Microprocessor Architectures for Embedded Systems

Carlos Rijo[1]

[1] *Instituto Superior de Engenharia do Porto*

Abstract

Since its first development, microprocessors have evolved greatly improving on their characteristic, performance and features. With this evolution, the microprocessors stared to be also implemented in embedded systems of personal or industrial purposes and used in development boards that are used as hobby projects, teaching tools and prototype development.
Microprocessors can be found in common devices like: Personal Computers, Network and Internet of Things (IoT) devices, Mobile devices, Gaming consoles and Digital cameras.
Overall, microprocessors play a critical role in most every system we interact today.
This paper will address the implementation and usage of microprocessors in embedded systems.

Keywords: *Microprocessor, Embedded Systems, Development boards.*

1 Introduction

The first microprocessor developed was the Intel 4004 in 1971 by Intel Corporation. It was relatively simple and had limited capabilities with 4-bit microprocessor that was used in calculators and other small devices.

However, they paved the way for the development of more advanced microprocessors that have revolutionized the field of computer technology.

In the 1980s, the microprocessor industry underwent significant growth, with the development of 32-bit microprocessors and, as they offered improved performance compared to earlier microprocessors, they began to be used in personal computers and other devices. The microprocessors continued its evolution and in the 1990s and 2000s it was developed 64-bit microprocessors and the introduced multi-core processors.

Nowadays microprocessors are an essential component of modern computing and can found in almost all electronic devices such as Personal Computers (PC), Network and Internet of Things(IoT) devices, mobile devices (smartphones, smartwatches, tablets), gaming consoles, televisions, home appliances, digital cameras and many others. They also play an important role in fields like automobile, medical, traffic control, house automation and security, telecommunications, transportation and manufacturing.

The development and evolution of microprocessors not only increased the power and performance of our personal computers, but also increased the number of gadgets that we use on a daily basis, improved long-range communications and increased our safety in our daily lives.

2 Microprocessors

A microprocessor or MPU (MicroProcessor Unit) is a Central Processing Unit (CPU) that executes a predefined set of instructions continuously and performs arithmetic and logical operations on the received data. It can be defined as a single integrated circuit (IC) in which are implemented the core elements of a computer system such as:

- Arithmetic Logic Unit (ALU): Executes arithmetic and logical operations with the input data from a device or from a memory;
- Control Unit: Controls the instructions and flow of data within the computer;
- Register array: In which includes registers;
- Memory Unit: Data storage;
- Set of transistors to carry out the instructions of the microprocessor.

This standard functionalities are shared among all microprocessors but they can vary depending on their characteristics in terms of their architecture, performance, power consumption and features.

2.1 Architecture

Microprocessors can have different architectures. Each architecture will define the Instruction Set Architecture (ISA), memory architecture, data paths, registers, and control unit.

The ISA of a microprocessor, defined by the manufacturer, is a set of rules that define the range of instructions that a microprocessor is able to execute and, therefore, every program that runs in it is bound and encoded by the defined instruction set architecture.
This definition will determinate the capabilities of the processor (and its performance) and the types of tasks it is able to execute.
Some common microprocessor architectures are: x86, ARM, MIPS, and PowerPC.

2.1.1 Von Neumann architecture

Developed by John von Neumann, this architecture is based on the concept of separating the memory and the processing unit. This microprocessors have a single set or registers for both instructions and data which allows the instructions to be fetched (from the memory), executed and then store the results back in the memory. Some common examples of Von Neumann microprocessors are:

- Intel Core
- AMD Ryzen
- Qualcomm Snapdragon
- ARM Cortex
- IBM Power

2.1.2 Harvard architecture

This microprocessor architecture separates into two separate and distinct spaces the memory space dedicated to instructions from the memory which allows the access instructions memory and data memory simultaneously, increasing the efficiency of the processor. The efficiency of the processor can be increased further by implementing two separate buses for the instruction and data memory allowing the access to both memories almost simultaneously. The Harvard architecture is often used in microcontrollers like:

- PIC microcontrollers
- 8051 microcontrollers
- MSP430 microcontrollers
- AVR microcontrollers

2.1.3 RISC architecture

The RISC (Reduced Instruction Set Computing) architecture has as its goal the improvement of its performance by using a reduced number of simpler instruction sets, meaning that the basic instructions set that it can execute are smaller (compared to other architectures) but they can be executed faster.

Another characteristic of RISC architectures is that its instructions have a fixed length (typically 32 bits or 64 bits) which allows encoding a set of different instructions, while still keeping the instructions simple and easy to decode and execute. And since this architecture has a large number of registers (memory dedicated to store data temporally), the use of memory is more efficient since it reduces need to access the main memory and the number of transitions.

Within RISC architecture there are several architectures (with its respective ISAs) with different characteristics. Here hare some examples worth mentioning:

- ARM (Advanced RISC Machine) - Used mainly on portable and embedded systems
  - Low power consumption
  - High performance
  - Versatility
- MIPS (Microprocessor without Interlocked Pipeline Stages) - Used mainly on network devices
  - Low power consumption
  - High performance
  - Versatility
- PowerPC - Used mainly on embedded systems
  - High performance
  - Versatility
- RISC-V - Open-source. Used mainly on high-performance computing and embedded systems
  - High performance
  - Modular and Customizable
  - Open Source

2.1.4 CISC architecture

CISC (Complex Instruction Set Computing) is an architecture that uses a larger and more complex instruction sets that allows the microprocessor to have a wider range of functionalities and perform multiple operations in a single instruction. CISC microprocessors usually are more powerful since they can perform more complex tasks but that may result in a slower and less efficient performance due to the increased complexity..

This architecture is generally easier to program since there are more compatible with high-level programming languages.

Within CISC architecture there are two notable architectures (with its respective ISAs) worth mentioning:

- Intel x86 processors
- AMD processors

2.2 Performance

As explained before each architecture have different efficiencies regarding its own instruction set so, depending on their architecture, microprocessors can vary significantly in terms of their performance. It can be measured by analysing:

- Clock speed - The number of instructions they can execute per second. Higher clock speed means higher performance but also meas higher power consumption
- Number of cores - The number of independent processing units. With more cores it is possible to perform multiple tasks concurrently and improve workloads.
- Cache size - Cache is a small memory that stores frequently used instructions and data. Higher cache size means reducing the number of times the microprocessor needs to access main memory.

2.3 Power consumption

When designing or implementing devices that are powered by a limited power source, like batteries, the power consumption of a microprocessor is an important consideration to have.

Discarding the rest of the system and focusing only on the microprocessor, the chosen architecture, its clock speed and its workload will define the power load of the microprocessor which, indirectly, will dictated the power consumption of the system.

2.4 Features

Along side its core elements (ALU, Control unit, register array and memory unit, microprocessors can also have more features, such as: support for external memory, wired and wireless communications, access to General Purpose inputs and Outputs (GPIOs) and support for different types of peripherals.

Given so, the next list will enumerate some examples of microprocessors with different features:

1. System on chip (SoC) - Designed to contain all the components needed to build a system, including the additional functionality, that would normally be provided by auxiliary components, for example:
   - Wired and wireless communications
   - General Purpose inputs and Outputs (GPIOs)
   - External memory,
   - Analog-to-Digital Converter (ADC)
   - Digital-to-Analog Converter(DAC)
2. Microcontroller (MCU) - Designed to contain a few additional components such as RAM, ROM, and programmable GPIOs.
3. Digital Signal Processor (DSP) - Designed specifically to process digital signals like audio and video.
4. Floating point unit (FPU) - Designed specifically to create and manipulate floating point values.
5. Graphics Processing Unit (GPU) - Designed specifically to handle graphics-intensive tasks, such as the creation and manipulation of images through a set of optimized geometric operations.
6. Field-Programmable Gate Arrays (FPGAs): Designed to be programmed to perform a specific task using a Hardware Description Languages (HDL). Composed by an array of programmable logic blocks that can be connected together to achieve the desired system and so FPGAs can be easily re-programmed.

2.5 Temporal restriction

In a system with temporal restrictions, the microprocessor is responsible for executing instructions and performing tasks within the required time frame. This may require the increase of clock speed and scheduling techniques to ensure that the most important tasks are completed on time.

There are two main types of real-time systems: hard real-time systems and soft real-time systems.

In a soft real-time system, the microprocessor is responsible for executing instructions and performing tasks within a certain time frame, but missing a deadline may not have serious consequences since sometimes it can be managed or compensated for.

In a hard real-time system, the microprocessor must execute instructions and perform tasks with a high degree of predictability and reliability. If a task misses a deadline it can result in a failure of the entire system.

3 Implementation in Embedded Systems

Embedded systems can be considered as a subpart of a larger system. They can defined as the main system or as secondary within the system and are usually designed to perform a specific set of tasks.
Normally, they are designed to have: a controller unit (microprocessor), a power management system, a memory storage, GPIOs interfaces, communication interfaces and other peripherals.

Microprocessors can be used in embedded systems since they provide the processing power and flexibility needed to perform the tasks required by the system. They can be programmed to execute specific instructions and perform a variety of different functions, such as data processing, communication, and control.

To develop an embedded system has usually five steps: Requirements gathering, System design, Implementation, Testing and Deployment.

3.1 Requirements gathering

At the start of the development of an embedded system the requirements must be gather and defined. The requirements should address the system constraints, such as:

- Tasks to execute
- Performance
- Response in real-time to internal or external events
- Power consumption and delivery
- Size
- Cost
- Communications (wired or wireless) and respective protocols
- I/O integration (sensors and actuators)
- Programming language
- Memory usage
- User safety

3.2 System design

The next step should to design the overall system. In this step it must be designed the hardware and software components, and the development of any necessary peripherals or I/O interfaces.

The selection of the microprocessor must be included having in mind the requirements defined previously and the microprocessor functionalities (architecture, performance, power consumption and features).

Having in mind the topics explained in the previous chapter, the microprocessors architecture (and the respective ISA) must be the first characteristic to decide since. This selection will determinate the first power consumption of the system and what micro-architecture is going to be used: a microprocessor unit (MPU), a microcontroller unit (MCU) or a FPGA for example.

Then, the performance of the microprocessor should be studied analysing its clock speed range, number of cores and possible cache size. This will allow to determinate a possible workload of the system.

And lastly the features of the microprocessor. According to the requirements it may be possible to choose one or more ICs that correspond to the system needs.

In case the system has temporal restrictions is necessary to select a microprocessor with a high clock speed and a suitable micro-architecture to ensure that the microprocessor is able to meet the temporal restrictions of the system.

The key challenges in choosing the microprocessors, when designing the system, is to balance the performance and power consumption of the microprocessors.

3.3 Implementation

After the system has been designed, the next step is to implement the hardware components, by create the first prototypes and software components, by writing the software code.

Since it is difficult to test standalone microprocessors there were created development boards that can be be easily programmed and tested before they are integrated into a larger system. This boards integrate microprocessors with other components such as memory, I/O interfaces, and other peripherals to create a simple, but complete, systems in a single "off-the-shelf" board. They include a variety of tools and libraries for debugging and are also prepared to be programmed by high level programming languages including C, C++ and Python facilitating the first steps of prototyping.

There are many different types of microprocessor development boards available. They vary mainly in two points: the microprocessor architecture used and the peripheral interface available. Some common development boards are:

- Arduino boards - AVR architecture based on RISC and Harvard architecture
- ESP32 boards - RISC-V architecture
- Raspberry Pi boards - ARM architecture based on RISC architecture
- Jetson Nano boards - ARM architecture based on RISC architecture
- Odroid XU4 - MIPS architecture based on RISC architecture

It is important to note that even though microprocessors are able to meet temporal restrictions not all available development boards can meet this request. They need to have the necessary processing power and real-time capabilities like the clock speed and number of cores of the CPU, the amount of memory available and, in case of single board computers, the real-time capabilities of the operating system being used. And since not all microprocessors have development boards, the selection of a board for prototype purposes must be carefully analyzed.

3.4 Testing

Once the system has been implemented, it is important to thoroughly test both hardware and software components to check if the system performs as expected. The main test that can be performed are:

1. Reliability Test - by overflowing the workflow of the microprocessor and evaluate the ability to handle high levels of workload (more tasks or data than it can handle), as well as its ability to recover from errors that may occur.
    - Data processing: Reading and/or writing data from and to memory, performing calculations, and manipulating data in various ways.
    - Communication: Transmitting and receiving data over various types of communication interfaces.
    - Control: Controlling and monitoring external devices (sensors or actuators)
2. Robustness Test - by making the system reach harsh environments and conditions
    - Temperature cycling: Exposing the microprocessor to a range of temperatures to ensure that it can operate reliably over a wide temperature range.
    - Humidity test: Exposing the microprocessor to high levels of humidity to ensure that it is resistant to moisture and corrosion.
    - Vibration and Shock test: Subjecting the microprocessor to mechanical vibrations and impact forces to ensure that it can withstand the stresses of operating in a dynamic environment.
    - Power cycling: Switching, repeatedly, the microprocessor On and Off to ensure that it can operate reliably over an extended period of time.

3.5 Deployment

After the embedded system has been tested and can perform the defined system requirements it is ready to be deployed and integrated in the system.

4 Conclusion

Even thought microprocessors share the same core components (ALU, control unit, register array and memory unit) each architecture has its own characteristics, instructions sets and features making the microprocessors different from each other. This opens a large number of possibilities to use microprocessors. For example, microprocessors with CISC architecture have a large and more complex instructions set giving them a large set of applications and compatibility with software which makes the microprocessor a good option for servers and desktop computers like Intel's x86 microprocessors (Intel Core, and Intel Xeon processors).

Another example is RISC architecture that, due to being more efficient because they have smaller and simpler instructions sets that can be decoded and executed more easily, has been replacing CISC microprocessors in modern systems. Since microprocessors with RISC architecture are more efficient, have high performance and have low power consumption, they can be used in portable devices (smartphones and tablets), servers, and in systems with temporal restrictions like automobiles, appliances, and industrial control systems.

But microprocessor architectures are not mutually exclusive since a microprocessor can have characteristics of more than on type of architecture. This "hybrid" architecture can provide more balance of performance and increase the range of applicability. Some examples of "hybrid" architectures:

- RISC-Harvard architecture - RISC-based core combined with separate memory spaces for instructions and data
    - AVR architecture used in Arduino boards
- RISC-CISC architecture - RISC-based core combined with a larger set of instructions that are more complex

    - Intel Pentium processors

- RISC-Von Neumann architecture - RISC-based core combined with a single memory space for both instructions and data.

    - ARM Cortex-A processors

So, microprocessors play an important role in embedded systems since the will controller unit of a partial part of the system or of the full system. Therefore the selection of the microprocessor must have in mind the system requisites and the microprocessor architecture and features so that the system can have the expected behaviour.

To help the selection of the microprocessor, there are available multiple development boards that can test the performance of the microprocessor along side external memories, I/Os interfaces and communication interfaces and ease the prototyping and system design phases.

References

[1] *Microprocessors: the engines of the digital age* Steve Furber. 2017 Royal Society.

[2] *The evolution of computer architectures* L. Dadda. 1991 IEEE.

[3] *John von Neumann and the Evolutionary Growth of Complexity: Looking Backward, Looking Forward* Barry McMullin. 2000 IEEE.

[4] *Microprocessors/MCUs for Internet of Things* João Goes. 2017 IEEE.

[5] *Microprocessor Controllers of Automatic Systems and Automatic Control Systems* Valeria P. Safronova, Marina V. Yakovleva. 2019 IEEE.

[6] *Survey on Microprocessor Architecture and Development Trends* YaoYingbiao, Zhang Jianwu. 2008 IEEE.