

## PROGRAMMATION DU STM32F446 à l'aide du Raspberry Pi

Sur votre ordinateur « **WINDOWS** » **(peut être effectué directement avec le RASPBERRY Pi)**

a- Télécharger les fichiers « zip » du code source MMDVM ici:

<https://github.com/g4klx/MMDVM>

b- Décompresser le fichier MMDVM-master.zip dans le répertoire MMDVM-master.

renommer le sous répertoire MMDVM

c- Télécharger les fichiers de la librairie STM32F4XX\_Lib.zip ici:

<https://groups.yahoo.com/neo/groups/mmdvm/files>

Décompresser le fichier \_STM32F4XX\_Lib dans son répertoire et copier le répertoire STM32F4XX dans le répertoire MMDVM du DD (ex : C:\MMDVM).

d- Modification du fichier config.h :

éditer le fichier config.h situé dans le répertoire MMDVM, faire les modifications

```
// Frequencies such as 10.0 Mhz (48000 * 208.333) or 20 Mhz
// (48000 * 416.666) are not suitable.
//
// For 12 MHz
#define EXTERNAL_OSC 12000000      (TXO incorporer dans la carte)
// For 12.288 MHz
// #define EXTERNAL_OSC 12288000
// For 14.4 MHz
// #define EXTERNAL_OSC 14400000
// For 19.2 MHz
// #define EXTERNAL_OSC 19200000

// Allow the use of the COS line to lockout the modem
// #define USE_COS_AS_LOCKOUT

// Use pins to output the current mode
#define ARDUINO_MODE_PINS          (mise en service des leds de status)

// For the original Arduino Due pin layout
// #define ARDUINO_DUE_PAPA

// For the ZUM V1.0 and V1.0.1 boards pin layout
// #define ARDUINO_DUE_ZUM_V10

// For the SP8NTH board
// #define ARDUINO_DUE_NTH

// For ST Nucleo-64 STM32F446RE board
#define STM32F4_NUCLEO_MORPHO_HEADER      (MMDVM TOUFIK)
// #define STM32F4_NUCLEO_ARDUINO_HEADER

// To use wider C4FSK filters for DMR, System Fusion and P25 on
// transmit
// #define WIDE_C4FSK_FILTERS_TX
// To use wider C4FSK filters for DMR, System Fusion and P25 on
// receive
// #define WIDE_C4FSK_FILTERS_RX

// Pass RSSI information to the host
// #define SEND_RSSI_DATA              (à modifier si besoin)

// Use the modem as a serial repeater for Nextion displays
#define SERIAL_REPEATER              (ecran Nextion)

#endif
```

e- à l'aide de WINSCP copier le répertoire MMDVM de votre DD sur la carte SD du Raspberry Pi  
dans le répertoire de votre choix : pour moi : /Applications/MMDVM d'autre /opt/MMDVM

## PREPARATION DE LA CARTE SD DU RASPBERRY PI

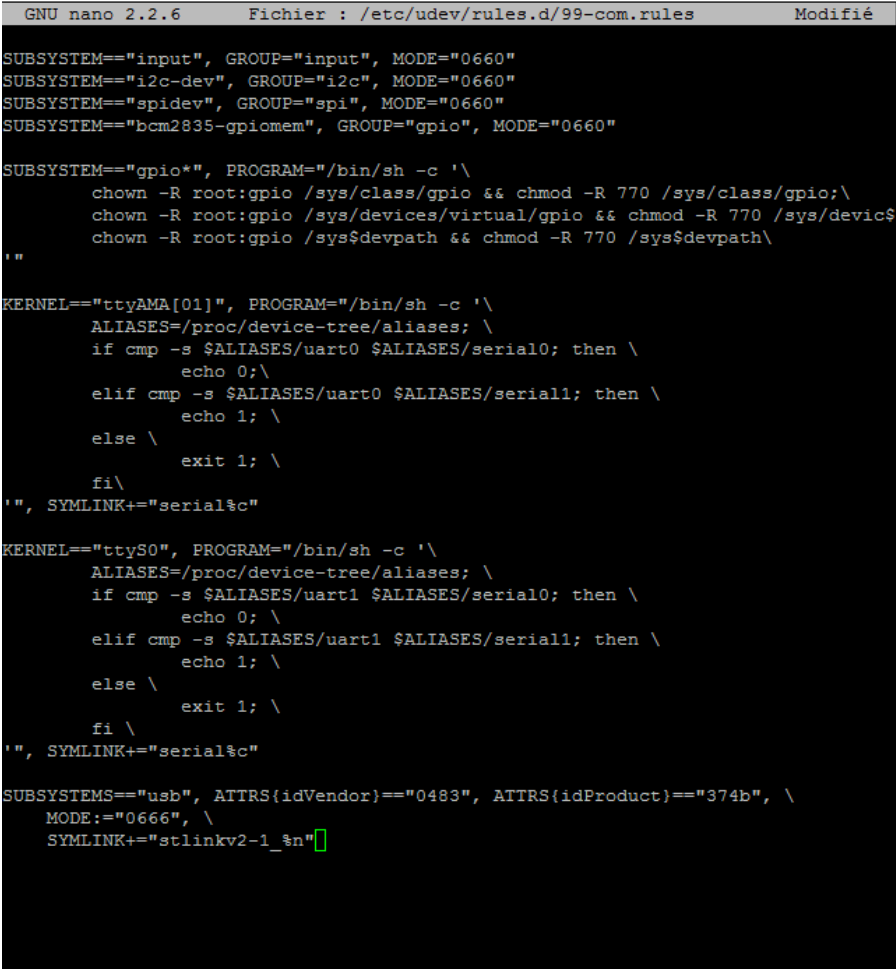
- a- sudo apt-get update
- b- sudo apt-get install gcc-arm-none-eabi gdb-arm-none-eabi -y
- c- sudo apt-get install git autoconf libtool make pkg-config libusb-1.0-0 libusb-1.0-0-dev -y
- d- git clone git://git.code.sf.net/p/openocd/code openocd-code
- e- cd openocd-code
- f- ./bootstrap
- g- ./configure --enable-sysfsgpio --enable-bcm2835gpio (prendre une tasse de café.....)
- h- make (prendre un bol de café.....)
- i- sudo make install

## MODIFICATIONS DES FICHIERS de la carte sd

- a- taper dans un terminal : « sudo nano /etc/udev/rules.d/99-com.rules »

ajouter à la fin:

```
SUBSYSTEMS=="usb", ATTRS{idVendor}=="0483", ATTRS{idProduct}=="374b", \
MODE=="0666", \
SYMLINK+="stlinkv2-1_%n"
```



```
GNU nano 2.2.6 Fichier : /etc/udev/rules.d/99-com.rules Modifié ^
SUBSYSTEM=="input", GROUP="input", MODE="0660"
SUBSYSTEM=="i2c-dev", GROUP="i2c", MODE="0660"
SUBSYSTEM=="spidev", GROUP="spi", MODE="0660"
SUBSYSTEM=="bcm2835-gpiomem", GROUP="gpio", MODE="0660"

SUBSYSTEM=="gpio*", PROGRAM="/bin/sh -c '\
chown -R root:gpio /sys/class/gpio && chmod -R 770 /sys/class/gpio;\
chown -R root:gpio /sys/devices/virtual/gpio && chmod -R 770 /sys/devic$
chown -R root:gpio /sys$devpath && chmod -R 770 /sys$devpath\
' "

KERNEL=="ttyAMA[01]", PROGRAM="/bin/sh -c '\
ALIASES=/proc/device-tree/aliases; \
if cmp -s $ALIASES/uart0 $ALIASES/serial0; then \
echo 0;\
elif cmp -s $ALIASES/uart0 $ALIASES/serial1; then \
echo 1; \
else \
exit 1; \
fi\
'", SYMLINK+="serial%c"

KERNEL=="ttyS0", PROGRAM="/bin/sh -c '\
ALIASES=/proc/device-tree/aliases; \
if cmp -s $ALIASES/uart1 $ALIASES/serial0; then \
echo 0; \
elif cmp -s $ALIASES/uart1 $ALIASES/serial1; then \
echo 1; \
else \
exit 1; \
fi \
'", SYMLINK+="serial%c"

SUBSYSTEMS=="usb", ATTRS{idVendor}=="0483", ATTRS{idProduct}=="374b", \
MODE=="0666", \
SYMLINK+="stlinkv2-1_%n"
```

b- taper dans un terminal : « sudo nano /usr/local/share/openocd/scripts/interface/stlink-v2.cfg »

modifier la ligne « hla\_vid\_pid 0x0483 0x3748 » par « hla\_vid\_pid 0x0483 0x374b »

```
GNU nano 2.2.6 Fichier : .../openocd/scripts/interface/stlink-v2.cfg Modifié
#
# STMicroelectronics ST-LINK/V2 in-circuit debugger/programmer
#
interface hla
hla_layout stlink
hla_device_desc "ST-LINK/V2"
hla_vid_pid 0x0483 0x374b

# Optionally specify the serial number of ST-LINK/V2 usb device. ST-LINK/V2
# devices seem to have serial numbers with unreadable characters. ST-LINK/V2
# firmware version >= V2.J21.S4 recommended to avoid issues with adapter serial
# number reset issues.
# eg.
#hla_serial "\xaa\xbc\x6e\x06\x50\x75\xff\x55\x17\x42\x19\x3f"
```

**FAIRE UN REBOOT DU SYSTEME**

## **PROGRAMMATION DU FIRMWARE DE LA CARTE STM32F446**

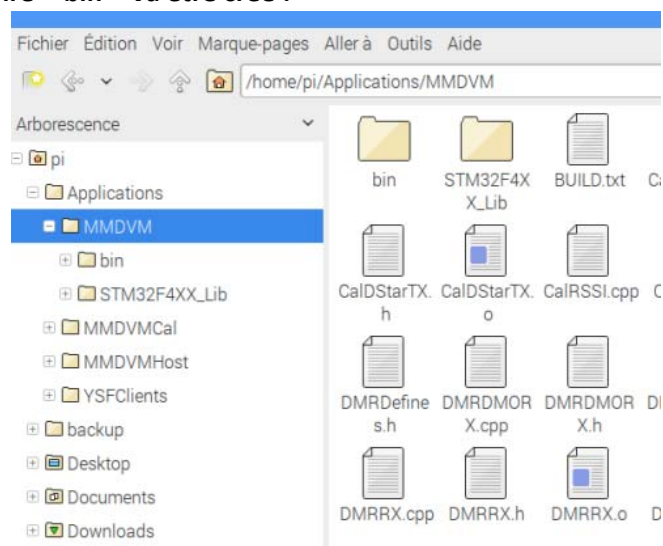
**A - Maintenant construisons le firmware à injecter dans le STM32F446**

Dans un terminal taper « make clean » « entrée »

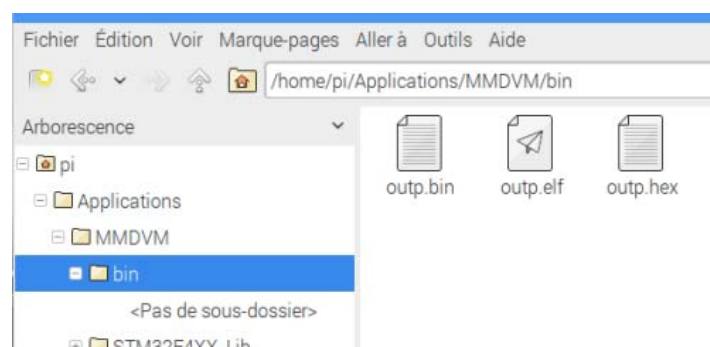
Puis taper « make nucleo » « entrée »

laisser faire la construction dans l'écran du terminal

**B - un répertoire « bin » va être créé :**



Dans le répertoire « bin » les fichiers vont être créés



```

arm-none-eabi-g++ -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_StdPeriph_Driver/include/ -DUSE_STDPERIPH
_DRIVER -DSTM32F4XX -DSTM32F446xx -DSTM32F4_NUCLE0 -DHSE_VALUE=8000000 -Os -fno-
exceptions -ffunction-sections -fdata-sections -fno-builtin -fno-rtti -DCUSTOM_M
EW -DNO_EXCEPTIONS IODue.cpp -o IODue.o
Compiled IODue.cpp!

arm-none-eabi-g++ -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_StdPeriph_Driver/include/ -DUSE_STDPERIPH
_DRIVER -DSTM32F4XX -DSTM32F446xx -DSTM32F4_NUCLE0 -DHSE_VALUE=8000000 -Os -fno-
exceptions -ffunction-sections -fdata-sections -fno-builtin -fno-rtti -DCUSTOM_M
EW -DNO_EXCEPTIONS CalDStarRX.cpp -o CalDStarRX.o
Compiled CalDStarRX.cpp!

arm-none-eabi-g++ ./STM32F4XX_Lib/STM32F4xx_StdPeriph_Driver/source/stm32f4xx_t
im.o ./STM32F4XX_Lib/STM32F4xx_StdPeriph_Driver/source/stm32f4xx_usart.o ./STM32
F4XX_Lib/STM32F4xx_StdPeriph_Driver/source/stm32f4xx_adc.o ./STM32F4XX_Lib/STM32
F4xx_StdPeriph_Driver/source/misc.o ./STM32F4XX_Lib/STM32F4xx_StdPeriph_Driver/s
ource/stm32f4xx_dac.o ./STM32F4XX_Lib/STM32F4xx_StdPeriph_Driver/source/stm32f4x
x_rcc.o ./STM32F4XX_Lib/STM32F4xx_StdPeriph_Driver/source/stm32f4xx_gpio.o ./STM
32F4XX_Lib/Device/system_stm32f4xx.o ./STM32F4XX_Lib/Device/startup/startup_stm3
2f4xx.o ./IO.o ./CalDMR.o ./RSSIR.o ./SampleRB.o ./SerialPort.o ./CalRSSI.o ./D
MRRX.o ./DStarRX.o ./DMRTX.o ./CwIDTX.o ./SerialSTM.o ./DMRDMORX.o ./Utils.o ./I
OSTM.o ./YSFTX.o ./MMDVM.o ./P2STX.o ./P2SRX.o ./SerialArduino.o ./IOTeensy.o ./
DMRIdleRX.o ./DMRSLOTType.o ./YSFRX.o ./DMRSLOTRX.o ./SerialRB.o ./DMRDMOTX.o ./
CalDStarTX.o ./DStarTX.o ./IODue.o ./CalDStarRX.o -T stm32f4xx_link.ld -mcpu=cor
tex-m4 -mthumb -mlittle-endian -mfpv4-sp-d16 -mfloat-abi=hard -mthumb-interw
ork --specs=nosys.specs STM32F4XX_Lib/CMSIS/Lib/GCC/libarm_cortexM4lf_math.a -O
s --specs=nano.specs -o bin/outp.elf
Linking complete!

arm-none-eabi-size bin/outp.elf
   text    data     bss     dec     hex filename
   50716    760    30452    81928    14008 bin/outp.elf
arm-none-eabi-objcopy -O ihex bin/outp.elf bin/outp.hex
Objcopy from ELF to IHEX complete!

arm-none-eabi-objcopy -O binary bin/outp.elf bin/outp.bin
Objcopy from ELF to BINARY complete!

pi@raspberrypi:~/Applications/MMDVM $

```

c - Dans un terminal sous le répertoire MMDVM

Taper « make clean »

Puis taper « make deploy »

```

pi@raspberrypi:~/Applications/MMDVM $ make deploy
/usr/local/bin/openocd -f /usr/local/share/openocd/scripts/board/stm32f4discover
y.cfg -c "program bin/outp.elf verify reset exit"
Open On-Chip Debugger 0.10.0-rc2-dev-00001-g1c4aa20 (2017-01-18-17:33)
Licensed under GNU GPL v2
For bug reports, read
    http://openocd.org/doc/doxygen/bugs.html
Info : The selected transport took over low-level target control. The results mi
ght differ compared to plain JTAG/SWD
adapter speed: 2000 kHz
adapter_nsrst_delay: 100
none separate
srst_only separate srst_nogate srst_open_drain connect_deassert_srst
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
Info : Unable to match requested speed 2000 kHz, using 1800 kHz
Info : clock speed 1800 kHz
Info : STLINK v2 JTAG v19 API v2 SWIM v3 VID 0x0483 PID 0x374B
Info : using stlink api v2
Info : Target voltage: 3.256011
Info : stm32f4x.cpu: hardware has 6 breakpoints, 4 watchpoints
adapter speed: 2000 kHz
target halted due to debug-request, current mode: Thread
xPSR: 0x01000000 pc: 0x080021f8 msp: 0x20020000
adapter speed: 8000 kHz
** Programming Started **
auto erase enabled
Info : device id = 0x10006421
Info : flash size = 512kbytes
target halted due to breakpoint, current mode: Thread
xPSR: 0x61000000 pc: 0x20000046 msp: 0x20020000
wrote 65536 bytes from file bin/outp.elf in 3.010412s (21.260 KiB/s)
** Programming Finished **
** Verify Started **
target halted due to breakpoint, current mode: Thread
xPSR: 0x61000000 pc: 0x2000002e msp: 0x20020000
target halted due to breakpoint, current mode: Thread
xPSR: 0x61000000 pc: 0x2000002e msp: 0x20020000
verified 51476 bytes in 0.342650s (146.708 KiB/s)
** Verified OK **
** Resetting Target **
adapter speed: 2000 kHz
shutdown command invoked
pi@raspberrypi:~/Applications/MMDVM $

```

## PROGRAMMATION DU FICHIER MMDVM.ini

Si nécessaire pour paramétrer un écran Nextion

Editer le fichier MMDVM.ini :

Modification : « Port=modem »

```
[Nextion]
Port=modem
# Port=/dev/ttyAMA0
Brightness=50
DisplayClock=1
UTC=0
IdleBrightness=20
```

**Relancer MMDVMHost (Reboot)**

**BON AMUSEMENT**

**F1IZL**