**CLion remote developing and debugging for RPI and other remote Linux systems**

**Date: 30.10.2024**

**Written by: GRIGORI DMITRII**

1. **Introduction**

This project involves exploring the methods for developing and debugging applications using cross-compiling and remote debugging for Raspberry Pi and/or other distinct Linux platforms, utilizing the CLion development environment. Additionally, the creation of a tutorial covering these processes is required.

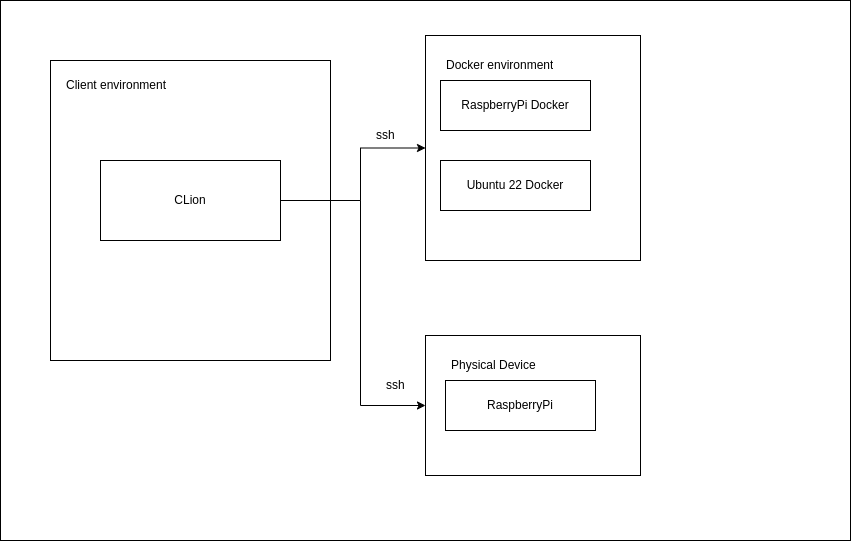
1. **System Overview**

As a development environment, CLion will be used for this project, specifically 2024.2.3 version (tutorial may vary for future versions) and a students’ email account will be used for free license. If a student account is not used, the license is limited to 30 days.

For remote developing environment: several will be included in this documentation, which include:

* Raspberry PI on hardware
* Raspberry PI in Docker environment
* OS based on Linux (Debian 12, Ubuntu 22 and others may apply)

1. **Architecture**



1. **Plan of realization**

**1. Install CLion (2024.2.3 version)**:

* Download and install CLion from the JetBrains website.
* Register with a student email for a free license or activate a 30-day trial if necessary.

**2. Configure CLion**

* Ensure CLion is configured with necessary plugins for remote development and cross-compilation support.
* Set up toolchains and CMake for remote development.

**3. Install required Toolchains**

* For Raspberry Pi, install **arm-none-eabi-gcc** or any cross-compiler suitable for ARM-based devices.
* For Raspberry Pi hardware, install **gdbserver** and **rsync**
* For Docker-based development, ensure the Docker container includes necessary build tools and cross-compilation libraries.

**4. Configure Cross-Compiling in Clion**

* Create a toolchain in CLion specifically for cross-compilation, pointing to the cross-compiler installed.
* Set up CMake configurations that specify cross-compilation targets.

**5. Configure Remote Toolchains in Clion**

* Go to **Settings > Build, Execution, Deployment > Toolchains**.
* Add a remote toolchain, specifying the target machine's IP and authentication details.
* Create a remote debug configuration in CLion, specifying the remote gdbserver details and application path on the target machine.

**6. Run and test**

* Create a "Hello, World!" C/C++ application in Clion.
* Build the application using the cross-compiler toolchain and deploy it to the remote target.
* Use CLion’s remote debugger to run and step through the code on the target device.

RASPBERRY STEPS

<https://stackoverflow.com/questions/68101936/how-to-increase-the-size-of-dev-root-on-a-docker-image-on-a-raspberry-pi/68328492#68328492>

docker run -it -p 5022:5022 -v C:\Users\dmitrii.grigori\Documents\teme\an3\sem1\SO2\proiect\raspberry\_v:/sdcard lukechilds/dockerpi

login: pi

password: raspberry

sudo poweroff

HOST: (wsl)

cd /mnt/c/Users/dmitrii.grigori/Documents/teme/an3/sem1/SO2/proiect/raspberry\_v

sudo apt install qemu-utils fdisk

sudo qemu-img resize -f raw filesystem.img 10G

startsector=$(fdisk -u -l filesystem.img | grep filesystem.img2 | awk '{print $2}')

sudo parted filesystem.img --script rm 2

sudo parted filesystem.img --script "mkpart primary ext2 ${startsector}s -1s"

restart docker container:

docker run -it -p 5022:5022 -v C:\Users\dmitrii.grigori\Documents\teme\an3\sem1\SO2\proiect\raspberry\_v:/sdcard lukechilds/dockerpi

login and run:

sudo resize2fs /dev/sda2 8G

A screen shot of a computer

Description automatically generated

sudo nano /etc/apt/sources.list

sudo nano /etc/apt/sources.list.d/raspi.list

**edit** deb http://raspbian.raspberrypi.org/raspbian/ **buster** main contrib non-free rpi

**with** deb http://raspbian.raspberrypi.org/raspbian/ **bookworm** main contrib non-free rpi

ctrl+o, enter, ctrl+x

sudo apt-get update

sudo apt-get upgrade -y (~10min)

A screenshot of a computer error

Description automatically generated

Press q

Until here…

sudo apt-get dist-upgrade (~10 mins)

ssh pi@localhost -p 5022, password: raspberry

sudo su



Apt-get install mc