

Build instructions for MMDVM firmware using STM32F4XX or STM32F7XX

Andy CA6JAU, Jim KI6ZUM

November 11, 2020

Contents

1	Raspberry Pi	1
1.1	Nucleo-64 F446RE	1
1.2	Nucleo-144 F767ZI	2
1.3	MMDVM-Pi	3
1.3.1	Enable serial port in Raspberry Pi 3 or Pi Zero W	3
1.3.2	Installation of necessary software (only once)	4
1.3.3	Firmware compilation for MMDVM-Pi	4
2	Linux Ubuntu	5
3	Windows versions < 10	6
4	Windows 10 with Ubuntu	7
5	macOS	8
6	Pin definitions for different STM32 boards	9
6.1	Pin definitions for STM32F4 Discovery Board:	9
6.2	Pin definitions for Nucleo-64 F446RE boards (Morpho header): .	10
6.3	Pin definitions for Nucleo-64 F446RE boards (Arduino header): .	11
6.4	Pin definitions for MMDVM-Pi:	11
6.5	Pin definitions for Nucleo-144 F767ZI boards (Morpho header): .	12
7	Final notes	12

1 Raspberry Pi

1.1 Nucleo-64 F446RE

- If you are using Pi-Star, expand filesystem (if you haven't done before):

```
sudo pistar-expand  
sudo reboot
```

- Enable RW filesystem if you are using Pi-Star:

```
rpi-rw
```

- Install toolchain and necessary packages:

```
sudo apt-get install git gcc-arm-none-eabi gdb-arm-none-eabi libstdc++-arm-  
none-eabi-newlib autoconf libtool pkg-config libusb-1.0-0 libusb-  
1.0-0-dev
```

- Install OpenOCD:

```
git clone https://github.com/ntfreak/openocd  
cd openocd  
./bootstrap  
./configure  
make  
sudo make install
```

- Download the sources:

```
git clone https://github.com/g4klx/MMDVM  
cd MMDVM  
git submodule init  
git submodule update
```

- Edit Config.h:

```
nano Config.h
```

- Usually you could enable (for Morpho connector):

```
#define MODE_LEDS  
#define STM32F4_NUCLEO_MORPHO_HEADER  
#define SEND_RSSI_DATA  
#define SERIAL_REPEATER  
#define USE_DCBLOCKER  
#define USE_ALTERNATE_POCSAG_LEDS
```

- Compile the code:

```
make nucleo
```

- If you are using Pi-Star, stop services:

```
sudo pistar-watchdog.service stop  
sudo systemctl stop mmdvmhost.timer  
sudo systemctl stop mmdvmhost.service
```

- Upload the firmware:

```
sudo make deploy
```

1.2 Nucleo-144 F767ZI

- If you are using Pi-Star, expand filesystem (if you haven't done before):

```
sudo pistar-expand  
sudo reboot
```

- Install toolchain and necessary packages:

```
sudo apt-get install git gcc-arm-none-eabi gdb-arm-none-eabi libstdc++-arm-  
none-eabi-newlib autoconf libtool pkg-config libusb-1.0-0 libusb-  
1.0-0-dev
```

- Install OpenOCD:

```
git clone https://github.com/ntfreak/openocd  
cd openocd  
./bootstrap  
./configure  
make  
sudo make install
```

- Download the sources:

```
git clone https://github.com/g4klx/MMDVM  
cd MMDVM  
git submodule init  
git submodule update
```

- Edit Config.h according your preferences:

```
nano Config.h
```

- Usually you could enable:

```
#define MODE_LEDS  
#define SEND_RSSI_DATA  
#define SERIAL_REPEATER  
#define USE_DCBLOCKER  
#define USE_ALTERNATE_POCSAG_LEDS
```

- Compile the code:

```
make f767
```

- If you are using Pi-Star, stop services:

```
sudo pistar-watchdog.service stop  
sudo systemctl stop mmdvmhost.timer  
sudo systemctl stop mmdvmhost.service
```

- Upload the firmware:

```
sudo make deploy-f7
```

1.3 MMDVM-Pi

1.3.1 Enable serial port in Raspberry Pi 3 or Pi Zero W

This is necessary only if you are installing a fresh copy of Raspbian OS. Images like Pi-Star are already OK.

- Edit /boot/cmdline.txt:

```
sudo nano /boot/cmdline.txt
```

(remove the text: console=serial0,115200)

- Disable services:

```
sudo systemctl disable serial-getty@ttyAMA0.service
sudo systemctl disable bluetooth.service
```

- Edit /boot/config.txt:

```
sudo nano /boot/config.txt
```

and add the following lines at the end of /boot/config.txt:

```
enable_uart=1
dtoverlay=pi3-disable-bt
```

- Reboot your RPi:

```
sudo reboot
```

1.3.2 Installation of necessary software (only once)

- If you are using Pi-Star, expand filesystem (if you haven't done before):

```
sudo pi-star-expand
sudo reboot
```

- Update package lists:

```
sudo apt-get update
```

- Enable RW filesystem if you are using Pi-Star:

```
rpi-rw
```

- Install toolchain and necessary packages:

```
sudo apt-get install git gcc-arm-none-eabi gdb-arm-none-eabi libstdc++-arm-
none-eabi-newlib autoconf libtool pkg-config libusb-1.0-0 libusb
-1.0-0-dev
```

- Remove libi2c-dev and stm32flash packages if you are using Pi-Star:

```
sudo apt-get remove libi2c-dev
sudo apt-get remove stm32flash
```

- Install the latest stm32flash:

```
cd ~
git clone https://git.code.sf.net/p/stm32flash/code stm32flash
cd stm32flash
make
sudo make install
```

- Another place to get the latest stm32flash is:

<https://sourceforge.net/projects/stm32flash/files/>

1.3.3 Firmware compilation for MMDVM-Pi

- Download firmware sources:

```
cd ~
git clone https://github.com/g4klx/MMDVM
cd MMDVM
git submodule init
git submodule update
```

- Edit Config.h according your preferences:

```
nano Config.h
```

- You can select for example:

```
// #define EXTERNAL_OSC 12000000 (disable any external TCXO)
// #define ARDUINO_DUE_ZUM_V10 (this option doesn't matter for STM32 devices)
#define MODE_LEDS
#define SEND_RSSI_DATA
#define SERIAL_REPEATER
#define USE_DCBLOCKER
#define USE_ALTERNATE_POCSAG_LEDS
```

- Compile:

```
make pi
```

- If you are using Pi-Star, stop services:

```
sudo pistar-watchdog.service stop
sudo systemctl stop mmdvmhost.timer
sudo systemctl stop mmdvmhost.service
```

- Upload the firmware:

```
sudo make deploy-pi
```

2 Linux Ubuntu

- Remove the official package:

```
sudo apt-get purge binutils-arm-none-eabi gcc-arm-none-eabi gdb-arm-none-eabi libstdc++-arm-none-eabi-newlib libnewlib-arm-none-eabi
```

- Add 3rd party repository:

```
sudo add-apt-repository ppa:team-gcc-arm-embedded/ppa
sudo apt-get update
```

- Check the GCC package version in the PPA repository:

```
sudo apt-cache policy gcc-arm-embedded
```

- Install software requirements:

```
sudo apt-get install build-essential git gcc-arm-embedded qemu-system-arm
symlinks expect autoconf libtool pkg-config libusb-1.0-0 libusb-1.0-0-dev
```

- Install latest openOCD tool from the sources:

```
git clone https://github.com/ntfreak/openocd
cd openocd
./bootstrap
./configure
make
sudo make install
```

- Get the latest source code from GitHub:

```
git clone https://github.com/g4klx/MMDVM
cd MMDVM
git submodule init
git submodule update
```

- Clean the directory before build:

```
make clean
```

- Edit Config.h according your preferences.

- Then you can use the following commands to build the source code for the Nucleo-64 F446RE board:

```
make nucleo
```

or for the STM32F4 Discovery board:

```
make dis
```

or for the MMDVM-Pi board:

```
make pi
```

or for the Nucleo-144 F767ZI board:

```
make f767
```

- Upload the firmware using OpenOCD (USB, internal ST-Link), for STM32F4 family:

```
sudo make deploy
```

or for STM32F7 family:

```
sudo make deploy-f7
```

3 Windows versions < 10

- Download and install the ST-Link programming software from here (note: you will have to register with ST to be able to download the installer): <http://www.st.com/en/embedded-software/stsw-link004.html>
- Download and install Git for Windows (use default options): <https://git-scm.com/download/win>

- Download the GNU ARM embedded toolchain from here: <https://launchpad.net/gcc-arm-embedded/+download>

Install the GNU ARM tools in the default location.

- Launch the "GCC Command Prompt" from "GNU Tools for ARM Embedded Processors" (Start Menu).
- Get the latest source code from GitHub (change USERNAME for a valid Windows user name):

```
cd C:\Users\USERNAME\  
git clone https://github.com/g4klx/MMDVM  
cd MMDVM  
git submodule init  
git submodule update
```

- Download the GNU make utility: <http://gnuwin32.sourceforge.net/packages/make.htm>

Download the binaries zip file and extract make.exe and put it in the same directory as Makefile.

Download the dependencies zip file and extract libintl3.dll and libiconv2.dll and put them in the same directory as Makefile.

- Clean the directory before build:

```
make clean
```

- Edit Config.h according your preferences.
- Then you can use the following commands to build the source code for the Nucleo-64 F446RE board:

```
make nucleo
```

or for the STM32F4 Discovery board:

```
make dis
```

or for the MMDVM-Pi board:

```
make pi
```

or for the Nucleo-144 F767ZI board:

```
make f767
```

The .hex file will be in the bin folder.

4 Windows 10 with Ubuntu

- Download and install the ST-Link programming software from here (note: you will have to register with ST to be able to download the installer): <http://www.st.com/en/embedded-software/stsw-link004.html>
- Install bash using these instructions: <http://www.pcworld.com/article/3106463/windows/how-to-get-bash-on-windows-10-with-the-anniversary-update.html>
- Once you have bash installed, install GCC for ARM. It must be \geq version 4.9:

```
sudo apt-get install gcc
sudo apt-get install make
sudo apt-get remove gcc-arm-none-eabi
sudo add-apt-repository ppa:team-gcc-arm-embedded/ppa
sudo apt-get install -y git gcc-arm-embedded=5-2015q4-1~trusty1
```

- Make sure git is installed. If not install it with:

```
sudo apt-get install git
```

- Get the latest source code from GitHub:

```
git clone https://github.com/g4klx/MMDVM
cd MMDVM
git submodule init
git submodule update
```

- Clean the directory before build:

```
make clean
```

- Edit Config.h according your preferences.
- Then you can use the following commands to build the source code for the Nucleo-64 F446RE board:

```
make nucleo
```


or for the STM32F4 Discovery board:

```
make dis
```

or for the MMDVM-Pi board:

```
make pi
```

or for the Nucleo-144 F767ZI board:

```
make f767
```

The .hex file will be in the bin folder.

5 macOS

- First install Homebrew (not root!):

```
/usr/bin/ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

- Install additional software:

```
brew install libusb autogen automake wget pkg-config cmake openocd
```

- Install the ARM GCC toolchain:

```
wget https://launchpad.net/gcc-arm-embedded/5.0/5-2016-q3-update/+download/  
gcc-arm-none-eabi-5_4-2016q3-20160926-mac.tar.bz2  
tar xvf gcc-arm-none-eabi-5_4-2016q3-20160926-mac.tar.bz2  
cd gcc-arm-none-eabi-5_4-2016q3/  
sudo cp -Rf * /usr/local/
```

- Get the latest source code from GitHub:

```
git clone https://github.com/g4klx/MMDVM  
cd MMDVM  
git submodule init  
git submodule update
```

- Clean the directory before build:

```
make clean
```

- Edit Config.h according your preferences.

- Then you can use the following commands to build the source code for the Nucleo-64 F446RE board:

```
make nucleo
```

or for the STM32F4 Discovery board:

```
make dis
```

or for the MMDVM-Pi board:

```
make pi
```

or for the Nucleo-144 F767ZI board:

```
make f767
```

- Upload the firmware using OpenOCD (USB, internal ST-Link), for STM32F4 family:

```
make deploy
```

or for STM32F7 family:

```
make deploy-f7
```

6 Pin definitions for different STM32 boards

6.1 Pin definitions for STM32F4 Discovery Board:

PTT	PB13	output	P1 Pin37
COSLED	PA7	output	P1 Pin17
LED	PD15	output	P1 Pin47
COS	PA5	input	P1 Pin15
DSTAR	PD12	output	P1 Pin44
DMR	PD13	output	P1 Pin45
YSF	PD14	output	P1 Pin46
P25	PD11	output	P1 Pin43
RX	PA0	analog input	P1 Pin12
RSSI	PA1	analog input	P1 Pin11
TX	PA4	analog output	P1 Pin16
EXT_CLK	PA15	input	P2 Pin40
- Host communication:			
USART3 - TXD PC10 - RXD PC11			
- Serial repeater (3.3 V):			
UART5 - TXD PC12 - RXD PD2			

6.2 Pin definitions for Nucleo-64 F446RE boards (Morpho header):

PTT	PB13	output	CN10 Pin30
COSLED	PB14	output	CN10 Pin28
LED	PA5	output	CN10 Pin11
COS	PB15	input	CN10 Pin26
DSTAR	PB10	output	CN10 Pin25
DMR	PB4	output	CN10 Pin27

YSF	PB5	output	CN10 Pin29
P25	PB3	output	CN10 Pin31
MDSTAR	PC4	output	CN10 Pin34
MDMR	PC5	output	CN10 Pin6
MYSF	PC2	output	CN7 Pin35
MP25	PC3	output	CN7 Pin37
RX	PA0	analog input	CN7 Pin28
RSSI	PA1	analog input	CN7 Pin30
TX	PA4	analog output	CN7 Pin32
EXT_CLK	PA15	input	CN7 Pin17

- Host communication:
 USART2 - TXD PA2 - RXD PA3 (already connected with ST-Link VCP)
 - Serial repeater (3.3 V):
 UART5 - TXD PC12 - RXD PD2

6.3 Pin definitions for Nucleo-64 F446RE boards (Arduino header):

PTT	PB10	output	CN9 Pin7
COSLED	PB3	output	CN9 Pin4
LED	PB5	output	CN9 Pin5
COS	PB4	input	CN9 Pin6
DSTAR	PA1	output	CN8 Pin2
DMR	PA4	output	CN8 Pin3
YSF	PB0	output	CN8 Pin4
P25	PC1	output	CN8 Pin5
RX	PA0	analog input	CN8 Pin1
RSSI	PC0	analog input	CN8 Pin6
TX	PA5	analog output	CN5 Pin6
EXT_CLK	PB8	input	CN5 Pin10

- Host communication:
 USART2 - TXD PA2 - RXD PA3 (already connected with ST-Link VCP)
 - Serial repeater (3.3 V):
 USART1 - TXD PA9 - RXD PA10

6.4 Pin definitions for MMDVM-Pi:

PTT	PB13	output
-----	------	--------

COSLED	PB14	output
LED	PB15	output
COS	PC0	input
DSTAR	PC7	output
DMR	PC8	output
YSF	PA8	output
P25	PC9	output
RX	PA0	analog input
RSSI	PA7	analog input
TX	PA4	analog output
EXT_CLK	PA15	input
- Host communication:		
USART1 - TXD PA9 - RXD PA10		
- Serial repeater (3.3 V):		
UART5 - TXD PC12 - RXD PD2		

6.5 Pin definitions for Nucleo-144 F767ZI boards (Morpho header):

PTT	PB13	output	CN12 Pin30
COSLED	PB14	output	CN12 Pin28
LED	PA5	output	CN12 Pin11
COS	PB15	input	CN12 Pin26
DSTAR	PB10	output	CN12 Pin25
DMR	PB4	output	CN12 Pin27
YSF	PB5	output	CN12 Pin29
P25	PB3	output	CN12 Pin31
MDSTAR	PC4	output	CN12 Pin34
MDMR	PC5	output	CN12 Pin6
MYSF	PC2	output	CN11 Pin35
MP25	PC3	output	CN11 Pin37
RX	PA0	analog input	CN11 Pin28
RSSI	PA1	analog input	CN11 Pin30
TX	PA4	analog output	CN11 Pin32
EXT_CLK	PA15	input	CN11 Pin17

- Host communication:
 USART3 - TXD PD8 - RXD PD9 (already connected with ST-Link VCP)

- Serial repeater (3.3 V):
UART5 - TXD PC12 - RXD PD2

7 Final notes

- You will find more instructions here: https://github.com/juribeparada/MMDVM_man
- The source code of this document (LaTeX) is here: https://github.com/juribeparada/MMDVM_man/blob/master/STM32_MMDVM/mmdvm_stm32_build.tex