Calcolo Parallelo dall'Infrastruttura alla Matematica

Calcolo parallelo: perché, quali infrastratture, quali problemi?

Laurea Triennale e Magistrale in Matematica

Fabio Durastante

April 28, 2023

Dipartimento
di Matematica
Università di Pisa

Table of Contents

Clusters

1 Building our machine

Cluster

"Clusters are an ensemble of **off-the-shelf computers** integrated by an **interconnection network** and operating within a single administrative domain and usually within a single machine room. Commodity clusters employ **commercially available networks** (e.g., **Ethernet**, Myrinet) as opposed to custom networks (e.g., IBM SP-2). *Beowulf-class* clusters incorporate mass-market PC technology for their compute nodes to achieve the best price/performance."

Beowulf Cluster Computing with Linux

Nodes and Network

1 Building our machine

A **node** is responsible for all activities and capabilities associated with executing an application program and supporting a sophisticated software environment:

1. instruction execution;
2. high-speed temporary information storage;
3. high-capacity persistent information storage, and
4. **communication** with the external environment, including **other nodes**.

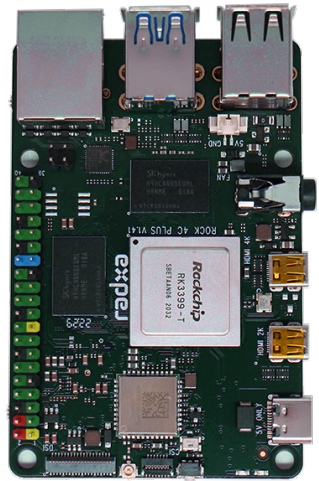
A **network** is a combination of *physical transport* and *control mechanisms* associated with a **layered hierarchy** of **message** encapsulation.

Building our machine

Let's start with **the nodes**:

Radxa ROCK 4C+

CPU	Arm® big.LITTLE™ technology: Dual Cortex® A72 frequency 1.5GHz and a Quad Cortex A53 frequency 1.0GHz,
GPU	Arm Mali™ T860MP4 GPU,
RAM	Dual channel 4GB 64bit LPDDR4,
LAN	1x Gigabit Ethernet port,
HD	1x micro SD card slot.



Building our machine

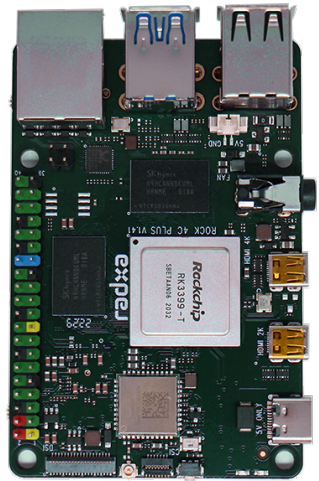
1 Building our machine

Let's start with **the nodes**:

Radxa ROCK 4C+

CPU	Arm® big.LITTLE™ technology: Dual Cortex® A72 frequency 1.5GHz and a Quad Cortex A53 frequency 1.0GHz,
GPU	Arm Mali™ T860MP4 GPU,
RAM	Dual channel 4GB 64bit LPDDR4,
LAN	1x Gigabit Ethernet port,
HD	1x micro SD card slot.

Why two processors?

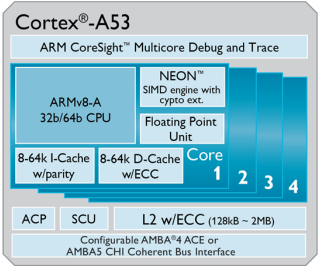


The ARM big.LITTLE architecture

The idea

ARM big.LITTLE is a **heterogeneous computing architecture** coupling *relatively* energy-saving and slower processor cores (LITTLE) with *relatively* more powerful and power-hungry ones (big).

- Only one ”side” or the other will be active at once,
- All cores have access to the same memory regions, so workloads can be swapped between *big* and *LITTLE* cores on the fly.



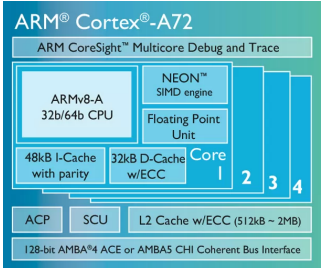
LITTLE

The ARM big.LITTLE architecture

The idea

ARM big.LITTLE is a **heterogeneous computing architecture** coupling *relatively* energy-saving and slower processor cores (LITTLE) with *relatively* more powerful and power-hungry ones (big).

- Only one ”side” or the other will be active at once,
- All cores have access to the same memory regions, so workloads can be swapped between *big* and *LITTLE* cores on the fly.



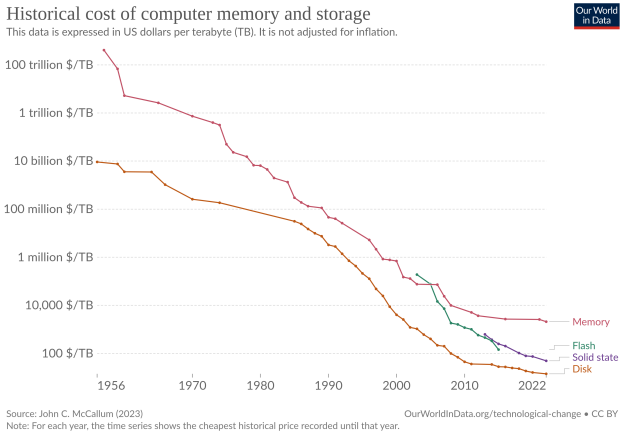
big

The 4GB 64bit LPDDR4 Memory

1 Building our machine

The LPDDR4 acronym stands for **L**ow-**P**ower **D**ouble **D**ata **R**ate **4** dynamic RAM.

Along with processor speed (*Moore's Law*), memory capacity has grown at a phenomenal rate, quadrupling in size approximately every three years.



HD: Storage

1 Building our machine

Each node will use Kingston 64 GB Micro SD (SDHC Class 10) SDCS/32GB with

- OS - Local to the node.
- Home - Where the users files and program will reside, it will be a *shared file system*.

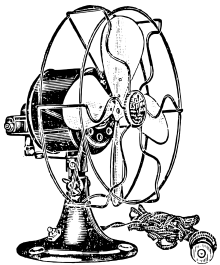
Power requirements and cooling

1 Building our machine

The Radxa ROCK 4C+ is **powered with a 5V source**.

- USB C 5V/3A,
- 5V Power applied to the GPIO PIN 2 & 4.

The recommended power source capacity is at least 5V/3A.



*"The Radxa ROCK 4C+ will operate perfectly well without any additional cooling and is **designed for sprint performance** - expecting a light use case on average then ramping up the CPU speed when needed (e.g. when loading a webpage). If a user wishes to load the system continually or operate it at a high termperature (sic.) at full performance, further cooling may be needed."*

OS: What flavor of Linux?

1 Building our machine

The Radxa ROCK 4C+ has **Debian/Ubuntu Linux support**, images can be obtained from:

https://wiki.radxa.com/Rockpi4/downloads

- Ubuntu 20.04.6 LTS (Focal Fossa),
- Server install image $\Rightarrow$ No GUI!
- Again, *why Linux*? Linux is the unchallenged champion for building compute engines with commodity parts: www.top500.org.

OS Setup

≈ 10 Gb for the OS:

- Compilers: GCC Suite v10.3.0,
- MPI: OpenMPI v4.0.3,
- OpenBLAS v0.3.8,
- Valgrind v3.15.0.

≈ 40 Gb of shared filesystem for the homes.

OS Setup

1 Building our machine

We will use the **Gluster** **F**ile **S**ystem (www.gluster.org)

≈ 10 Gb for the OS:

- Compilers: GCC Suite v10.3.0,
- MPI: OpenMPI v4.0.3,
- OpenBLAS v0.3.8,
- Valgrind v3.15.0.
- GlusterFS v7.2

≈ 40 Gb of shared filesystem for the homes.

OS Setup

1 Building our machine

≈ 10 Gb for the OS:

- Compilers: GCC Suite v10.3.0,
- MPI: OpenMPI v4.0.3,
- OpenBLAS v0.3.8,
- Valgrind v3.15.0.
- GlusterFS v7.2

≈ 40 Gb of shared filesystem for the homes.

We will use the **Gluster** **F**ile **S**ystem (www.gluster.org)

"Gluster is a distributed scale-out filesystem that allows rapid provisioning of additional storage based on your storage consumption needs. It incorporates automatic failover as a primary feature. All of this is accomplished without a centralized metadata server."

OS Setup

1 Building our machine

≈ 10 Gb for the OS:

- Compilers: GCC Suite v10.3.0,
- MPI: OpenMPI v4.0.3,
- OpenBLAS v0.3.8,
- Valgrind v3.15.0.
- GlusterFS v7.2

≈ 40 Gb of shared filesystem for the homes.

We will use the **Gluster F**ile **S**ystem (www.gluster.org)

"Gluster is a distributed scale-out filesystem that allows rapid provisioning of additional storage based on your storage consumption needs. It incorporates automatic failover as a primary feature. All of this is accomplished without a centralized metadata server."

All the **nodes** need to have the **same configuration** and **software**!

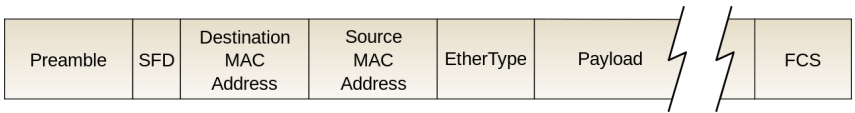
Table of Contents

2 Network architecture

The LAN Network

2 Network architecture

The Ethernet message packet comprises a sequence of **multibit fields**, one of which is variable length. The fields include a **combination** of **network control information** and **data payload**.

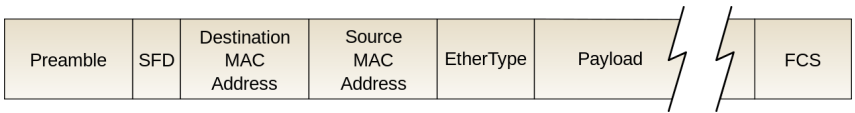


- TCP/IP is the *de facto* standard network communication protocol:
  - The **destination** of an Internet Protocol packet is specified by a 32-bit IP address, e.g., 192.168.1.2.

The LAN Network

2 Network architecture

The Ethernet message packet comprises a sequence of **multibit fields**, one of which is variable length. The fields include a **combination** of **network control information** and **data payload**.



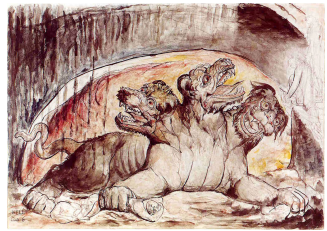
- TCP/IP is the *de facto* standard network communication protocol:
  - The **destination** of an Internet Protocol packet is specified by a 32-bit IP address, e.g., 192.168.1.2.
- Before assigning IP addresses to our nodes, designing the network topology, and booting all the machine, we need to **decide how the system will be accessed**: how a user can log in to a system and use the machine?

How do we access the system?

2 Network architecture

The Standalone System: unattached to any external networks, the user need to be in the same room of the machine.

The Universally Accessible Machine: every node is accessible from the entire Internet.



The Guarded Beowulf: reserved IPs to all internal nodes, and single front-end with an IP address accessible from outside.

How do we access the system?

2 Network architecture

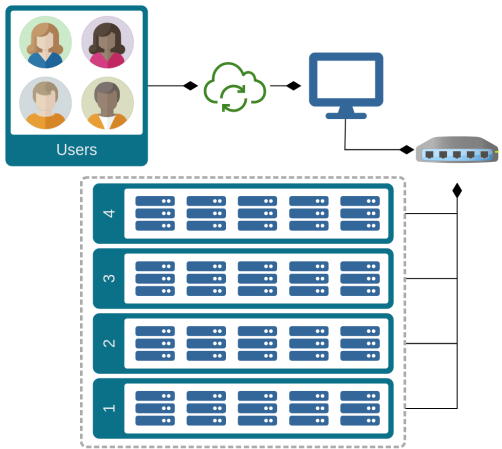
The Standalone System: unattached to any external networks, the user need to be in the same room of the machine.

The Universally Accessible Machine: every node is accessible from the entire Internet.

The Guarded Beowulf: reserved IPs to all internal nodes, and single front-end with an IP address accessible from outside.

Our network architecture

2 Network architecture



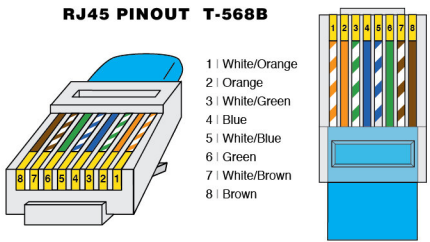
We will build a **guarded Beowulf** with an access node, where the user will log-in.

All the **nodes** will be connected to the **switch** via Ethernet connection, and will be powered through it.
The **access node** will be configured on the IP:

```
IP: 131.114.10.121
Gateway: 131.114.10.1
```

Ethernet Cables

2 Network architecture



1. Cut the cable to the length needed,
2. Strip back the cable jacket approximately 2.5 cm,
3. Use the **568-B wiring scheme** on both ends for a standard patch cable.

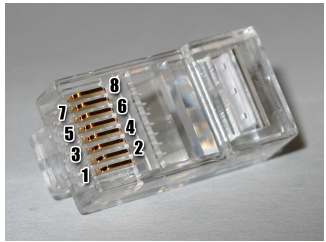
The **maximum length** for a cable segment is **100 meters**. If longer runs are required repeaters or switches are necessary.

Ethernet Cables

Pin	Pair	Wire	Color
1	2	1	
2	2	2	
3	3	1	
4	1	2	
5	1	1	
6	3	2	
7	4	1	
8	4	2	

ANSI/TIA-568 Standard

1. Cut the cable to the length needed,
2. Strip back the cable jacket approximately 2.5 cm,
3. Use the **568-B wiring scheme** on both ends for a standard patch cable.



Power over Ethernet (PoE)

2 Network architecture

Power over Ethernet, or PoE, describes any of *several standards* or ad hoc systems that **pass electric power** *along* with **data** on **twisted-pair Ethernet cabling**.

- 24 10/100/1000Mbps RJ45 PoE+ Ports Switch with 2 SFP Slots,
- **PoE power budget** is up to **250 W** (in *laboratory environment*).

Compliant with IEEE802.3af/at
Isolation: 2.5kV
Power Input: DC44 57V
Power Output: DC5V 3A

Our “Rack”

2 Network architecture

- We will divide our 20 nodes into these racks,
- The fans must be connected on the red and black GPIOs (power and ground).

Table of Contents

Install the OS

3 Let's rock and roll

Since all the nodes have to be equal we move in **two steps**:

1. Install and configure the first node,
2. Make copies of the same SD for all the other nodes *or* execute same configuration commands everywhere.

Install the OS

3 Let's rock and roll

Since all the nodes have to be equal we move in **two steps**:

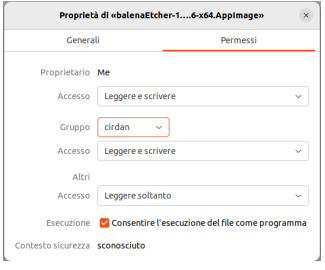
1. Install and configure the first node,
2. Make copies of the same SD for all the other nodes *or* execute same configuration commands everywhere.

Let's start with **step 1**, we go to wiki.radxa.com/Rockpi4/downloads and download

- Etcher - A user friendly Image Writer,
- Ubuntu 20 Server(Linux 4.4).

We use Etcher to write the system image on the SD, on Linux this will be a `.appimage` file, so first of all we have to *make it executable*.

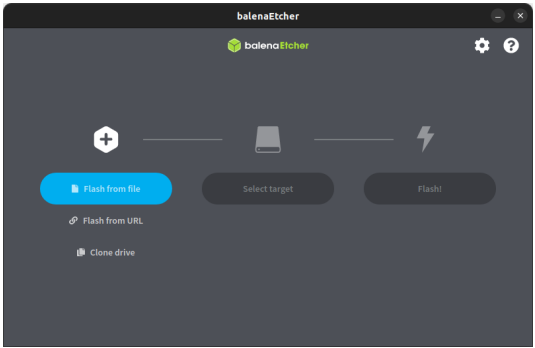
Remark: we need *root privileges*!



Install the OS

3 Let's rock and roll

Etcher is intuitive enough to use:



We select the Ubuntu 20 Server(Linux 4.4) image we have downloaded, then select the SD as target and the we flash the SD.

Gluster FS & Other Software

3 Let's rock and roll

To install GlusterFS in Ubuntu we could **follow these steps**:

```
apt install software-properties-common
add-apt-repository ppa:gluster/glusterfs-7
apt update
apt install glusterfs-server
```

Since we need to install the **same software** on **all** the **nodes**, we will use a default `#!/bin/bash` script to be run on each node with something like

```
for node in host1 host2; do
 scp /tmp/script.sh user@$node:/tmp/script.sh
 if [[ "$?" == "0" ]];then # checks that last command didn't fail
  ssh -oBatchMode=yes user@$node /tmp/script.sh
 fi
done
```

where `host?` is the name of the node and `user` the user that has to execute the code.

Initialization of the nodes

3 Let's rock and roll

The code should look something like:

```
#!/bin/bash
mkdir /scratch
chown -R rock:rock /home/rock
# Don't ask for anything
export DEBIAN_FRONTEND=noninteractive
# Cambia il fuso orario
timedatectl set-timezone Europe/London
# Fixa la chiave pubblica delle repository
# (da https://forum.radxa.com/t/gpg-error-with-ubuntu-server-20-04/1
↪  3392)
wget -O - apt.radxa.com/focal-stable/public.key | sudo apt-key add -
```

continue on the next slide

Initialization of the nodes

3 Let's rock and roll

continued from previous slide

```
# Update & Upgrade
apt update -y && apt upgrade -y
# Install required packages
apt -y install build-essential gcc openmpi-bin openmpi-common
↪  libopenmpi-dev slurm python3-pip valgrind tree git curl man-db
↪  mc parallel neovim unrar atool
```

This information is also available on the Wiki on the Git repository.

Work in Progress

3 Let's rock and roll

We still need to **configure**

- The `SLURM` queue manager,
- The *distributed* file-system GlusterFS.

The other important point is making the *accounts*!