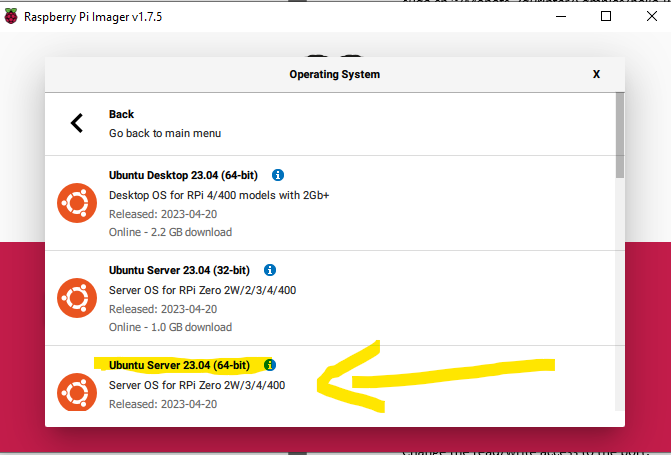
RoMoController Instalation tutorial

The recommended OS for the Raspberry Pi is Ubuntu Server 23.04 or higher because it contains the latest toolchain and apps, I cannot guarantee this document would be valid with a different distribution.



All commands for terminal will have an italic font.

1. *sudo apt update*
2. Install cmake, min version 3.24

*sudo apt install cmake*

If you are not using the latest Ubuntu OS you may not have cmake v3.24+ in the package manager, follow these steps to install a newer cmake version(otherwise skip to step 3):

You can download the current latest release for arm chips using this command:

*curl -OL* [*https://github.com/Kitware/CMake/releases/download/v3.26.4/cmake-3.26.4-linux-aarch64.sh*](https://github.com/Kitware/CMake/releases/download/v3.26.4/cmake-3.26.4-linux-aarch64.sh)

then make the script an executable using:

*sudo chmod +x cmake-3.26.4-linux-aarch64.sh*

Then run the script to download and unzip using:

*./cmake-3.26.4-linux-aarch64.sh*

Now you should have the cmake executable in the current directory, we need to add it to the PATH so you can access it without the path prefix:

*sudo nano ~/.bashrc*

at the end of .bashrc write this line:

export PATH=”/home/$USER/cmake-3.26.4-linux-aarch64/bin:$PATH”

1. Install clang, you may need to install clang manually if you are not using the latest Ubuntu OS, the minimum version of clang should be version 5 for C++17 support.

Run this command to install all tools:

*sudo apt install clang-format clang-tidy clang-tools clang clangd libc++-dev libc++1 libc++abi-dev libc++abi1 libclang-dev libclang1 liblldb-dev libllvm-ocaml-dev libomp-dev libomp5 lld lldb llvm-dev llvm-runtime llvm python3-clang*

1. *sudo apt install ninja-build*
2. Add your user to the dialout group so you can access the serial port without sudo rights:

*sudo adduser $USER dialout*

1. *sudo reboot*
2. *cd ~*
3. *git clone* [*https://github.com/vladsomai/Mobots-3dPrinter.git*](https://github.com/vladsomai/Mobots-3dPrinter.git)
4. *cd Mobots-3dPrinter*
5. We must specify the path to the compiler in the build.sh file. We can find the clang path by using this command:

*clang++ –version*

you should see the installation dir of clang, now set this path in the build.sh:

*sudo nano ./build.sh*

modify the C\_COMPILER and CXX\_COMPILER variables to the path you got from clang++ --version

after you save the file, execute it:

*./build.sh Release*

You should see the build files generated in the “build” folder together with the executable “RoMoController”

1. *cd ./build/RoMoController*
2. Set the serial port:

Use this command to find all the connected serial ports

*sudo dmesg | grep tty*

for example I see “/dev/ttyUSB0”

You can use minicom to setup the serial port and test the motors reset when sending the reset all command from the following 2 commands in the command line:

*sudo echo -en '\xFF\x1B\x00' > binary.file*

*cat binary.file > /dev/ttyUSB0*

Modify the SERIAL\_PORT from config.ini to the correct serial port:

sudo nano ~/Mobots-3dPrinter/build//RoMoController/config.ini

1. Run the executable using:

*./RoMoController*

After running the executable, the motors should start moving according to the gcode file, you will see a log file created in the same directory as the executable, the name of the log file is the current date, the app will generate max 2 logs and alternate between them when they reach 2mb in size(e.g. log\_1 is full, start log\_2, log\_2 is full, delete log\_1 and start log\_1 from scratch and so on). The log file shall describe the steps the app makes, show errors or info, the log is not 100% complete it may not contain all the errors at the moment.