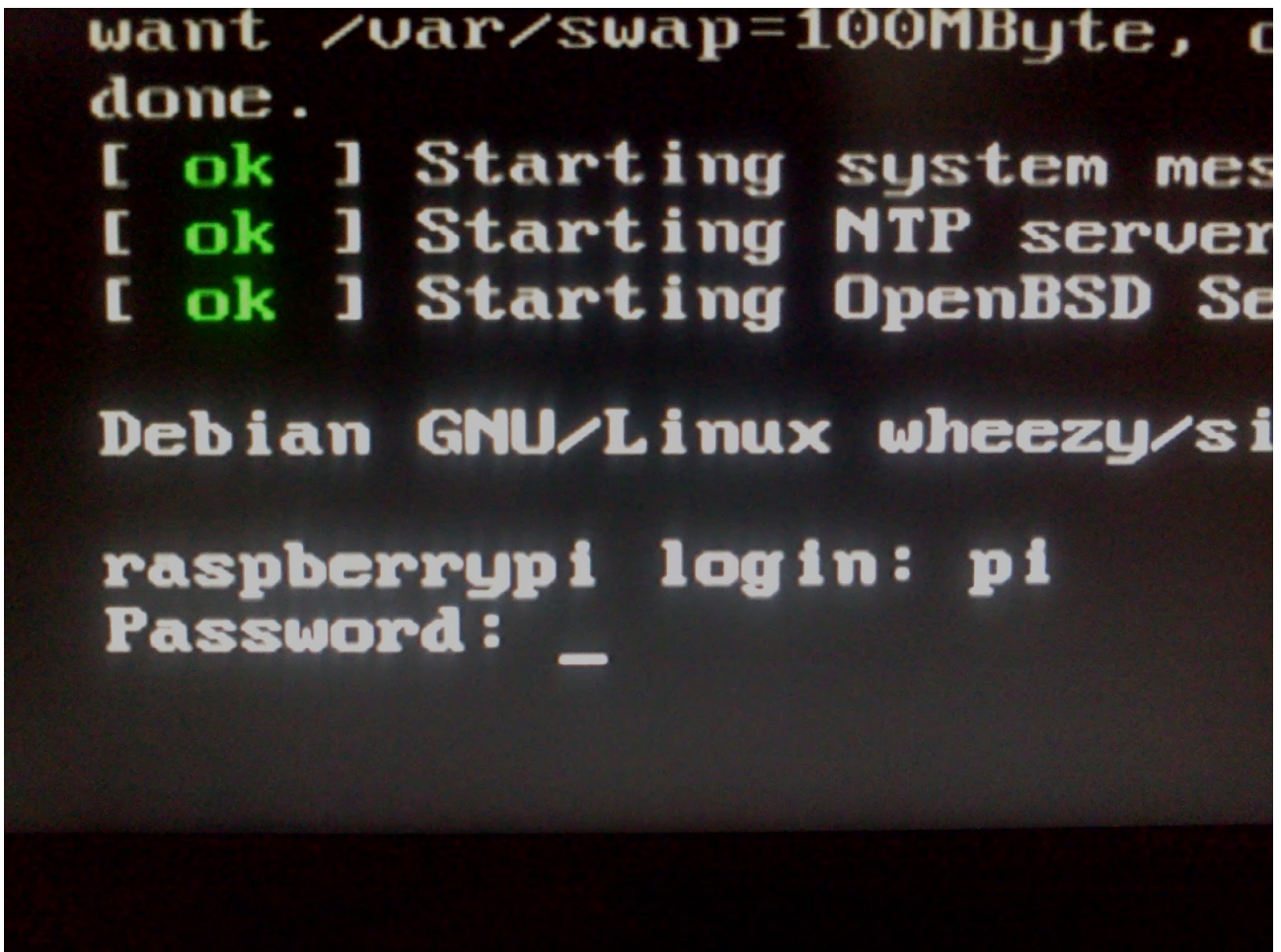
- Easy :

1. Plug a display by HDMI or composite output
2. Plug a USB keyboard

- More advanced :

1. Plug an Ethernet cable link to a local network included your PC
2. With PC, discover which IP address is assigned to Raspberry
3. Connect remotely to raspberry using SSH (putty software is your friend)

At this stage you see a prompt



```
want /var/swap=100MByte, d
done.
[ ok ] Starting system mes
[ ok ] Starting NTP server
[ ok ] Starting OpenBSD Se

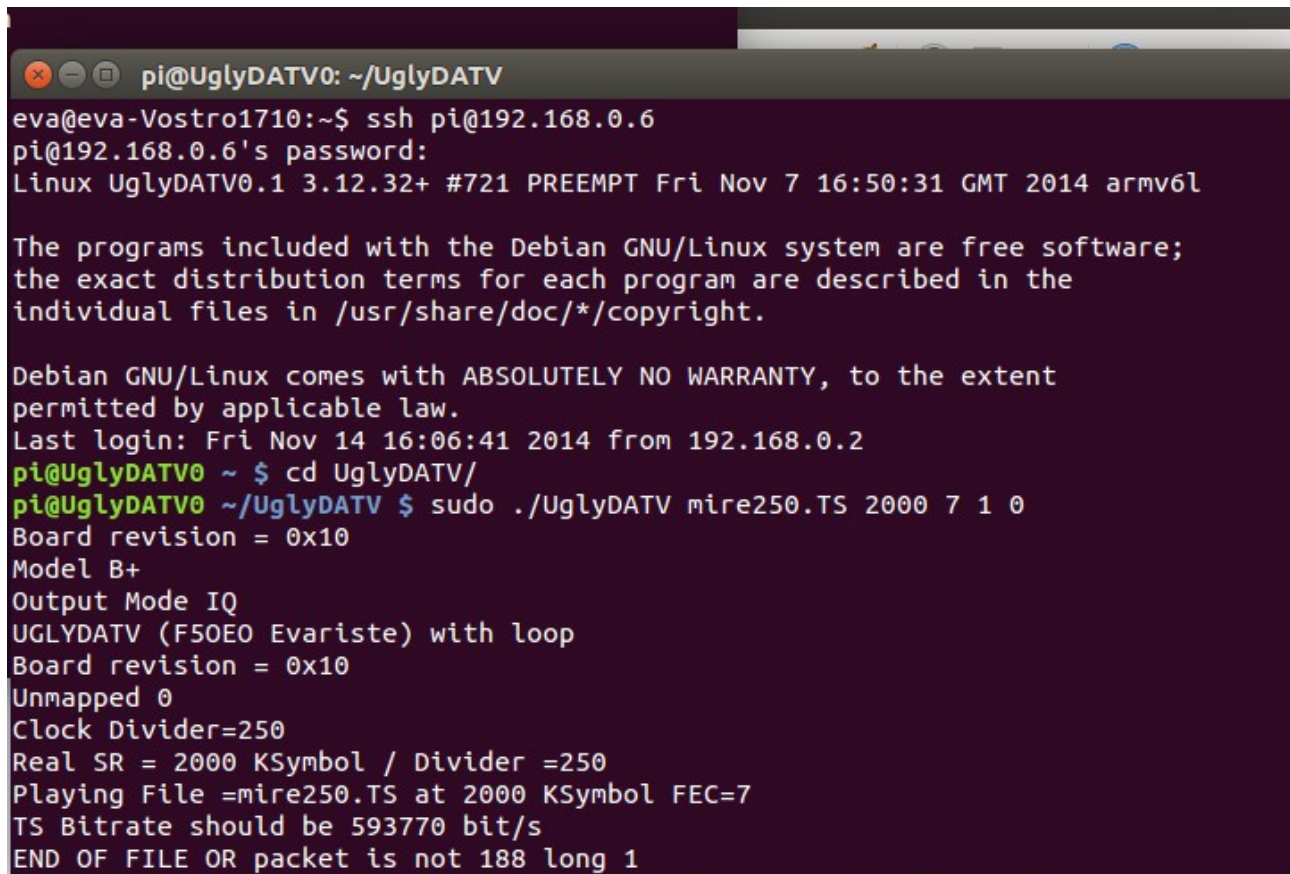
Debian GNU/Linux wheezy/si

raspberrypi login: pi
Password: _
```

Type pi for login and tv for password.

Note : while typing the password, it is normal that nothing is displayed on the screen (no stars like on windows)

You are now few step to test first transmission : type commands like the picture bellow



```
pi@UglyDATV0: ~/UglyDATV
eva@eva-Vostro1710:~$ ssh pi@192.168.0.6
pi@192.168.0.6's password:
Linux UglyDATV0.1 3.12.32+ #721 PREEMPT Fri Nov 7 16:50:31 GMT 2014 armv6l

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Nov 14 16:06:41 2014 from 192.168.0.2
pi@UglyDATV0 ~ $ cd UglyDATV/
pi@UglyDATV0 ~/UglyDATV $ sudo ./UglyDATV mire250.TS 2000 7 1 0
Board revision = 0x10
Model B+
Output Mode IQ
UGLYDATV (F50E0 Evariste) with loop
Board revision = 0x10
Unmapped 0
Clock Divider=250
Real SR = 2000 KSymbol / Divider =250
Playing File =mire250.TS at 2000 KSymbol FEC=7
TS Bitrate should be 593770 bit/s
END OF FILE OR packet is not 188 long 1
```

It launch the IQ modulator at 2000 Ksymbol, FEC 7/8. You should be able to receive it on a set top box if you have plugged a IQ modulator (at least find the channel (service name) , maybe not the video at this point).

## UglyDATV Software details

UglyDATV software take a Transport stream input file. It then processes it to transform to a DVB-S IQ compliant stream according to Symbol Rate (Rate of transmission) and FEC (Error correction).

Effective rate of data is calculated by

Transport Stream effective rate = Symbol Rate \* 2 (QPSK=2 bits by symbol) \* 188/204 (16 bytes added at each 188 packet ) \* FEC

For example for a SR 2000 and FEC 7/8 :

TS= 2000\*2\*188/204\*7/8 = 3225490 bit/s (seems that there is a bug in the UglyDATV calculation)

For a live stream, this Transport Rate should be feed in UglyDATV in order to not have underflow neither overflow.

For file playing, UglyDATV pump the data at the required rate and thus has a perfect timing. However, in the Transport Stream file itself, there are some clock inside which help the set top box to decoded properly the video and synchronize it with the sound. If the transport file you play is not calculated for the rate you send in UglyDATV, you could have issues with decoding the video which will be very slow or drop frame.

## Parameters of UglyDATV

`sudo ./UglyDATV TransportFile SymbolRate FEC Loop Mode`

TransportFile : a transport stream file containing packets of 188 bytes

SymbolRate : Rate in Ksymbols (tested from 250 to 3000)

Loop : To loop the input file (usefull to have a continuous transmit of short TS File)

Mode : 0 is for mode IQ

## Some scripts

Some scripts are present in the UglyDATV folder. It helps to launch several processes at the same time, for example launching the camera (raspivid) , put the camera stream in a Transport stream format (ffmpeg) and then feed UglyDATV.

To launch it : `./a.sh`

a.sh : Transmit the camera at 250KSymbol/s

Be carefull classical set top boxes could not receive this very low symbol rate. You should adapt the a.sh script if you want other SymbolRate

Modifying the script a.sh : `nano a.sh`

```
sudo nice -n -30 raspivid -s -n -w 320 -h 288 -b 300000 -t 0 -pf high -fps 15 -g 12 -ih -o videoes &
```

```
sudo /home/pi/UglyDATV/UglyDATV videots 250 7 0 0 &
```

```
sudo nice -n -30 /home/pi/UglyDATV/ffmpeg -loglevel debug -analyzeduration 0 -probesize 2048  
-r 15 -fpsprobesize 0 -max_delay 40000 -i videoes -f h264 -r 15 -minrate 300K -maxrate 300K  
-bufsize 20K -vcodec copy -bufsize 20K -f mpegts -mpegts_original_network_id 1  
-mpegts_transport_stream_id 1 -mpegts_service_id 100 -mpegts_pmt_start_pid 1000  
-mpegts_start_pid 1001 -metadata service_provider="F5OEO" -metadata service_name="F5OEO-  
1" -muxrate 403200 -y videots &
```

Number in bold have to be changed in order to send to an other SymbolRate.

If we want a Symbolrate of 2000 : modify 250 to 2000 : this is the symbolrate

And modify 403200 by 3225490 (which is the bitrate of the TS see above for details of calculation)

But at this point , we just setting the output rate of the TS. We have now a bigger canal, but our video encoding is still at low bitrate, it means that we mainly send some padding packets.

Let's have a higher quality in video by changing the resolution.

```
sudo nice -n -30 raspivid -s -n -w 320 -h 288 -b 300000 -t 0 -pf high -fps 15 -g 12 -ih -o videoes &  
sudo /home/pi/UglyDATV/UglyDATV videots 2000 7 0 0 &
```

```
sudo nice -n -30 /home/pi/UglyDATV/ffmpeg -loglevel debug -analyzeduration 0 -probesize 2048  
-r 15 -fpsprobesize 0 -max_delay 40000 -i videoes -f h264 -r 15 -minrate 300K -maxrate 300K  
-bufsize 20K -vcodec copy -bufsize 20K -f mpegts -mpegts_original_network_id 1  
-mpegts_transport_stream_id 1 -mpegts_service_id 100 -mpegts_pmt_start_pid 1000  
-mpegts_start_pid 1001 -metadata service_provider="F5OEO" -metadata service_name="F5OEO-  
1" -muxrate 3225490 -y videots &
```

320 : width of the video

288 : height of video

300000 : bitrate of video

15 : FrameRate

300K : bitrate of video (setting for the transport stream tool)

So we can change as we have plenty of bandwidth for a 720\*576 resolution at 25 frame/s and a video bitrate of 2800000

```
sudo nice -n -30 raspivid -s -n -w 720 -h 576 -b 2800000 -t 0 -pf high -fps 25 -g 12 -ih -o videoes  
&
```

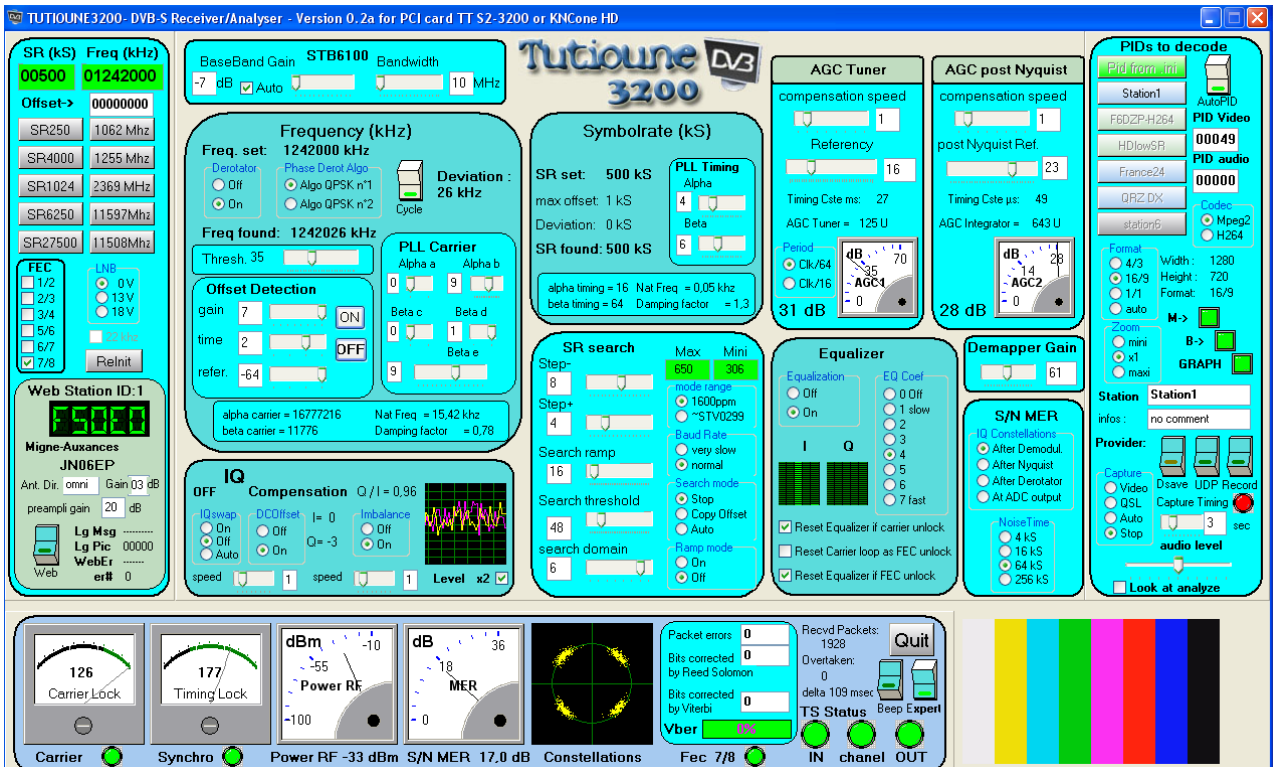
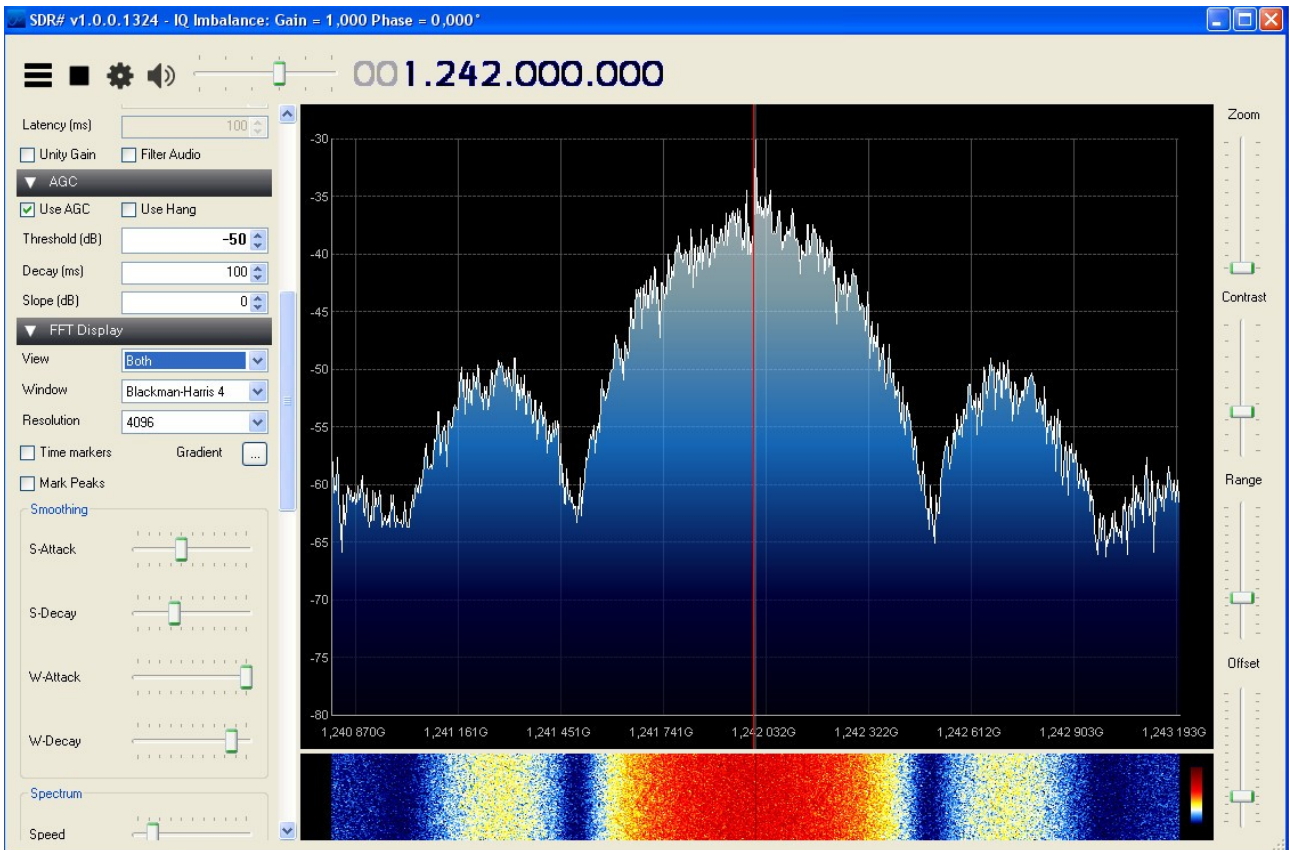
```
sudo /home/pi/UglyDATV/UglyDATV videots 2000 7 0 0 &
```

```
sudo nice -n -30 /home/pi/UglyDATV/ffmpeg -loglevel debug -analyzeduration 0 -probesize 2048  
-r 15 -fpsprobesize 0 -max_delay 40000 -i videoes -f h264 -r 15 -minrate 2800K -maxrate 2800K  
-bufsize 20K -vcodec copy -bufsize 20K -f mpegts -mpegts_original_network_id 1  
-mpegts_transport_stream_id 1 -mpegts_service_id 100 -mpegts_pmt_start_pid 1000  
-mpegts_start_pid 1001 -metadata service_provider="F5OEO" -metadata service_name="F5OEO-  
1" -muxrate 3225490 -y videots &
```

**Important note : Video bitrate should not exeding 80 % of the total bitrate :**

**3500K\*0.8=2800K**

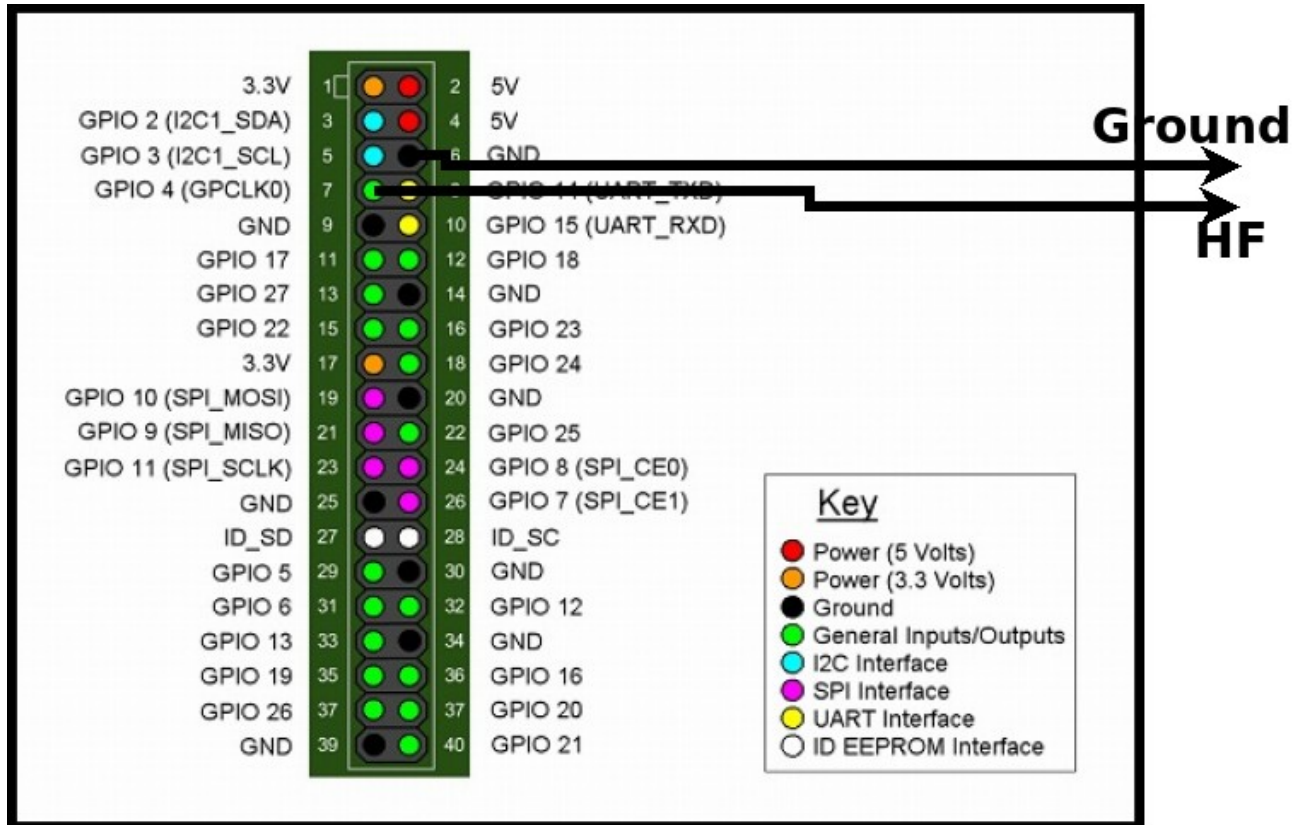
To stop the process : press CTRL C



## Mode UGLY

This is a particular mode : QPSK RF is directly available on the pin 7 of the raspberry GPIO.

It doesn't require any other hardware !



**CAREFULL : Schematic is wrong, HF is on pin 12 of the header (GPIO18) not on pin 7 !!!**

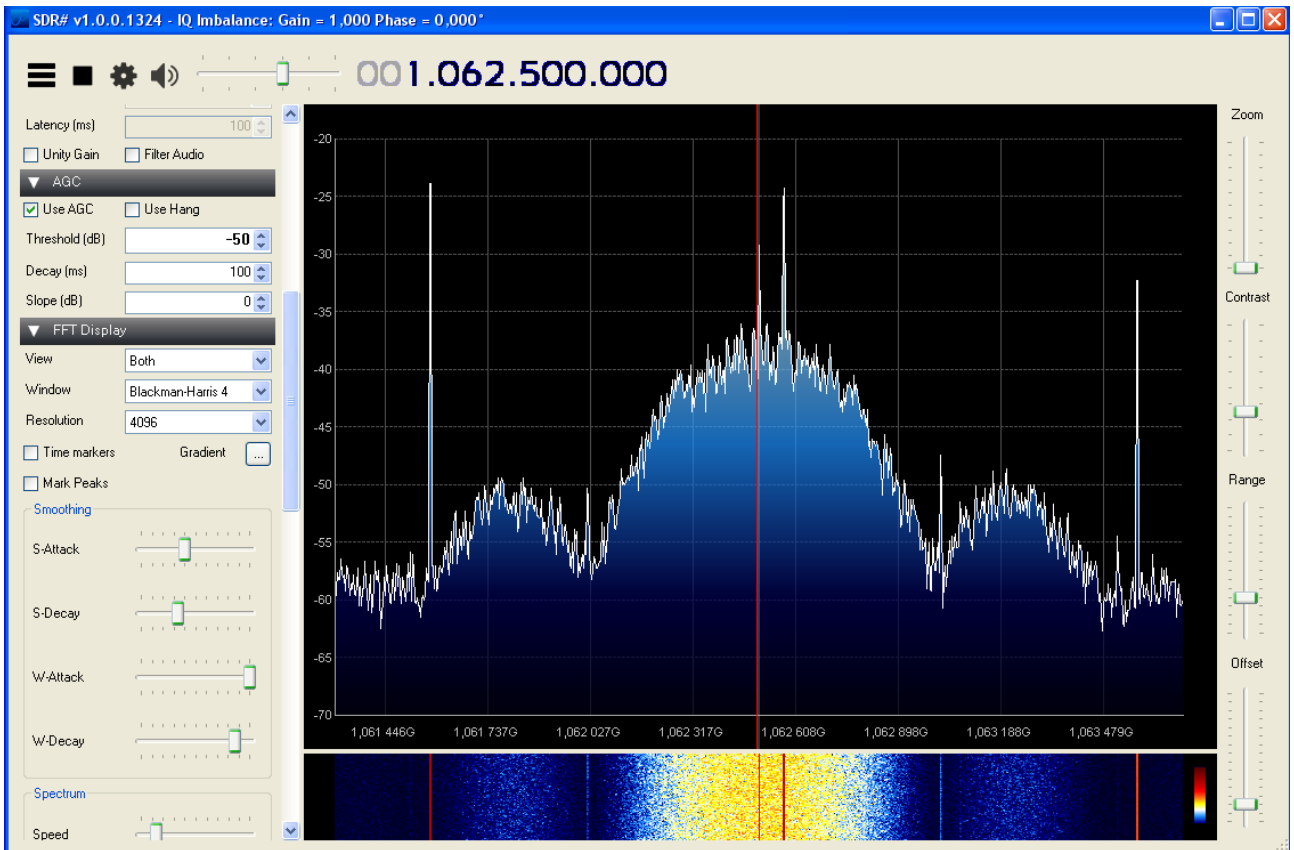
This called Ugly because the RF modulation is done with square signals which implies lot of harmonics (mainly even 1,3,5..).

RaspberryPi has a PLL Clock programmable wich can generate frequency up to 250 MHZ. As we have to be precise at  $\text{Frequency} \times 4$  (QPSK), we can generate a RF signal at up to 62.5MHZ.

As the signal is square, we can receive also all even harmonics : for example harmonic 7 of 62.5Mhz is 437.5MHZ which is in our ATV HAM Band. Adding a bandpass filter on 437 and you have a ready to send DATV RF Signal. There is no nyquist filtering thus we have all the SymbolRate harmonics.

For easy receiving on a set top box, we can listen the 17th harmonic (which has still an enough power to be received). So here is the result at 1.0625GHz.





TUTIUONE3200 - DVB-S Receiver/Analyser - Version 0.2a for PCI card TT S2-3200 or KNCone HD

BaseBand Gain: STB6100 Bandwidth: 10 MHz

SR (kS) Freq (kHz): 00500 01062500

Offset: 00000000

SR250: 1062 Mhz  
SR4000: 1255 Mhz  
SR1024: 2369 Mhz  
SR6250: 11597 Mhz  
SR27500: 11508 Mhz

FEC: 1/2, 2/3, 3/4, 5/6, 6/7, 7/8

Web Station ID: 1  
Migne-Auxances JN06EP

Frequency (kHz): 1062492 kHz  
Deviation: 51 kHz

Symbolrate (kS): 500 kS  
SR found: 500 kS

AGC Tuner: 14 dB  
AGC post Nyquist: 28 dB

PIDs to decode: Station1, FBDZPH264, HDlowSR, France24, station6

Station: Station1

Carrier Lock: 126  
Timing Lock: 168  
Power RF: -69 dBm  
S/N MER: 14,0 dB

Packet errors: 0  
Bits corrected by Reed Solomon: 169  
Bits corrected by Viterbi: 12  
vber: 0%

Recvd Packets: 2578  
Overtaken: 0  
delta 110 msec

IN channel OUT

## Command to use UGLY mode

This is the same software, just the latest parameter which is the frequency in MHZ of the command line set the mode to Ugly :

```
sudo ./UglyDATV /media/usb0/videots/avsync.ts 500 7 1 62.5
```

Board revision = 0x10

Model B+

TuneFrequency = 1062.500000 Mhz at harmonic 17

UGLYDATV (F5OEO Evariste) with loop

Board revision = 0x10

Unmapped 0

Clock Divider=1000

Real SR = 500 KSymbol / Divider =1000

Playing File =/media/usb0/videots/avsync.ts at 500000 KSymbol FEC=7

TS Bitrate should be 806372 bit/s

END OF FILE OR packet is not 188 long 0

Note : All the scripts which are used with IQ mode could be used in Ugly mode by setting the last parameter.

**IMPORTANT : Symbol Rate should not exceed 1500KSymbol/s**

## Raspberry Model B (Old one)

It is possible to use UglyDATV with old Model B. However the header has only 26 pins instead of 40 on B+.

For IQ mode, you need to solder 2 wires on the board:

See the picture below.

It take PWM0 and PWM1 signals which are I and Q. Taken from R21 and R27.

See schematics :

[http://www.raspberrypi.org/wp-content/uploads/2012/10/Raspberry-Pi-R2.0-Schematics-Issue2.2\\_027.pdf](http://www.raspberrypi.org/wp-content/uploads/2012/10/Raspberry-Pi-R2.0-Schematics-Issue2.2_027.pdf)

For Ugly Mode : this works identical compare to B+

