

REPETIDOR DMR PUERTO LOS LEONES ED4ZAH

Contenido

REPETIDOR DMR PUERTO LOS LEONES ED4ZAH.....	1
INTRODUCCIÓN	1
PREPARACION RASPBERRY.....	2
INSTALACIÓN HERRAMIENTAS DE DESARROLLO	2
CONFIGURAR MMDVMHOST COMO SERVICIO.....	4
ACTUALIZACIÓN DMRIids	5
CONFIGURACIÓN MODEM ST32M.....	7
CONEXIONADO EQUIPOS Y MODEM	10
CONECTOR MOTOROLA HOMEMADE.....	11
MODIFICACIÓN SALIDA AUDIO	12
AJUSTE DEL EQUIPO TX	13

INTRODUCCIÓN

Este documento describe como se ha realizado la configuración, instalación y puesta en marcha del repetidor DMR de la Sección de Ure Guadarrama [@ea4urg](#) .

El objetivo de este documento es facilitar el trabajo a otras secciones o Radio Clubs que deseen montar su propio repetidor.

El trabajo descrito a continuación ha sido llevado a cabo por el radio club Mike Alfa Delta [@radio mad](#) .

Este documento se puede distribuir libremente siempre y cuando no sea modificado y se distribuya en su formato original.

PREPARACION RASPBERRY

Una vez grabada la SD con la última versión disponible de raspbian pasamos a configurar nuestra raspberry.

En las últimas versiones de raspbian es necesario crear un fichero llamado SSH en la SD para que la raspberry permita ser accedida mediante ese protocolo... El fichero ha de llamarse "ssh" sin mas, sin extensión y no es necesario que tenga nada dentro del fichero. Así que una vez grabada la SD, sacarla, meterla de nuevo, y crear ese fichero.

Por defecto el usuario es pi y el password raspberry

INSTALACIÓN HERRAMIENTAS DE DESARROLLO

Instalamos las herramientas de desarrollo y algunos paquetes que también utilizaremos

```
sudo apt-get update
sudo apt-get install build-essential git nano screen htop
```

Ahora descargamos de GIT el software necesario

```
cd /opt/
sudo git clone https://github.com/g4klx/MMDVMHost.git
sudo git clone https://github.com/g4klx/MMDVMCal.git
```

Y procedemos a compilar MMDVMHost y MMDVMCal

```
cd /opt/MMDVMHost
sudo make
cd /opt/MMDVMCal
sudo make
```

```
pi@raspberrypi:/opt/MMDVMHost $ sudo make
echo "const char *gitversion = \"139be054be42503b52f6b03fea972d287a2395a7\";" > GitVersion.h
g++ -g -O3 -Wall -std=c++0x -pthread -c -o AMBEFEC.o AMBEFEC.cpp
g++ -g -O3 -Wall -std=c++0x -pthread -c -o BCH.o BCH.cpp
g++ -g -O3 -Wall -std=c++0x -pthread -c -o BPTC19696.o BPTC19696.cpp
g++ -g -O3 -Wall -std=c++0x -pthread -c -o Conf.o Conf.cpp
g++ -g -O3 -Wall -std=c++0x -pthread -c -o CRC.o CRC.cpp
g++ -g -O3 -Wall -std=c++0x -pthread -c -o Display.o Display.cpp
```

Tardará un poco....

ahora vamos a editar el fichero MMDVM.INI y configurarlo

```
cd /opt/MMDVMHost
sudo nano MMDVM.ini
```

Una vez configuramos lanzamos MMDVMHost para verificar si va todo correcto

```
sudo ./MMDVMHost ./MMDVM.ini
```

```
I: 2017-09-19 20:43:16.754 RX Level: 50.0%
I: 2017-09-19 20:43:16.754 CW Id TX Level: 50.0%
I: 2017-09-19 20:43:16.755 D-Star TX Level: 50.0%
I: 2017-09-19 20:43:16.755 DMR TX Level: 50.0%
I: 2017-09-19 20:43:16.755 YSF TX Level: 50.0%
I: 2017-09-19 20:43:16.756 P25 TX Level: 50.0%
I: 2017-09-19 20:43:16.756 RX Frequency: 438325000Hz (438325000Hz)
I: 2017-09-19 20:43:16.757 TX Frequency: 430725000Hz (430725000Hz)
M: 2017-09-19 20:43:16.757 Opening the MMDVM
I: 2017-09-19 20:43:18.779 MMDVM protocol version: 1, description: MMDVM 20170501 (D-Star/DMR/System Fusion/P25/RSS
/CW Id) GitID #0000000
I: 2017-09-19 20:43:18.800 Display Parameters
I: 2017-09-19 20:43:18.800 Type: None
I: 2017-09-19 20:43:18.801 DMR Network Parameters
I: 2017-09-19 20:43:18.801 Address: bm-ea.dyndns.org
I: 2017-09-19 20:43:18.801 Port: 62031
I: 2017-09-19 20:43:18.802 Local: random
I: 2017-09-19 20:43:18.802 Jitter: 300ms
I: 2017-09-19 20:43:18.802 Slot 1: enabled
I: 2017-09-19 20:43:18.803 Slot 2: enabled
I: 2017-09-19 20:43:18.803 Mode Hang: 3s
I: 2017-09-19 20:43:19.075 Info Parameters
I: 2017-09-19 20:43:19.075 Callsign: ED4ZAH
I: 2017-09-19 20:43:19.076 RX Frequency: 438325000Hz
I: 2017-09-19 20:43:19.076 TX Frequency: 430725000Hz
I: 2017-09-19 20:43:19.076 Power: 1W
I: 2017-09-19 20:43:19.077 Latitude: 40.714230deg N
I: 2017-09-19 20:43:19.077 Longitude: -4.138111deg E
I: 2017-09-19 20:43:19.078 Height: 1556m
I: 2017-09-19 20:43:19.078 Location: "Spain"
I: 2017-09-19 20:43:19.079 Description: "Multi-Mode Repeater"
I: 2017-09-19 20:43:19.079 URL: "www.google.es"
M: 2017-09-19 20:43:19.079 DMR, Opening DMR Network
I: 2017-09-19 20:43:19.080 CW Id Parameters
I: 2017-09-19 20:43:19.080 Time: 10 mins
I: 2017-09-19 20:43:19.080 Callsign: ED4ZAH
I: 2017-09-19 20:43:19.081 RSSI
I: 2017-09-19 20:43:19.081 Mapping File: RSSI.dat
I: 2017-09-19 20:43:19.081 Loaded 0 RSSI data mapping points from RSSI.dat
I: 2017-09-19 20:43:19.082 DMR Id Lookups
I: 2017-09-19 20:43:19.082 File: /opt/MMDVMHost/DMRIds.dat
I: 2017-09-19 20:43:19.082 Reload: 24 hours
W: 2017-09-19 20:43:19.082 Cannot open the Id lookup file - /opt/MMDVMHost/DMRIds.dat
I: 2017-09-19 20:43:19.083 DMR RF Parameters
I: 2017-09-19 20:43:19.083 Started the DMR Id lookup reload thread
I: 2017-09-19 20:43:19.083 Id: 2144087
I: 2017-09-19 20:43:19.084 Color Code: 1
I: 2017-09-19 20:43:19.084 Self Only: no
I: 2017-09-19 20:43:19.084 Embedded LC Only: no
I: 2017-09-19 20:43:19.085 Dump Talker Alias Data: yes
I: 2017-09-19 20:43:19.085 Prefixes: 0
I: 2017-09-19 20:43:19.085 Call Hang: 3s
I: 2017-09-19 20:43:19.086 TX Hang: 3s
I: 2017-09-19 20:43:19.086 Mode Hang: 10s
M: 2017-09-19 20:43:19.086 MMDVMHost-20170719 is running
```

Copiamos el fichero MMDVM.ini a HOME/PI

```
sudo cp /opt/MMDVMHost/MMDVM.ini /home/pi/MMDVM.ini
```

Copiamos el binario MMDVMHost y el MMDVMCal

```
sudo cp /opt/MMDVMHost/MMDVMHost /usr/local/bin/MMDVMHost
sudo cp /opt/MMDVMCal/MMDVMCal /usr/local/bin/MMDVMCal
```

CONFIGURAR MMDVMHOST COMO SERVICIO

Creamos el fichero de configuración del servicio

```
sudo nano /lib/systemd/system/mmdvmhost.service
```

Debe quedar así

```
[Unit]
Description=MMDVM Host Service
After=syslog.target network.target
[Service]
User=root
WorkingDirectory=/opt/MMDVMHost
ExecStart=/usr/bin/screen -S MMDVMHost -D -m /usr/local/bin/MMDVMHost
/home/pi/MMDVM.ini
ExecStop=/usr/bin/screen -S MMDVMHost -X quit
[Install]
WantedBy=multi-user.target
```

Le damos los permisos necesarios

```
sudo chmod 755 /lib/systemd/system/mmdvmhost.service
```

Recargamos los servicios y habilitamos el nuestro

```
sudo systemctl daemon-reload
sudo systemctl enable mmdvmhost.service
```

Arrancamos nuestro servicio

```
sudo systemctl start mmdvmhost.service
```

y verificamos si funciona

```
sudo systemctl status mmdvmhost.service
```

```

pi@raspberrypi:/opt/MMDVMHost $ sudo systemctl status mmdvmhost.service
● mmdvmhost.service - MMDVM Host Service
   Loaded: loaded (/lib/systemd/system/mmdvmhost.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2017-09-19 20:51:19 UTC; 5s ago
     Main PID: 15607 (screen)
    CGroup: /system.slice/mmdvmhost.service
            └─15607 /usr/bin/SCREEN -S MMDVMHost -D -m /usr/local/bin/MMDVMHost /home/pi/MMDVM.ini
              └─15609 /usr/local/bin/MMDVMHost /home/pi/MMDVM.ini

Sep 19 20:51:19 raspberrypi systemd[1]: Started MMDVM Host Service.
pi@raspberrypi:/opt/MMDVMHost $ ps -A | grep MMDVM
15609 pts/2    00:00:00 MMDVMHost
pi@raspberrypi:/opt/MMDVMHost $

```

Y FUNCIONA!!! Ya está configurado como servicio y tiene auto arranque tras un reinicio

ACTUALIZACIÓN DMRIds

Vamos a configurar la actualización de las Id de DMR

Editamos el fichero /opt/MMDVMHost/Linux/DMRIDUpdate.sh

Buscamos donde pone DMRIDFILE y lo dejamos así

```
DMRIDFILE=/opt/MMDVMHost/DMRIds.dat
```

Y ejecutamos el script de actualización

```
sudo /opt/MMDVMHost/linux/DMRIDUpdate.sh
```

Añadimos una entrada en el cron de forma que todos los días se actualicen las Ids, para ello ejecutamos

```
sudo crontab -e
```

y añadimos que lo ejecute todos los días el script de actualización

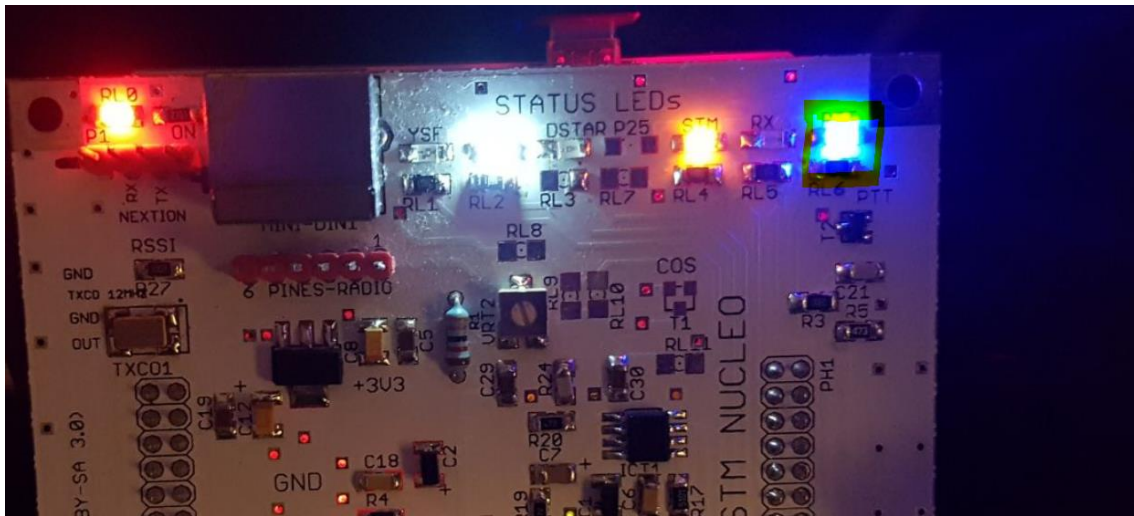
```
* 23 * * * /opt/MMDVMHost/linux/DMRIDUpdate.sh
```

Salvamos y listo, nuestro script de actualización se ejecutará todos los días a las 23hrs.

Verificamos que cuando haya tráfico se encienda el LED PTT

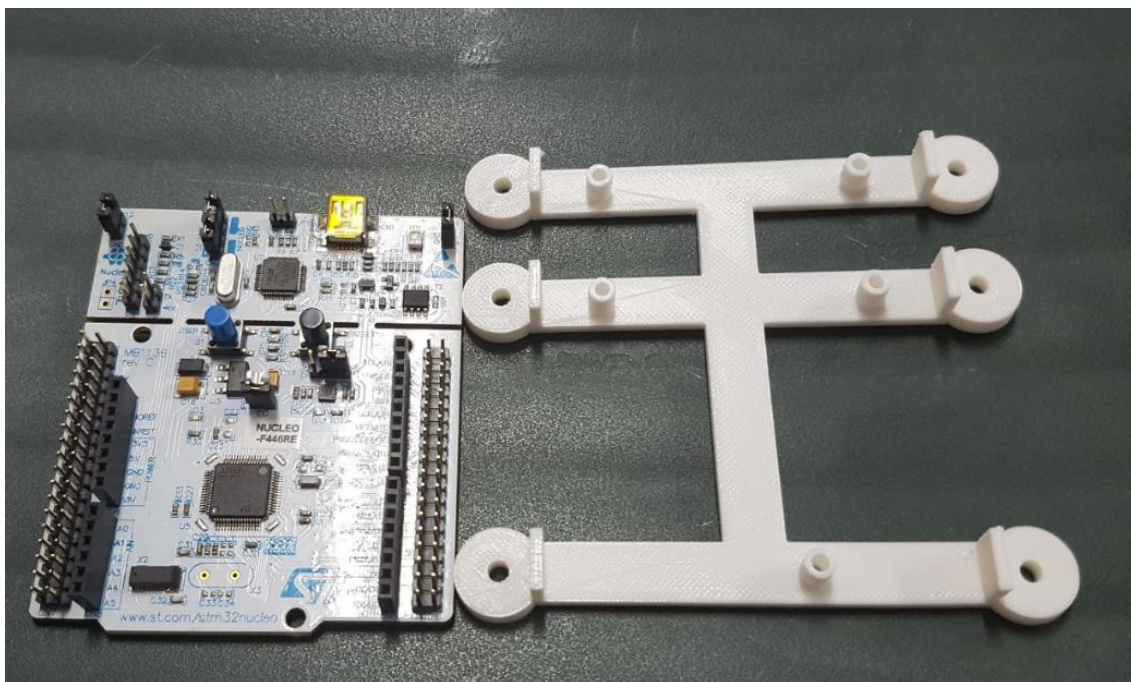
```
M: 2017-09-19 21:37:09.546 DMR, Logged into the master successfully
M: 2017-09-19 21:37:21.328 DMR Slot 1, received network voice header from 21
M: 2017-09-19 21:37:21.339 Debug: Mode set to DMR
M: 2017-09-19 21:37:22.028 DMR Slot 1, Embedded Talker Alias Header
M: 2017-09-19 21:37:22.029 0000: 04 00 4C 45 42 33 47 48 4E
M: 2017-09-19 21:37:22.734 DMR Slot 1, Embedded Talker Alias Block 1
M: 2017-09-19 21:37:22.734 0000: 05 00 20 45 64 75 61 72 64
M: 2017-09-19 21:37:23.449 DMR Slot 1, Embedded Talker Alias Block 2
M: 2017-09-19 21:37:23.450 0000: 06 00 6F 00 00 00 00 00 00
M: 2017-09-19 21:37:44.808 DMR Slot 1, received network end of voice transmi
M: 2017-09-19 21:37:49.890 Debug: Mode set to Idle
```

Eso es que hay tráfico y como vemos se enciende el led de PTT



CONFIGURACIÓN MODEM ST32M

Este es el modem que llevará nuestro repetidor



Se trata de un ST32 Nucleo

Siguiendo las instrucciones del colega F5UII compilamos MMDVM para la placa STM32 Nucleo (F446RE)

<https://www.f5uii.net/compilation-installation-configuration-mmdvm-stm32f4xx/>


```

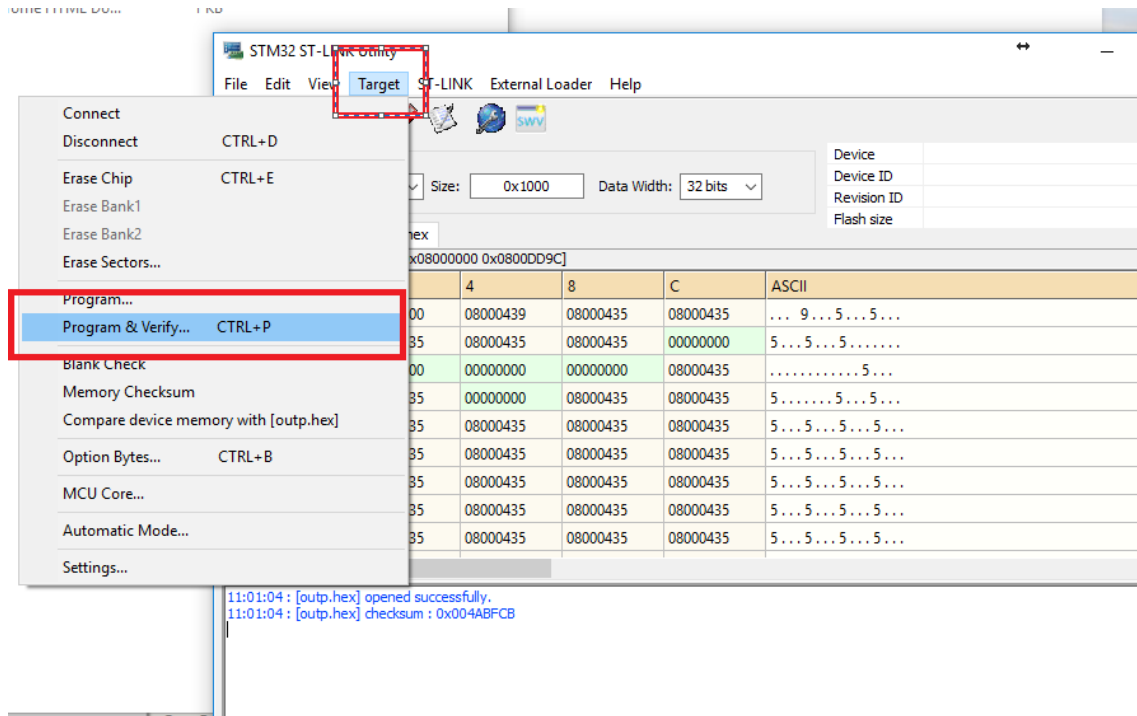
arm-none-eabi-g++ -Os -fno-exceptions -ffunction-sections -fdata-sections -fno-builtin -fno-rtti -DCUSTOM_NEW -DNO_EXCEPTIONS -c -mcpu-cortex-m4 -mthumb -mlittle-endian -mfpv4-sp-d16 -mfloat-abi-hard -mthumb-interwork -I. -ISTM32F4XX_Lib\CMSIS\Include/ -ISTM32F4XX_Lib\Device/ -ISTM32F4XX_Lib\STM32F4xx_StPeriph_Driver\include/ -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSTM32F446xx -DSTM32F4_NUCLEO -DHSE_VALUE=8000000 -DMADEBYMAKEFILE c:\temp\MMDVM\SerialRB.cpp -o c:\temp\MMDVM\SerialRB.o
Compiled "c:\temp\MMDVM\SerialRB.cpp"!
arm-none-eabi-g++ -Os -fno-exceptions -ffunction-sections -fdata-sections -fno-builtin -fno-rtti -DCUSTOM_NEW -DNO_EXCEPTIONS -c -mcpu-cortex-m4 -mthumb -mlittle-endian -mfpv4-sp-d16 -mfloat-abi-hard -mthumb-interwork -I. -ISTM32F4XX_Lib\CMSIS\Include/ -ISTM32F4XX_Lib\Device/ -ISTM32F4XX_Lib\STM32F4xx_StPeriph_Driver\include/ -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSTM32F446xx -DSTM32F4_NUCLEO -DHSE_VALUE=8000000 -DMADEBYMAKEFILE c:\temp\MMDVM\SerialSTM.cpp -o c:\temp\MMDVM\SerialSTM.o
Compiled "c:\temp\MMDVM\SerialSTM.cpp"!
arm-none-eabi-g++ -Os -fno-exceptions -ffunction-sections -fdata-sections -fno-builtin -fno-rtti -DCUSTOM_NEW -DNO_EXCEPTIONS -c -mcpu-cortex-m4 -mthumb -mlittle-endian -mfpv4-sp-d16 -mfloat-abi-hard -mthumb-interwork -I. -ISTM32F4XX_Lib\CMSIS\Include/ -ISTM32F4XX_Lib\Device/ -ISTM32F4XX_Lib\STM32F4xx_StPeriph_Driver\include/ -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSTM32F446xx -DSTM32F4_NUCLEO -DHSE_VALUE=8000000 -DMADEBYMAKEFILE c:\temp\MMDVM\SerialSTM\CMSIS.cpp -o c:\temp\MMDVM\SerialSTM\CMSIS.o
Compiled "c:\temp\MMDVM\SerialSTM\CMSIS.cpp"!
arm-none-eabi-g++ -Os -fno-exceptions -ffunction-sections -fdata-sections -fno-builtin -fno-rtti -DCUSTOM_NEW -DNO_EXCEPTIONS -c -mcpu-cortex-m4 -mthumb -mlittle-endian -mfpv4-sp-d16 -mfloat-abi-hard -mthumb-interwork -I. -ISTM32F4XX_Lib\CMSIS\Include/ -ISTM32F4XX_Lib\Device/ -ISTM32F4XX_Lib\STM32F4xx_StPeriph_Driver\include/ -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSTM32F446xx -DSTM32F4_NUCLEO -DHSE_VALUE=8000000 -DMADEBYMAKEFILE c:\temp\MMDVM\Utils.cpp -o c:\temp\MMDVM\Utils.o
Compiled "c:\temp\MMDVM\Utils.cpp"!
arm-none-eabi-g++ -Os -fno-exceptions -ffunction-sections -fdata-sections -fno-builtin -fno-rtti -DCUSTOM_NEW -DNO_EXCEPTIONS -c -mcpu-cortex-m4 -mthumb -mlittle-endian -mfpv4-sp-d16 -mfloat-abi-hard -mthumb-interwork -I. -ISTM32F4XX_Lib\CMSIS\Include/ -ISTM32F4XX_Lib\Device/ -ISTM32F4XX_Lib\STM32F4xx_StPeriph_Driver\include/ -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSTM32F446xx -DSTM32F4_NUCLEO -DHSE_VALUE=8000000 -DMADEBYMAKEFILE c:\temp\MMDVM\YSFRX.cpp -o c:\temp\MMDVM\YSFRX.o
Compiled "c:\temp\MMDVM\YSFRX.cpp"!
arm-none-eabi-g++ -Os -fno-exceptions -ffunction-sections -fdata-sections -fno-builtin -fno-rtti -DCUSTOM_NEW -DNO_EXCEPTIONS -c -mcpu-cortex-m4 -mthumb -mlittle-endian -mfpv4-sp-d16 -mfloat-abi-hard -mthumb-interwork -I. -ISTM32F4XX_Lib\CMSIS\Include/ -ISTM32F4XX_Lib\Device/ -ISTM32F4XX_Lib\STM32F4xx_StPeriph_Driver\include/ -DUSE_STDPERIPH_DRIVER -DSTM32F4XX -DSTM32F446xx -DSTM32F4_NUCLEO -DHSE_VALUE=8000000 -DMADEBYMAKEFILE c:\temp\MMDVM\YSFTX.cpp -o c:\temp\MMDVM\YSFTX.o
Compiled "c:\temp\MMDVM\YSFTX.cpp"!
arm-none-eabi-g++ c:\temp\MMDVM\system_stm32f1xx\startup_stm32f105xc.S c:\temp\MMDVM\STM32F4XX_Lib\Device\system_stm32f4xx.o c:\temp\MMDVM\STM32F4XX_Lib\Device\startup_stm32f4xx.o c:\temp\MMDVM\STM32F4XX_Lib\STM32F4xx_StPeriph_Driver\source\misc.o c:\temp\MMDVM\STM32F4XX_Lib\STM32F4xx_StPeriph_Driver\source\stm32f4xx_adc.o c:\temp\MMDVM\STM32F4XX_Lib\STM32F4xx_StPeriph_Driver\source\stm32f4xx_dac.o c:\temp\MMDVM\STM32F4XX_Lib\STM32F4xx_StPeriph_Driver\source\stm32f4xx_gpio.o c:\temp\MMDVM\STM32F4XX_Lib\STM32F4xx_StPeriph_Driver\source\stm32f4xx_rcc.o c:\temp\MMDVM\STM32F4XX_Lib\STM32F4xx_StPeriph_Driver\source\stm32f4xx_tim.o c:\temp\MMDVM\STM32F4XX_Lib\STM32F4xx_StPeriph_Driver\source\stm32f4xx_usart.o c:\temp\MMDVM\system_stm32f1xx.o c:\temp\MMDVM\CalDMR.o c:\temp\MMDVM\CalDStarRX.o c:\temp\MMDVM\CalDStarTX.o c:\temp\MMDVM\CalRSSI.o c:\temp\MMDVM\CWIDTX.o c:\temp\MMDVM\DMRDNRX.o c:\temp\MMDVM\DMRDNOTX.o c:\temp\MMDVM\DMRId1erX.o c:\temp\MMDVM\DMRRX.o c:\temp\MMDVM\DMRSIoTRX.o c:\temp\MMDVM\DMRSIoTType.o c:\temp\MMDVM\DMRTX.o c:\temp\MMDVM\DStarRX.o c:\temp\MMDVM\DStarTX.o c:\temp\MMDVM\IO.o c:\temp\MMDVM\IOdue.o c:\temp\MMDVM\IOSTM.o c:\temp\MMDVM\IOSTRX.o c:\temp\MMDVM\IOteensy.o c:\temp\MMDVM\MMDVM.o c:\temp\MMDVM\P2SRX.o c:\temp\MMDVM\P2STX.o c:\temp\MMDVM\RSSIRB.o c:\temp\MMDVM\SampleRB.o c:\temp\MMDVM\SerialArduino.o c:\temp\MMDVM\SerialPort.o c:\temp\MMDVM\SerialRB.o c:\temp\MMDVM\SerialSTM.o c:\temp\MMDVM\SerialSTM\CMSIS.o c:\temp\MMDVM\YSFRX.o c:\temp\MMDVM\YSFTX.o -Os -specs=nano.specs -T stm32f4xx_link.ld -mcpu-cortex-m4 -mthumb -mlittle-endian -mfpv4-sp-d16 -mfloat-abi-hard -mthumb-interwork --specs-nosys.specs STM32F4XX_Lib\CMSIS\Lib\GCC\libarm_cortexM4lf_math.a -o bin/outp.elf
Linking complete!
arm-none-eabi-size bin/outp.elf
  text  data  bss  dec  hex filename
 5988   744  4468  10800  189c0 bin/outp.elf
arm-none-eabi-objcopy -O ihex bin/outp.elf bin/outp.hex
Objcopy from ELF to IHX complete!
arm-none-eabi-objcopy -O binary bin/outp.elf bin/outp.bin
Objcopy from ELF to BINARY complete!
c:\temp\MMDVM>. \bin\make.exe nucleo

```

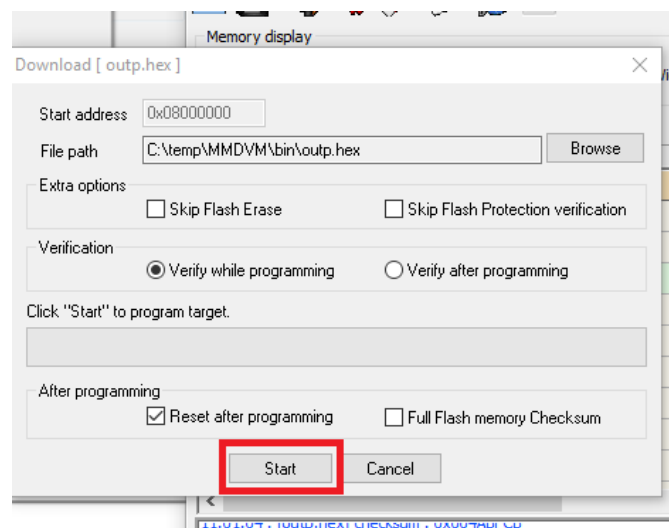
Una vez instalado el software para grabar en la board ST, según indican las instrucciones, seleccionamos el fichero que acabamos de compilar (out.hex)

The image shows two overlapping windows. The background window is the 'STM32 ST-LINK Utility' application, with its 'File' menu open, showing options like 'Open file...', 'Save file as...', and 'Exit'. The foreground window is a Windows Explorer window showing the file system path 'Este equipo > OS (C:) > temp > MMDVM > bin'. In this directory, there are three files: 'outp.bin', 'outp.elf', and 'outp.hex'. The 'outp.hex' file is selected, and the 'Nombre:' field at the bottom of the Explorer window contains 'outp.hex'.

Conectamos nuestra placa y le damos a programar y verificar



Pulsamos start y cruzamos los dedos :D :D



Ya tenemos el MODEM programado con el código MMDVM

```

11:03:03 : Connected via SWD.
11:03:03 : SWD Frequency = 4,0 MHz.
11:03:03 : Connection mode : Normal.
11:03:03 : Debug in Low Power mode enabled.
11:03:04 : Device ID:0x421
11:03:04 : Device flash Size : 512KBytes
11:03:04 : Device family :STM32F446xx
11:05:50 : Memory programmed in 3s and 750ms.
11:05:50 : Verification...OK
11:05:50 : Programmed memory Checksum: 0x004ABFCB
    
```

Debug in Low Power mode enabled. Device ID:0x421

CONEXIONADO EQUIPOS Y MODEM

Platine de filtrage MMDVM (F5UII)	Raccordement Postes Motorola MMDVM		Motorola GM340, GM350*, GM360	
	Broche PS2 N°	Signal	Couleur fil (câble PS2 shopchip)	RX Récepteur TX Emetteur
1	Data terminal ==> flat TX audio input	Rouge		pin 5
2	data ground	Orange	pin 7	pin 7
3	PTT TX input	Marron		pin 3 (GP1)
4	Data RX (flat) ==> terminal	Jaune	pin 11	
5	RX audio ==> terminal	Noir	NC	
6	sqelch radio output (COR)	Vert	pin 8 (GP3)	
blindage	Ground	non isolé		
Relier au + 12 V pour mise en marche du TRX sur retour tension (sans appui bouton)			pin 10	pin 10

2	4	6	8	10	12	14	16
1	3	5	7	9	11	13	15

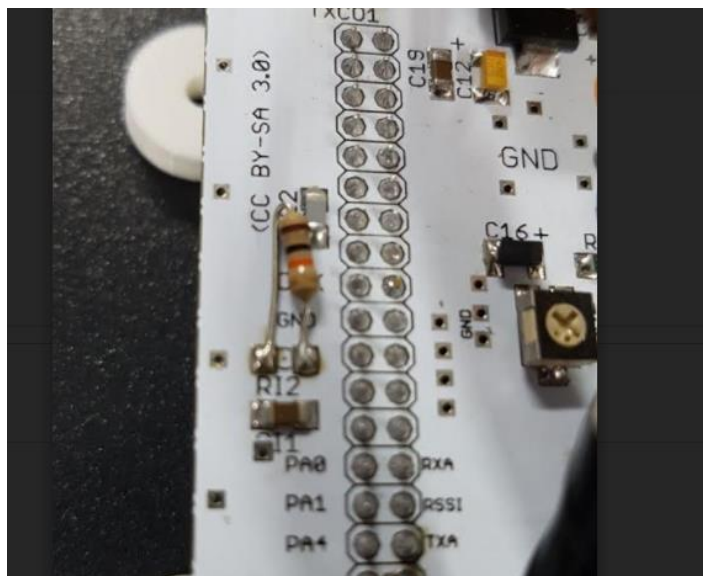
HLN9457a - Vue extérieure

PS2 - Vue extérieure

** Pour le GM350 4 canaux, le câblage sur la fiche accessoire arrière n'est pas réalisé. Une modification existe.*

Se ha realizado la conexión de los equipos conforme a la imagen superior. El pin para la señal RSSI es el pin 15.

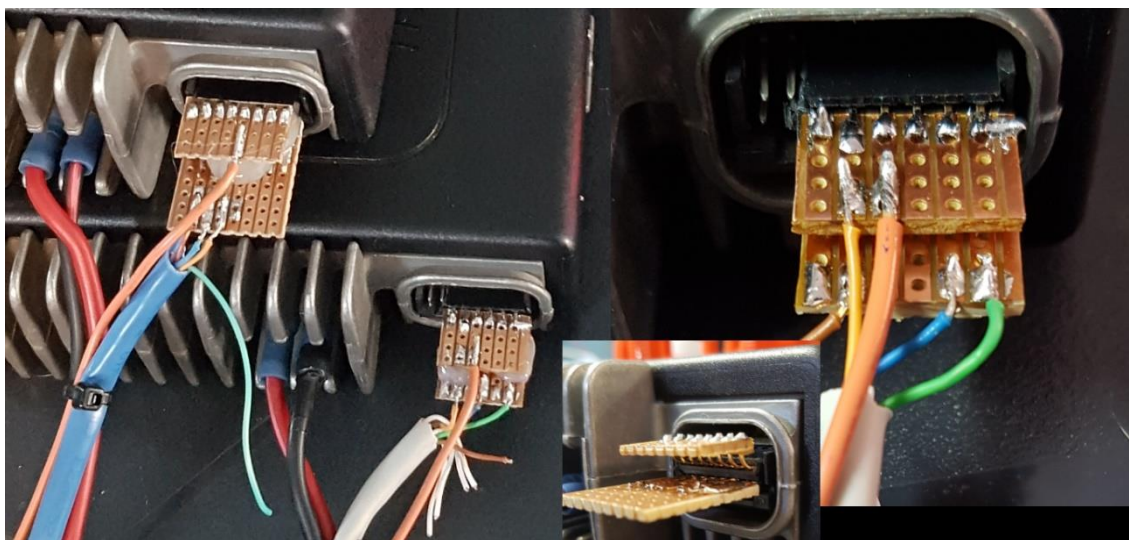
Respecto al RSSI hay que adaptar los voltajes para que la placa MMDVM no reciba mas de 3v que es su máximo tolerable. Para ello hemos añadido una resistencia de 10K a masa. Hay un punto para ello marcado como RI2



Esto hará que la tensión recibida del RSSI sea un 50% respecto a la original que envía la radio.

CONECTOR MOTOROLA HOMEMADE

No teníamos unos conectores de Motorola se han hecho unos caseros utilizando unos pines macho macho y unas PCB para construcciones caseras. El resultado es bastante bueno

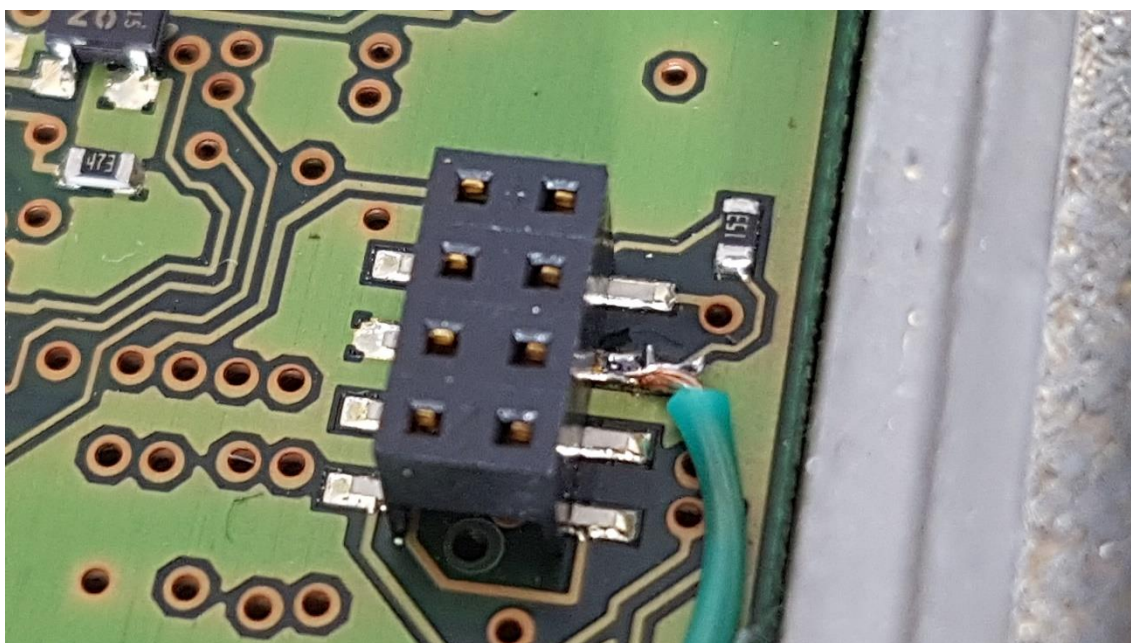
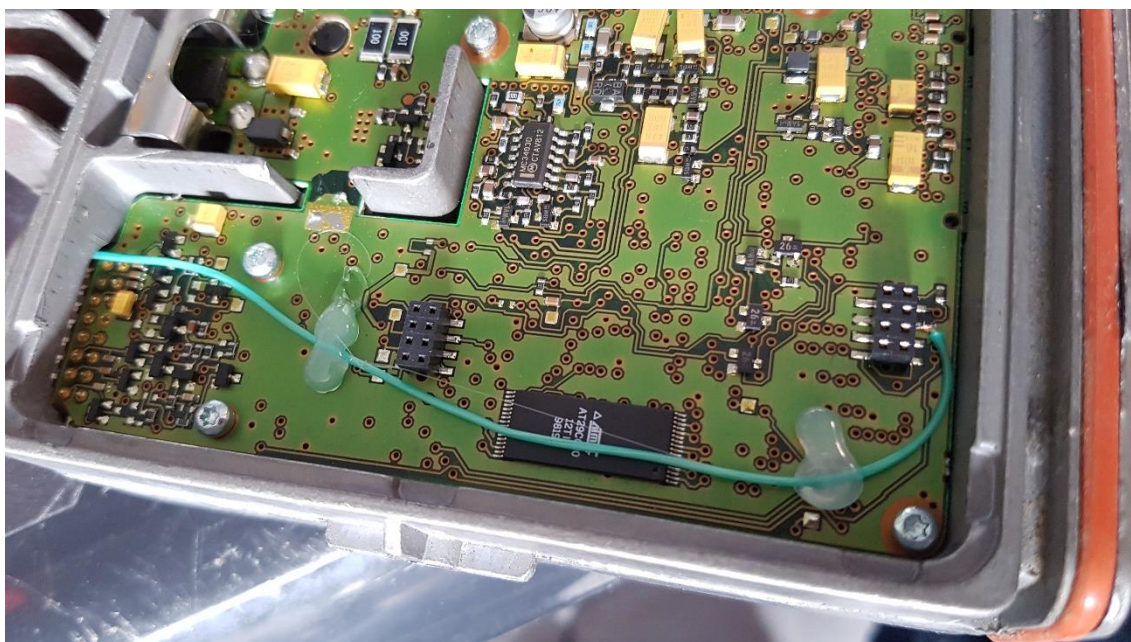


MODIFICACIÓN SALIDA AUDIO

Se realizo la modificación de la salida de audio ya que este modelo no saca el audio sin filtrar.

La información la encontramos aquí:

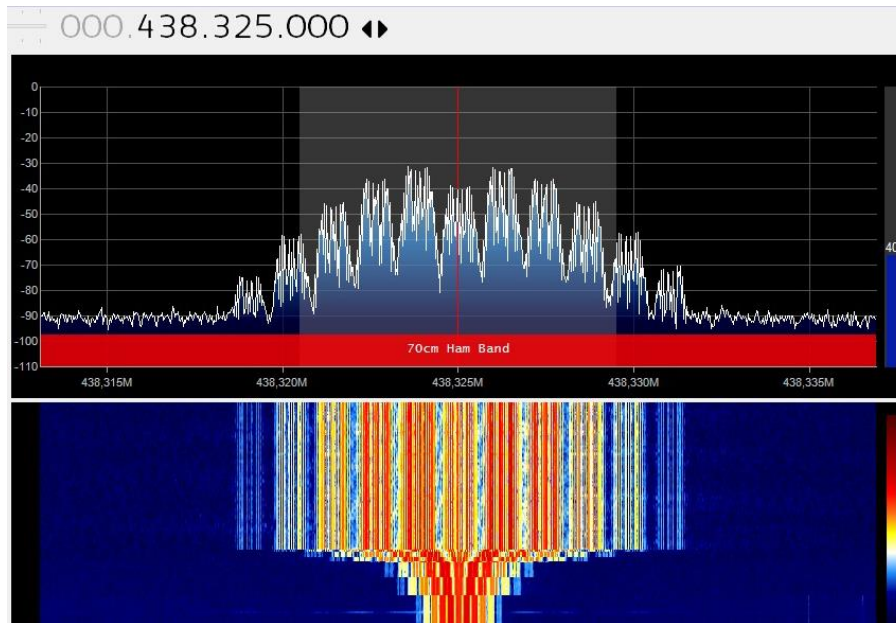
<http://sp4.jestok.com/5348/comment-page-1>



AJUSTE DEL EQUIPO TX

A falta de un analizador de espectro nos valemos de un airspy para ajustar el nivel de audio que le llegará a la Motorola en TX.

ANTES



DESPUES

