Characterization of the Raspberry Pi 4

An Analysis of Hardware and Software Capabilities for the Embedded and real time systems project exploration

Objective

This report aims to characterize the Raspberry Pi 4 (RPI 4) in terms of its hardware and software capabilities to assess its potential for embedded and real-time systems applications. By exploring the RPI 4's key features, the goal is to understand how its components interact, what limitations exist, and the overall suitability for any specific embedded systems use case. This would serve as an exploration before deep diving into the project.

1. Hardware Characterization

What SoC is on the target board? What version and manufacturer of the SoC?

To know the details of the SOC on the RPI 4 is by running the command:

- `sudo cat /proc/device-tree/model`

The Raspberry Pi 4 is equipped with the Broadcom BCM2711 System-on-Chip (SoC). This is a quad-core Cortex-A72 (ARM v8-A) 64-bit processor with a clock speed of 1.5 GHz. Broadcom is the manufacturer of this SoC.

What features are present on the board/SoC?

The Raspberry Pi 4 includes the following features:

- **Processor:** Quad-core Cortex-A72 64-bit CPU (1.5 GHz)
- **Memory:** 2GB, 4GB, or 8GB LPDDR4 SDRAM
- **GPU:** VideoCore VI, supporting OpenGL ES 3.0
- **Storage:** microSD slot, USB 3.0 for external storage, network boot support,

NVME support (can be extended from the CM4 module)

- **Networking:** Gigabit Ethernet, 2.4/5.0 GHz WiFi (802.11ac), Bluetooth 5.0
- **I/O Interfaces:**
  - 40-pin GPIO header (standard rpi interface pinout)
  - 2 × USB 3.0, 2 × USB 2.0
  - 2 × micro HDMI (4K output)
  - 1 × MIPI DSI (Display), 1 × MIPI CSI (Camera)
  - 1 × 3.5mm audio jack
  - Power via USB-C (PD mode)

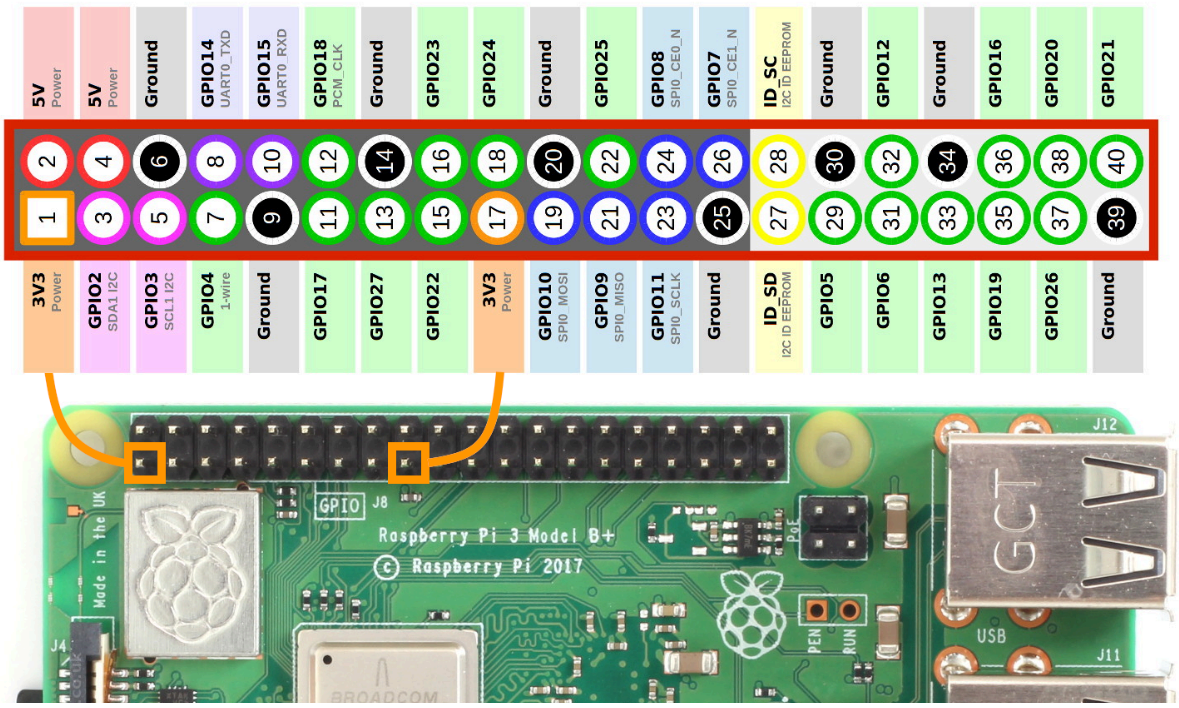
What size caches are present in the Raspberry Pi?

The Raspberry Pi 4 has the following cache hierarchy:

- **L1 Cache:** 48 KB instruction cache + 32 KB data cache per core
- **L2 Cache:** 1 MB shared among all cores
- **No L3 Cache**

What is the pin layout on this specific version of the target board?

The Raspberry Pi 4 has a **40-pin GPIO header**, which follows the same layout as previous Raspberry Pi models.



Key functions include:

- **Power:** 3.3V and 5V pins
- **Ground:** Multiple GND pins
- **Communication Protocols:**
- **I2C:** SDA (Pin 3), SCL (Pin 5)
- **SPI:** MOSI, MISO, SCLK, CE0, CE1
- **UART:** TX (Pin 8), RX (Pin 10)
- **PWM:** Available on multiple pins

How does the board boot? What is the process?

As mentioned in the RPI official Documentation webpage, the Raspberry Pi 4 follows this boot process:

1. The **BootROM** (built into the SoC) executes the first-stage bootloader from onboard firmware.
2. The firmware loads the second-stage bootloader (from EEPROM).
3. The second-stage bootloader reads the **config.txt** file on the microSD card or external boot device.
4. The Linux kernel is loaded, followed by the root filesystem.
5. The system initializes services and user applications.

How much memory is available on the board?

The Raspberry Pi 4 comes in **2GB, 4GB, or 8GB** LPDDR4 RAM configurations. I personally have the 8GB model which I verified using the htop interface.

What is the maximum memory configuration?

The maximum supported memory configuration is **8GB LPDDR4 RAM**. This is the same config as my RPI 4.

How many different types of memory are on the board?

There are three primary types of memory in the Raspberry Pi 4:

1. **LPDDR4 SDRAM** (main system memory)
2. **L2 Cache** (1MB shared among cores)
3. **Boot EEPROM** (stores bootloader firmware)

What is the performance of the memory?

The **LPDDR4 RAM** operates at **3200 MT/s**, providing improved bandwidth compared to previous Raspberry Pi models.

What is involved in adding an on-switch to the Raspberry Pi?

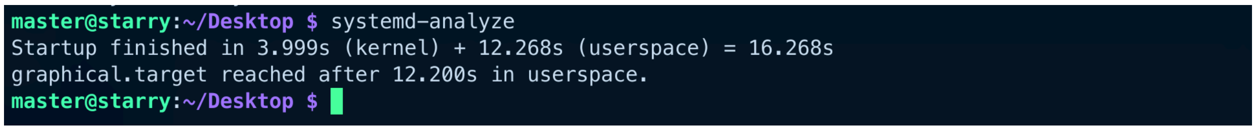
The Raspberry Pi does not have a built-in power button. To add a power-on switch:

1. Use a **momentary push button** across pins **5 (GPIO3) and 6 (GND)**.
2. When pressed, the Pi will wake up from shutdown (if configured in config.txt).
3. Alternatively, use an **inline power switch** with the USB-C power cable.

How long does it take between reboot and an active board? Is it deterministic?

The RPI 4 takes roughly 30 seconds to boot up with the Graphical OS version. The first method I tried to calculate that is by rebooting the rpi and running a script on my host machine to check the time it takes for the RPI to Ping again after connecting to the same network. This gave me an overall idea about the time it would take for me to connect with the RPI in a reboot event. But this was not enough to understand how these 30 seconds are consumed.

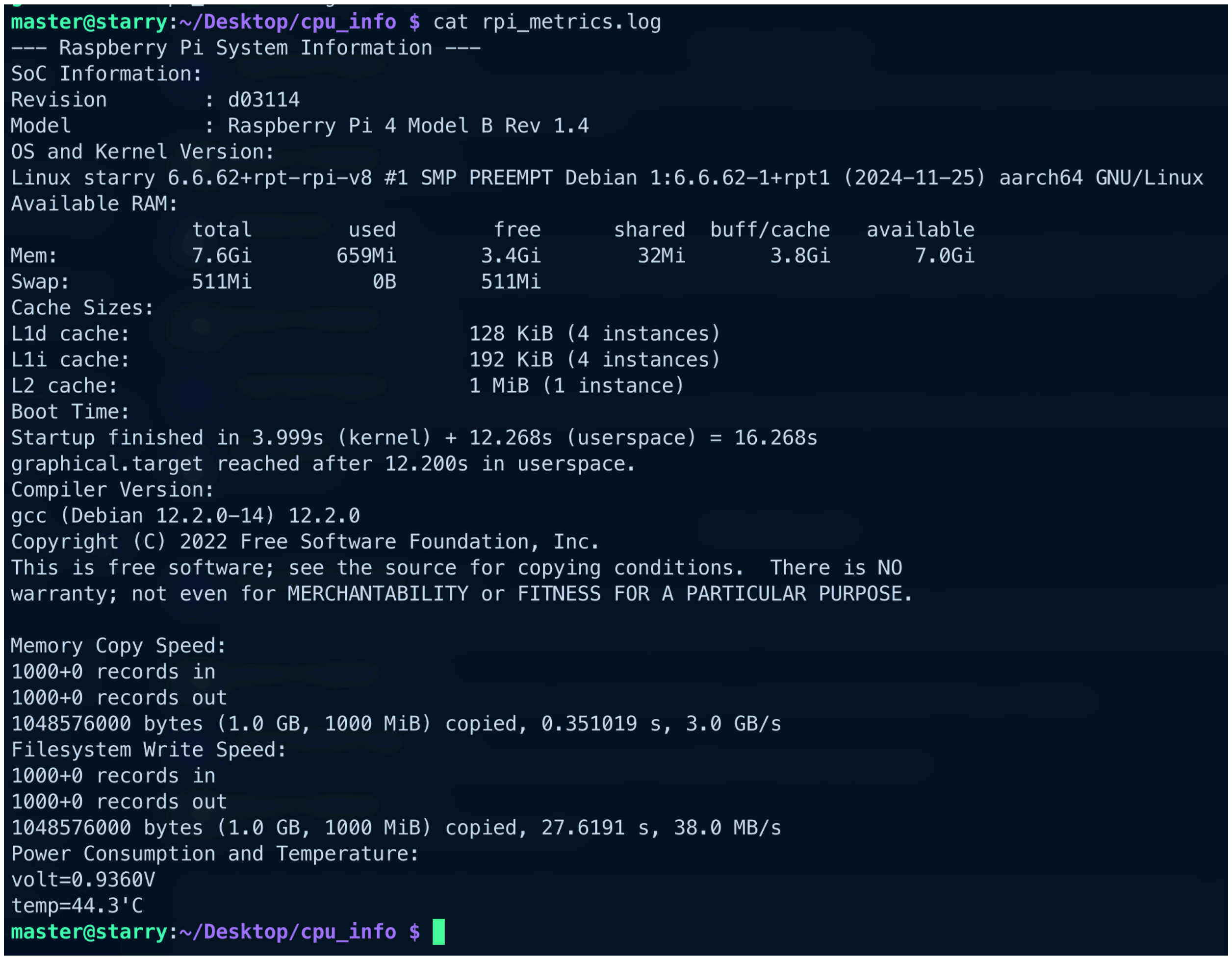
A very convenient way to determine the exact time for each step is by using an inbuilt command: systemd-analyze



```
master@starry:~/Desktop $ systemd-analyze
Startup finished in 3.999s (kernel) + 12.268s (userspace) = 16.268s
graphical.target reached after 12.200s in userspace.
master@starry:~/Desktop $
```

This totals the complete bootup time to 28.468 secs for this instance, So i am assuming it took about 1-2 seconds for the RPI 4 to connect to the wifi network to talk to my computer. I have also attached a detailed svg file with this report which can be generated using the same command.

Combining all the hardware characterization techniques, I have compiled a bash script to populate all the hardware specifications for the RPI 4. This is displayed in the screenshot below:



```
master@starry:~/Desktop/cpu_info $ cat rpi_metrics.log
--- Raspberry Pi System Information ---
SoC Information:
Revision        : d03114
Model           : Raspberry Pi 4 Model B Rev 1.4
OS and Kernel Version:
Linux starry 6.6.62+rpt-rpi-v8 #1 SMP PREEMPT Debian 1:6.6.62-1+rpt1 (2024-11-25) aarch64 GNU/Linux
Available RAM:
               total        used        free      shared  buff/cache   available
Mem:           7.6Gi       659Mi       3.4Gi        32Mi       3.8Gi       7.0Gi
Swap:          511Mi          0B       511Mi
Cache Sizes:
L1d cache:                            128 KiB (4 instances)
L1i cache:                            192 KiB (4 instances)
L2 cache:                             1 MiB (1 instance)
Boot Time:
Startup finished in 3.999s (kernel) + 12.268s (userspace) = 16.268s
graphical.target reached after 12.200s in userspace.
Compiler Version:
gcc (Debian 12.2.0-14) 12.2.0
Copyright (C) 2022 Free Software Foundation, Inc.
This is free software; see the source for copying conditions.  There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.

Memory Copy Speed:
1000+0 records in
1000+0 records out
1048576000 bytes (1.0 GB, 1000 MiB) copied, 0.351019 s, 3.0 GB/s
Filesystem Write Speed:
1000+0 records in
1000+0 records out
1048576000 bytes (1.0 GB, 1000 MiB) copied, 27.6191 s, 38.0 MB/s
Power Consumption and Temperature:
volt=0.9360V
temp=44.3'C
master@starry:~/Desktop/cpu_info $
```

2. Software Characterization

What is the name and release version of the Operating System used on the target board?

The Raspberry Pi 4 officially supports **Raspberry Pi OS** (formerly Raspbian). Other supported OS options include Ubuntu, Arch Linux Mint and many more custom distros. The latest Raspberry Pi OS version: **Bookworm (Debian 12-based)** was installed for this project.

What is the compiler name and version?

The default compiler is **GCC (GNU Compiler Collection)**. The GCC version can be verified by using the below mentioned command:

```
gcc --version
```

What software is available to enhance the pins? Test the software?

There are several software libraries enable GPIO control:

1. **pigpio** – Low-level control with hardware PWM support.
2. **RPi.GPIO** – Simple Python library for controlling GPIO.
3. **wiringPi** – C-based GPIO library (deprecated but still used).
4. **libgpiod** – Modern GPIO control via device tree interface.

I have simply relied on RPi.GPIO for my use case and can confirm that it works for RPI4.

Testing GPIO output with Python:

```
import RPi.GPIO as GPIO
GPIO.setmode(GPIO.BCM)
GPIO.setup(18, GPIO.OUT)
GPIO.output(18, GPIO.HIGH)
```

Thinking about the complications for mass manufacturing and production, things are a bit different for limited units (around 1000) vs mass production (more than 10000). I gathered few comparisons as written below

Aspect	**Limited Production (1000 units)**	**Mass Production (Over 10,000 units)**
Cost Efficiency	Higher cost per unit due to smaller scale production.	Lower cost per unit due to economies of scale.
Customization	Greater flexibility for customization and tweaks.	Limited customization; focus on standardization for efficiency.
Supply Chain/Lead Time	Relatively short lead time, but can vary with demand.	Complex supply chain; requires careful management for component sourcing.
Support & Maintenance	Strong community support; need for custom maintenance solutions.	Dedicated long-term support required; more formal warranty systems.

Quality Control	Basic quality checks; less consistency needed.	Rigorous quality control processes and automated testing.
Environmental Concerns	Minimal, usually not a major factor.	Must consider power consumption, cooling, and sustainability at scale.

Conclusion

This characterization of the Raspberry Pi 4 provides insight into its **hardware** (SoC, memory, GPIO, boot process) and **software** (OS, compiler, GPIO libraries, real-time capabilities). Understanding these aspects is essential for embedded system development on the Raspberry Pi 4.

References:

- https://www.raspberrypi.com/documentation/computers/raspberry-pi.html#eeprom-boot-flow
- https://forums.raspberrypi.com/viewtopic.php?t=313463

Time Taken for Report

The exploration and characterization took approximately 10 hours, which includes research, testing, and documentation. Time spent on developing code and scripts for hardware characterization was around 4-6 hours.

Scripts and test code for the sensor is provided with the document.