HVDN Communicator

Linux Distro Build Guide

Hudson Valley Digital Network

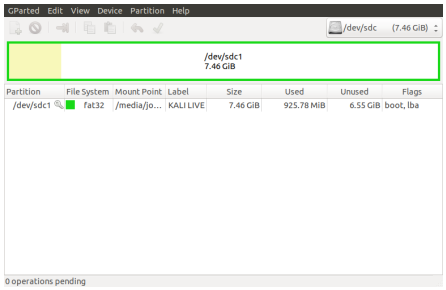
3 January 2020

v0.6

Introduction

This is a guide on how to build the distribution image for the HVDN Communicator. All reference to “RPi” in this guide are an abbreviation for the Raspberry Pi Zero W hardware. The desktop operating system used in preparing the build is assumed to be a Debian or Debian derivative distribution. Ubuntu 18.04 was use din the preparation of this guide.

Preparing Media

• From your PC mount an 8GB SD Card • Run **sudo gparted** partitioning software • From Gparted menu select **Devices** and SD card inserted • Any existing partitions on the SD card right click and select **Delete** • Right click on **unallocated**, select **new**, select **FAT32** as file system. Click **add** • Click **check mark** to apply all operations • Note only up to 8GB of the SD Card will be formatted	

• Download Raspbian and Extract from ZIP file
• Install Raspbian on freshly formatted SD card
	`wget --max-redirect=3 https://downloads.raspberrypi.org/raspbian_lite_latest`
	
	`unzip raspbian_lite_latest`
	
	`sudo dd bs=4M if=2019-09-26-raspbian-buster-lite.img of=/dev/sdg conv=fsync`

Connectivity

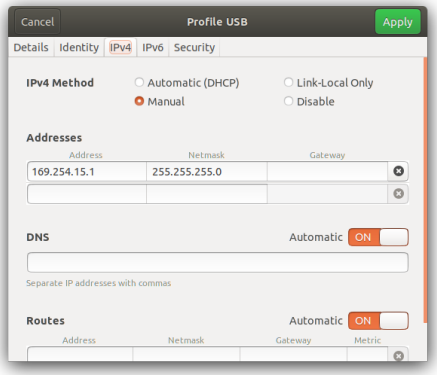
Via connected keyboard, monitor and mouse	

• Change to your home directory, run **sync** as sudo, and unmount the SD card
• Remove the SD card
	`cd ~`
	`sudo sync`
	`umount /media/sd-card/root`
	`umount /media/sd-card/boot`

Via IP through Wireless Connection	
• You can use this method if you wish to access the RPi via an existing WiFi network • Change to the **boot** directory on the SD card • Enable **ssh** access by creating a blank file named **ssh**	`cd /media/sd-card/boot` `touch ssh`
• Within the boot directory, create a file called **wpa_supplicant.conf** and edit	`vi wpa_supplicant.conf`

• When you have opened the new file**,** add the configuration at right and save
• Be sure to replace **SSID** with your local wireless network SSID
	`country=US`
	`ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev`
	`update_config=1`
	`network={`
	`  ssid="MyWiFiNetwork"`
	`  psk="aVeryStrongPassword"`
	`  key_mgmt=WPA-PSK`
	`}`

• Change to your home directory, run **sync** as sudo, and unmount the SD card
• Remove the SD card
	`cd ~`
	`sudo sync`
	`umount /media/sd-card/root`
	`umount /media/sd-card/boot`

Via IP through USB connection	
• In this configuration you will be powering the RPi via the USB cable to your PC. Be sure to use the correct USB cable • On your Linux host you need to add a USB Network Interface and a network with the Pi on the USB interface. Give it a static address (ie 169.254.15.1)	

• Change to the **boot** directory on the SD card
• Edit the **config.txt** file
	`cd /media/sd-card/boot`
	
	`vi config,txt`
	
	`Append the following line:`
	
	`dtoverlay=dwc2`
	
	`Then save the file.`

• While in the boot directory, edit the **cmdline.txt** file, replace a line, then save file.	`vi cmdline,txt`
	
	`Replace with the following all as one continuous line:`
	
	`dwc_otg.lpm_enable=0 console=serial0,115200 console=tty1`
	`root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline fsck.repair=yes`
	`rootwait modules-load=dwc2,g_ether quiet`
	`init=/usr/lib/raspi-config/init_resize.sh`
	
	`Than save the file.`

• Remaining in the boot directory, enable **ssh** access by creating a blank file named **ssh**	`touch ssh`

• Change to the **rootfs** directory
• Edit the **interfaces** file
	`cd /media/sd-card/rootfs/etc/network`
	`vi interfaces`
	
	`Append the following lines:`
	
	`allow-hotplug usb0`
	`iface usb0 inet static`
	`address 169.254.15.2`
	`netmask 255.255.255.0`
	`network 169.254.15.0`
	`broadcast 169.254.15.255`
	`gateway 169.254.15.1`
	
	`Then save the file`

• Change to your home directory, run **sync** as sudo, and unmount the SD card
• Remove the SD card
	`cd ~`
	`sudo sync`
	`umount /media/sd-card/root`
	`umount /media/sd-card/boot`

Pi Setup

• Install Adafruit Radio Bonnet and Antenna • Insert SD card • Connect RPI and Power on • Log in with default pi:raspberry	If accessing via IP over USB, connect PC USB port to RPi USB Data port (next to mini-HDMI port) If connecting via SSH for first time click **yes** to **accept fingerprint**
• Run Configuration tool	`sudo raspi-config`

• Navigate through each menu making selections as noted then exit tool	`1. Change User Password to <something>`
	`2. Network options`
	`N1 Change hostname to your call + number [1-15]`
	`yourcall-5`
	`3. Boot Options`
	`B1 Desktop/CLI`
	`choose B1 Console`
	`4. Localization`
	`I1 Change Local to en_US.UTF-8`
	`I2 Change Timezone`
	`5. Interfacing options`
	`P2 Enable SSH`
	`P4 Enable SPI`
	`P5 Enable I2C`
	`7. Advanced Options`
	`A3 memory Split`
	`Reduce GPU from 64 to 16`
	`8. Update`

• Run **sync** as sudo, and **reboot**	`sudo sync` `sudo reboot`

Install Packages, Libraries and Repository

Packages	

• Log back into the RPi.
• Ensure you are in the home directory
• Install the following packages
◦ pip3 – Python Package Index
◦ Git – For cloning repositories
	`cd ~`
	`usdo apt-get install python3-pip`
	`sudo apt-get install git`

Python Libraries	

• Install the following Python libraries
◦ Python Image Library
◦ APRS and APRSlib
◦ Adafruit Radio Bonnet Libraries
	`sudo apt-get install python3-pil`
	`sudo pip3 install aprs`
	`sudo pip3 install aprslib`
	`sudo pip3 install adafruit-circuitpython-rfm69`
	`sudo pip3 install adafruit-circuitpython-rfm9x`
	`sudo pip3 install adafruit-circuitpython-ssd1306`
	`sudo pip3 install adafruit-circuitpython-framebuf`

HVDN Repository	

• Ensure you are in the home directory
• Make two new directories called **hvdn-comm** and **hvdn-repo**
• Change directory to hvdn-repo and clone the HASViolet repo from Github
• Copy the HASviolet stable directory to **hvdn-comm**
	`cd ~`
	`mkdir hvdn-comm`
	`mkdir hvdn-repo`
	`cd hvdn-repo`
	`git clone https://github.com/hudsonvalleydigitalnetwork/hasviolet.git`
	`cp /home/pi/hvdn-repo/hasviolet/stable /home/pi/hvdn-comm`

Using HVDN Communicator

HVDN Communicator is data only currently designed to be used on local LoRa networks. It is installed in ***/home/pi/hvdn-comm***

HVDN Communicator is built with Python. Applications include;

- **hvdn_lora-beacon.py** sends a repeating broadcast message
- **hvdn_lora-chat.py** is a half-duplex messaging app
- **hvdn_lora-tx.py** sends a message to another LoRa station
- **hvdn_lora-rx.py** listens for messages from other LoRa stations

Three files dependend by all applications are;

- **hvdn-comm.ini** is a configuration file
- **rf95.py** is a Python Library for the HOPE RFM95 modules on the Raspberry Radio Bonnet
- **font5x8.bin** used by the OLED on the Adafruit Radio Bonnet

hvdn_lora-beacon.py

Beacon a LoRa message

```
Usage: hvdn_lora-beacon.py -c COUNT -t DELAY "message"

OPTIONS
    -c Number of times to repeat MESSAGE
    -t NUmber of seconds before repeat MESSAGE
    MESSAGE is message to be send within double quotes
```

hvdn_lora-chat.py

Half-duplex LoRa messaging app

```
Usage: ./hvdn_lora-chat [-r] [-s]

OPTIONS
  -h, --help      show this help message and exit
  -r, --raw_data  Receive raw data
  -s, --signal    Signal Strength
```

- Starts and loops in Listening Mode
- CTRL-Z to send a message, CTRL-C to exit program
- When in send mode
    - Recipient is node id (255 = broadcast address)
    - Message is whatever message followed by enter
    - Message is sent, return to listening mode

hvdn_lora-tx.py

Send a LoRa message

```
Usage: hvdn_lora-tx.py -d DESTINATION "message"

OPTIONS
        -d Destination ID
        MESSAGE is message to be send within double quotes
```

hvdn_lora-rx.py

Listens for messages from other LoRa stations

```
Usage: ./hvdn_lora-rx.py -r -s

OPTIONS
    -h, --help      show this help message and exit
    -r, --raw_data  Receive raw data
    -s, --signal    Signal Strength
```