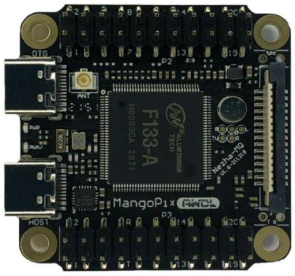
RT-Smart Setup

Setting up the RT-Smart micro-kernel on MangoPI RISC-V architecture (MANGOPI-NEZHA-MQ-01) using WSL 1 on Microsoft Windows 11.

HARDWARE

MANGOPI-NEZHA-MQ-01



Link: https://www.mouser.com/ProductDetail/RT-Thread/MANGOPI-NEZHA-MQ-01?qs=Znm5pLBrcALHG%252BSjHvKgpg%3D%3D&mgh=1&gclid=Cj0KCQiA3eGfBhCeARIsACpJNU-lkpseEfvPrBpbanjVO6I1jCWP56vp-OipaSRnkwdCpEID-urCiRgaAmDPEALw_wcB

USB-TO-SERIAL

Link: https://a.co/d/a2sa2Co

USB C-TYPE

Link: https://a.co/d/4ALR3CK

SOFTWARE

Windows 11 (Home Edition)
Windows Subsystem for Linux Version 1
Ubuntu 18.0.4

RT-SMART INSTALLATION

Open the ubuntu command prompt and issue the following commands to git checkout

git clone https://github.com/RT-Thread/userapps.git

cd userapps

git clone https://github.com/RT-Thread/rt-thread.git

To set the environment,

#./smart-env.bat riscv64

```
root@LAPTOP-J9VEUE58:/home/stanley/userapps# ./smart-env.sh riscv64
Arch      => riscv64
CC        => gcc
PREFIX    => riscv64-unknown-linux-musl-
EXEC_PATH => /home/stanley/userapps/tools/gnu_gcc/riscv64-linux-musleabi_for_x86_64-pc-linux-gnu/bin
root@LAPTOP-J9VEUE58:/home/stanley/userapps#
```

To download the respective tool chain change working directory to userapps/tools

#python3 get_toolchains.py riscv64

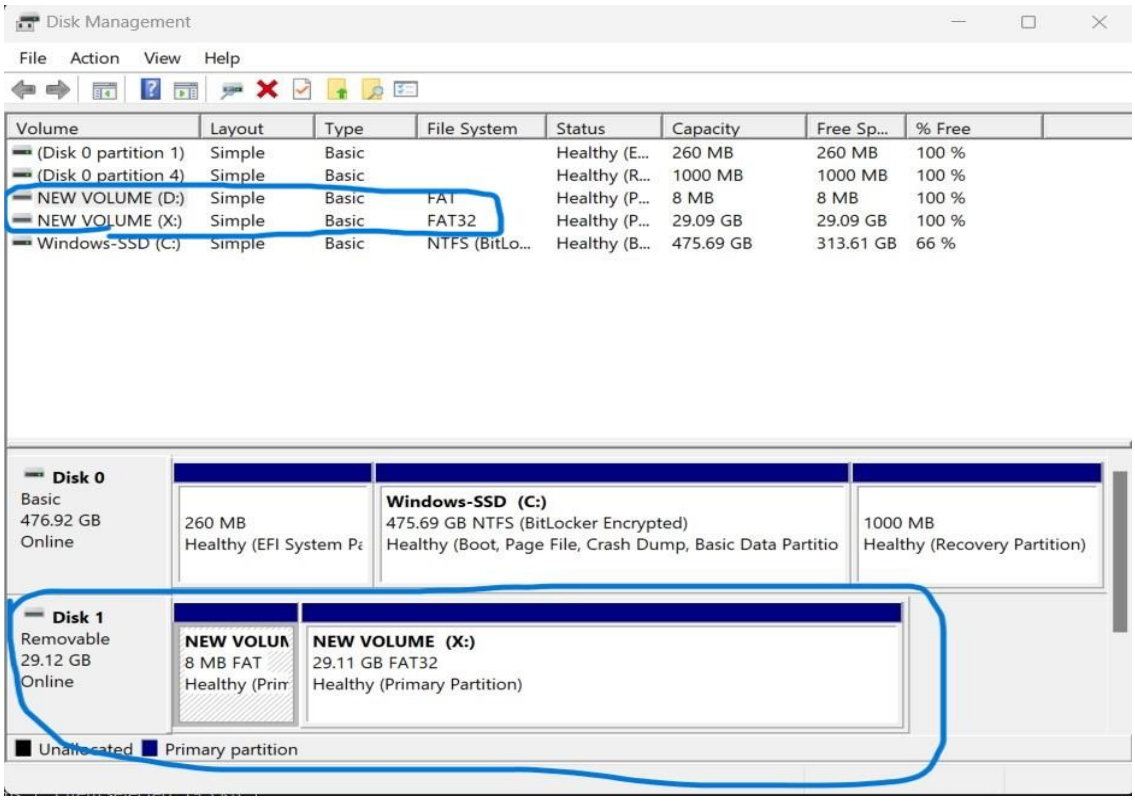
OR you can manually download it by following below steps

#wget
https://github.com/RT-Thread/toolchains-ci/releases/download/v1.7/riscv64-linux-musleabi_for_x86_64-pc-linux-gnu_latest.tar.bz2

#tar xjf riscv64-linux-musleabi_for_x86_64-pc-linux-gnu_latest.tar.bz2 -C /opt

SD Card Partition

Format the new SSD card as in below screenshot using Microsoft disk management tool or other 3rd party tool.



Compilation Guide

To compile the image,change the working directory to /root/rt-thread/bsp/allwinner/d1s/

#scons

Note: if there is any error, issue the following command to see if does resolve the issue

#scons –menuconfig

Do not change anything but just exit and save it. Re-run the following command again.

#scons

During the compilation process, ignore the warning messages. At the end of the process there will be a sd.bin at the current working directory as below screenshot.

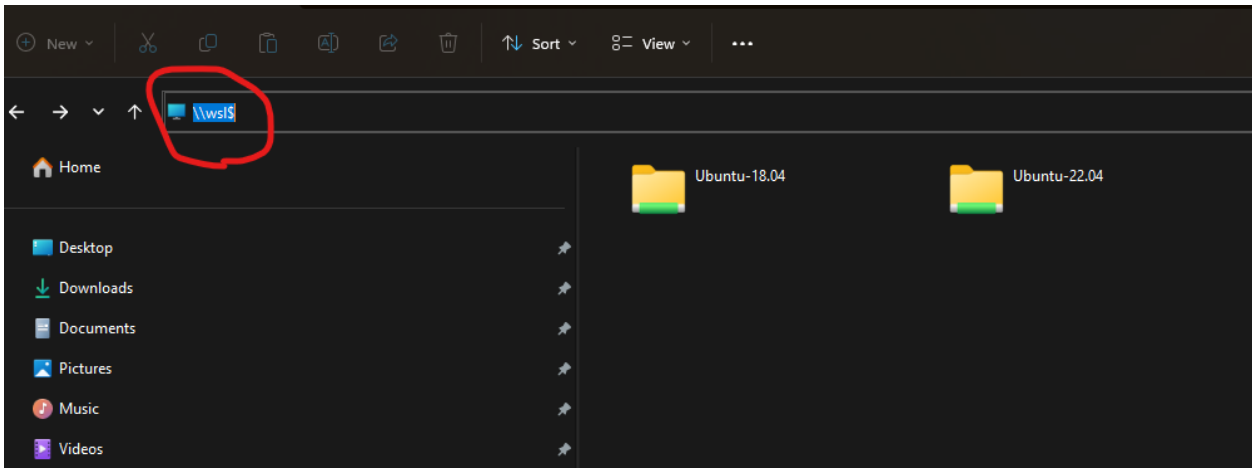
```
root@LAPTOP-J9VEUE58:~/rt-thread/bsp/allwinner/d1s# ls
Kconfig       SConscript    board     link_stacksize.lds  ports          rtthread.bin  sd.bin
README-M7.md  SConstruct    build     mkimage             rtconfig.h     rtthread.elf  toc1.cfg
README-MQ.md  __pycache__   figures   mksdcard.sh         rtconfig.py    rtthread.map  tools
README.md     applications  link.lds  mksdimg.sh          rtconfig.py.bk sbi.bin       u-boot.dtb
root@LAPTOP-J9VEUE58:~/rt-thread/bsp/allwinner/d1s#
```

Transfer rt-smart images to SD card

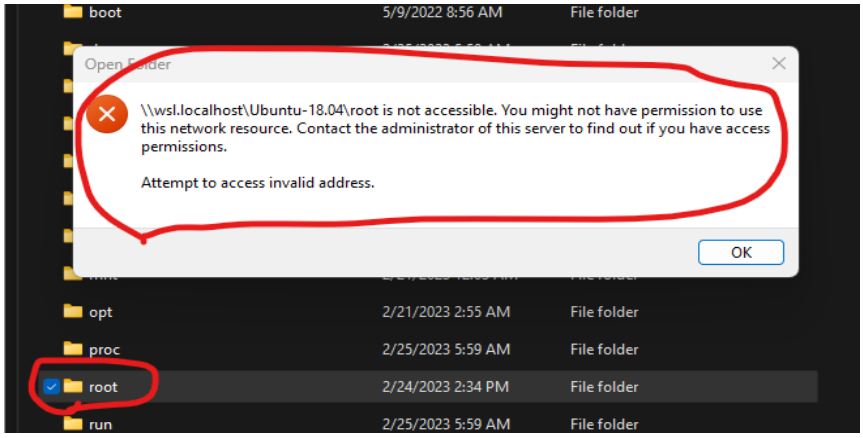
Transfer the following files to your host windows machine before burning to SD card,

/root/rt-thread/bsp/allwinner/d1s/sd.bin
/root/rt-thread/bsp/allwinner/d1s/tools/boot0_sdcard_sun20iw1p1_d1s.bin

In order to transfer the above 2 files, open a folder and type “\\wsl$” in the folder address bar. Select Ubuntu-18.04 if you have multiple versions running on your machine. Follow the path and copy and paste the files to your local machine desktop.



Note: if there is permission issue opening root folder as below screenshot,



Follow the below steps in Ubuntu's command prompt by changing the current working directory to /root/rt-thread/bsp/allwinner/d1s/.

#pwd (double check the current working directory)

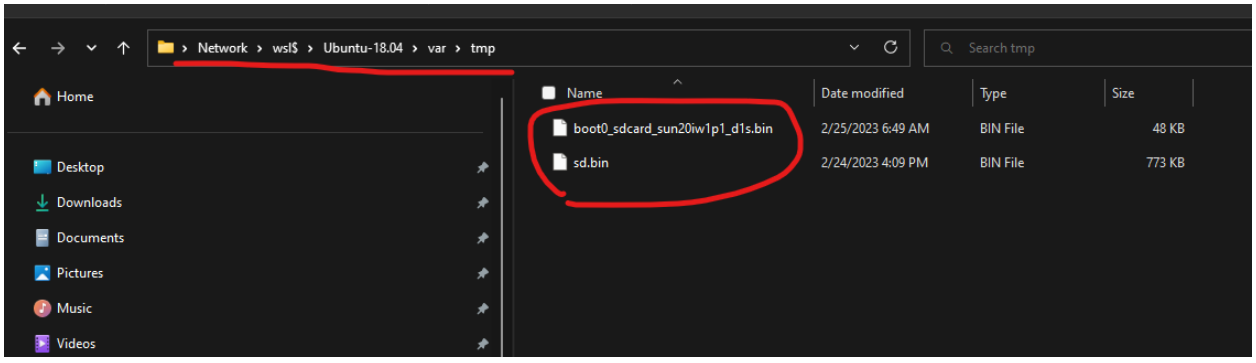
#ls -ald /var/tmp (double check the /var/tmp folder permission, it should be drwxrwxrwt)

#cp sd.bin /var/tmp/ (copy the sd.bin to /var/tmp folder)

#cd tools

#cp boot0_sdcard_sun20iw1p1_d1s.bin /var/tmp (copy the boot file to /var/tmp folder)

Now open the folder /var/tmp on your local machine to copy (not cut) the 2 files.



Download the images to SD card

Downloading the below 2 files required a 3rd party SD card writer software.

Boot0_sdcard_sun20iw1p1_d1s.bin
sd.bin

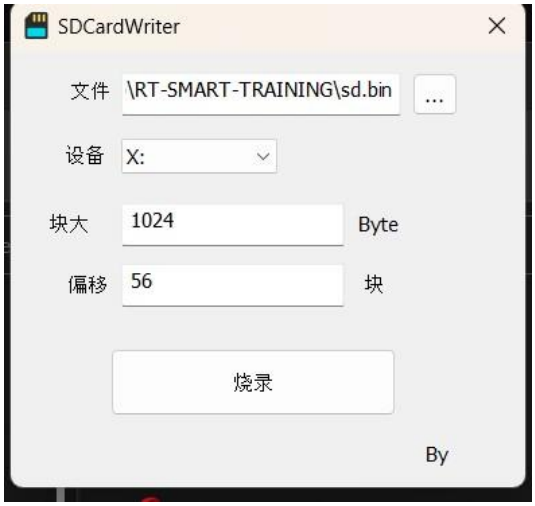
I used the following open SD card writer - https://github.com/malasy/SDCardWriter/releases.

Dev Board Persimmon Pie M7 as an example, the compiled user space executable file is packaged and burned into the EMMC on the board using the xfel tool
https://github.com/xboot/xfel

And download the files to the SD card according to the below screenshots.

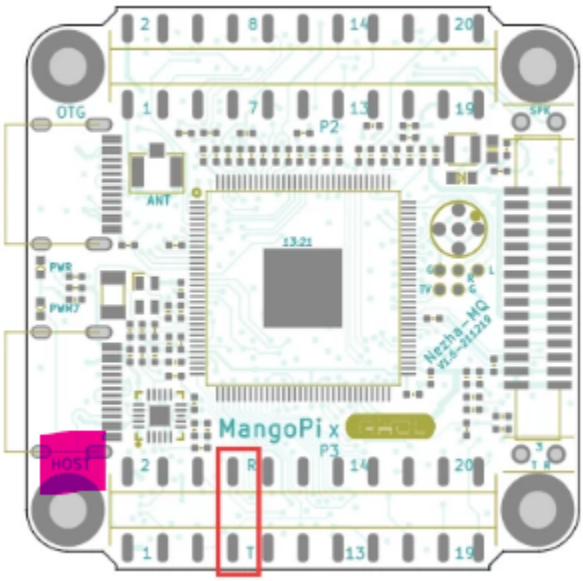
Boot0_sdcard_sun20iw1p1_d1s.bin

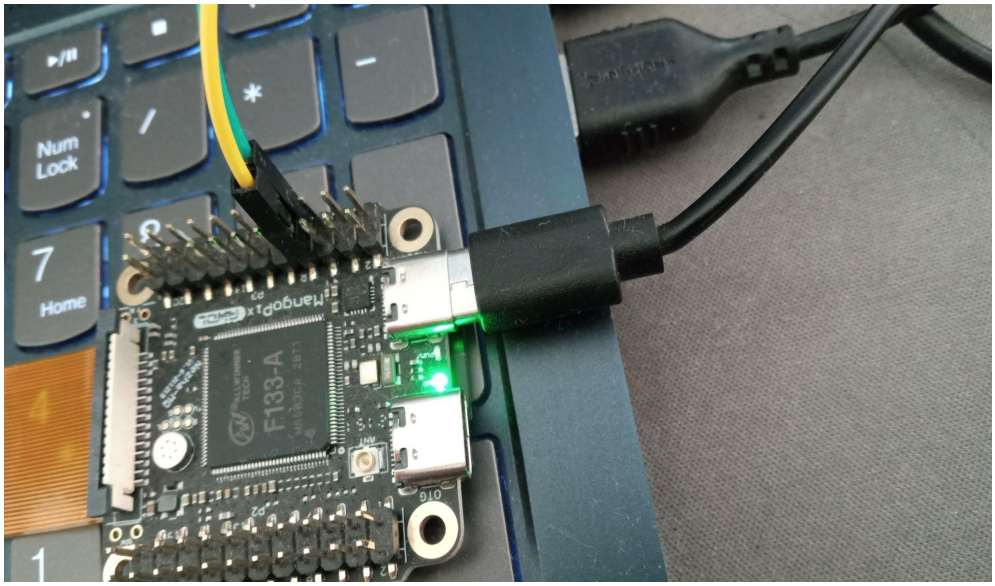
sd.bin



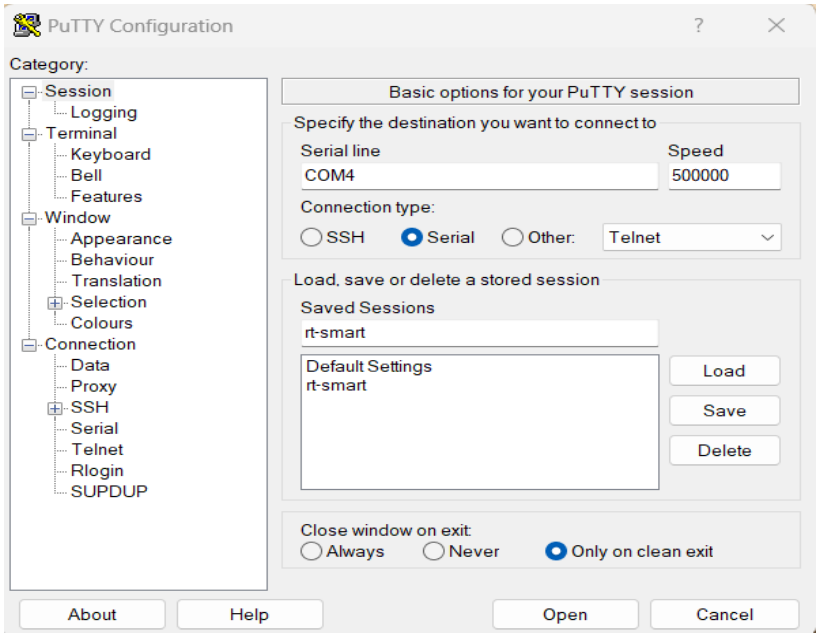
Connect to rt-smart terminal

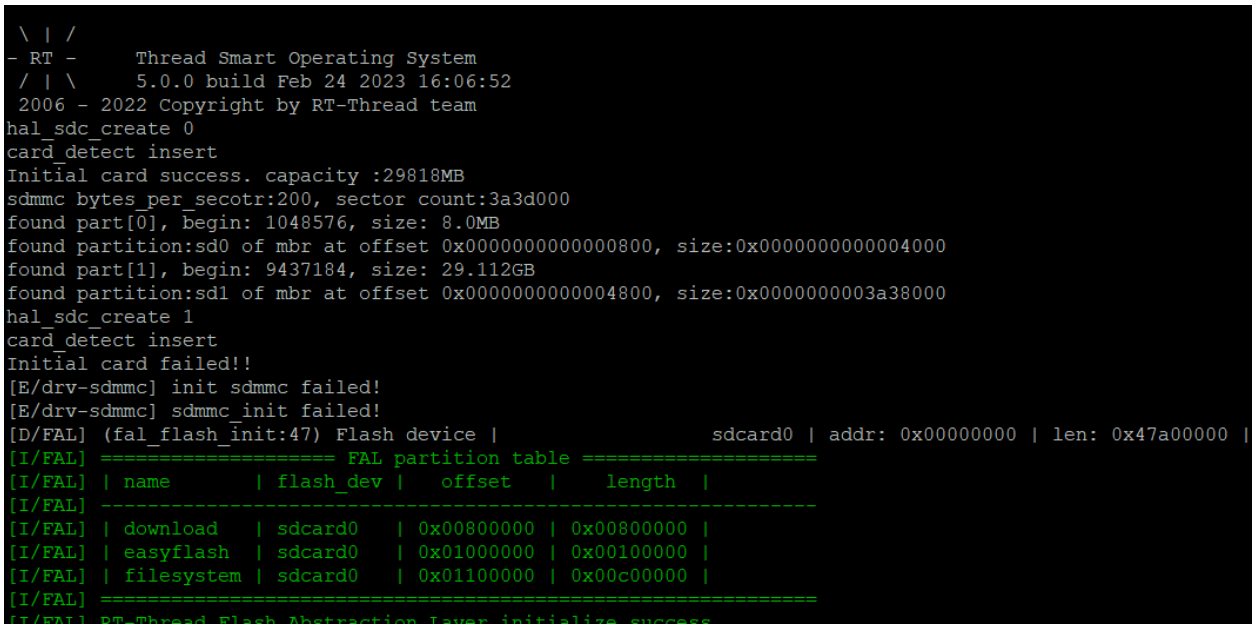
Connect Rx and Tx cable to the dev board and use the usb c type cable to power up the device.





Finally, use the Putty.exe to connect to rt-smart's terminal using 500000 baud rate. Serial COM port numbers will vary depending on different machines. Device Manager to find out the exact COM port number respective to your machine.





-Byte-me-stan
12/25/2023