

# Microcontroladores

Matheus Dias

## 1 Introdução

Na eletronica digital, categorizamos em dois tipos os sistemas digitais: **circuitos combinacionais e sequêciais**. Os circuitos combinacionais são compostos por um agregado de portas lógicas, onde o valor da saída depende exclusivamente do valor da ultima entrada. Os círcuitos sequênciais, são compostos por circuitos combinacionais e elementos de memória (flip-flops), pois sua saída depende das entradas posteriores.

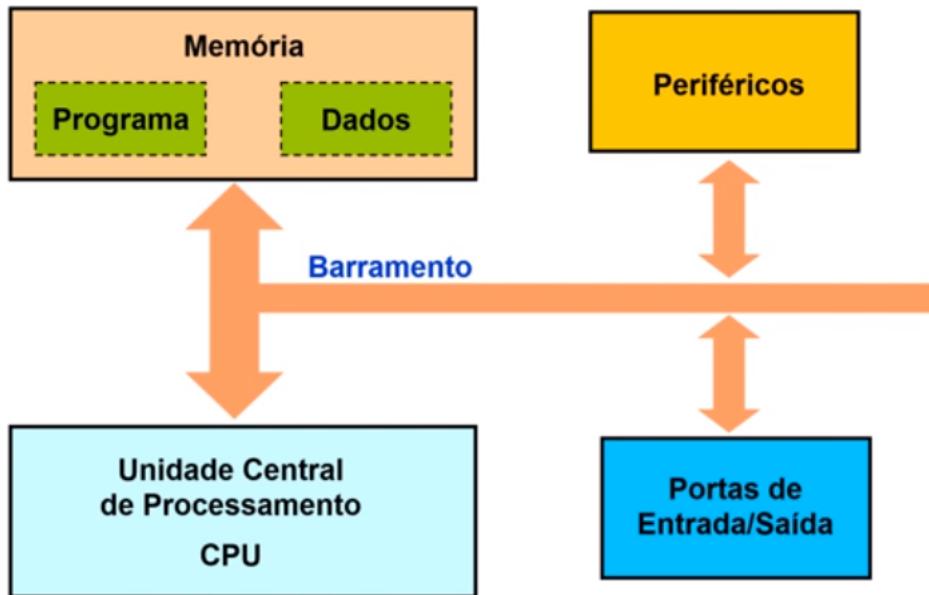
Sistemas processados, sistemas embutidos ou **sistemas embarcados**, é o nome que se dá para o conjunto de sistemas que possuem circuitos combinacionais, sequênciais e firmware (UEFI/BIOS). Esses sistemas tem a capacidade de realizar uma sequência programada de operações, ou seja, podemos **executar um programa**. Um sistema embarcado garante a vantagem de para um mesmo circuito, existir diversas aplicações distintas, pois sua configuração é feita via software, promovendo maior usabilidade. Não precisamos definir em hardware qual funcionalidade o microcontrolador deve ter. Importante destacar o conceito de firmware: software de baixo nível que configura diretamente o hardware, através de registradores, periféricos, etc. No contexto de computadores pessoais, é o primeiro programa carregado na memória que 'ensina' a CPU se comunicar com outras interfaces e carregar o sistema operacional.

Segue alguns exemplos de aplicações:

- Videogames;
- Calculadoras;
- Telefone Celular;
- Equipamentos de medição;
- Automóveis, caminhões, ônibus, etc;
- Equipamentos médicos;
- Ávionicos;

## 2 Arquitetura básica de um MCU

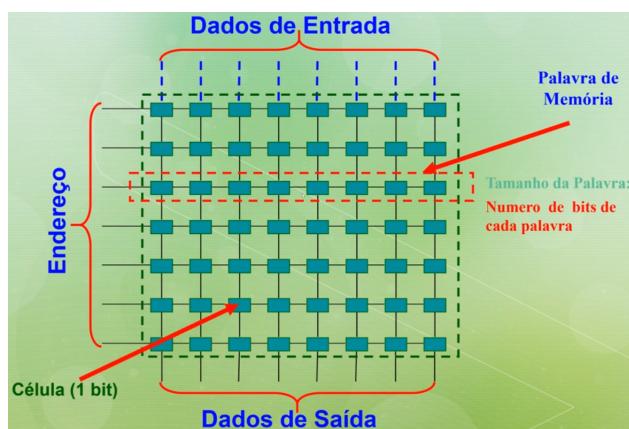
A estrutura básica de um microcontrolador:



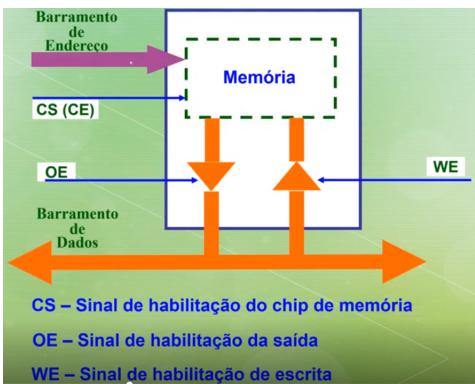
Qualquer entidade nominada de microcontrolador, contemplará esses blocos fundamentais encapsulados. A diferença entre um microcontrolador e um microprocessador, é que o ultimo, possuí somente a CPU implementada, fazendo-se necessário circuitaria adicional para ter funcionalidade.

### 2.1 Memória

A memória equivale a uma tabela composta de células, onde cada uma das células representa um único bit. Para cada linha ou palavra da memória, vemos 8 bits ou 8 celulas que 'seguram' um estado lógico. Cada uma dessas linhas possuem um endereço. Também vemos os dados de entrada e saída.



Um exemplo de chip de memória é o M28C64 EEPROM. Este, possuí 64 KBytes, ou seja, 8k vezes 8: oito mil endereços ou 8 mil linhas, onde cada palavra tem 1 Byte ou 8 bits. Vale mencionar os sinais de controle de uma memória encontrada em MCUs.



O sinal **CS** gerencia e habilita a memória para o barramento, em caso de haver mais de um chip de memoria. O sinal **OE** faz o conteúdo de um endereço de memória ser 'jogada' no barramento de dados. O sinal **WE** carrega o conteúdo do barramento de dados para um determinado endereço. Ou seja, OE lê o conteúdo da memória e WE escreve um conteúdo na memória, dado o conteúdo do barramento de endereço. Segue um esboço dos tipos de memória.

### Memória de programa

Memória de programa é onde o programa compilado é armazenado, ou seja, onde a sequência de operações que o MCU vai executar, fica guardado. Essa memória é do tipo não-volátil, portanto, não se perde quando o sistema é desligado.

- **ROM - Read Only Memory:** Memória somente de leitura, sendo gravada pelo fabricante do componente durante o processo de fabricação. Uma vez gravada, não pode mais ser modificada. É também chamada Memória Programada por Máscara (Masked ROM). Um negativo fotográfico, denominado máscara, é utilizado para criar as ligações elétricas no chip.
- **PROM - Programmable Read Only Memory:** Memória somente de leitura que pode ser programada pelo usuário em campo somente uma vez. Esse tipo de memória é também chamada OTP-ROM, que significa One Time Programmable ROM. Equivale a fusíveis ligados às células da memória onde quando queimado, grava o estado lógico na célula definitivamente. A vantagem é não precisar passar o código para o fabricante da memória.
- **EPROM - Eraseable Programmable Read Only Memory:** Memória somente de leitura, programável, que pode ser apagada através de luz ultravioleta. O “apagamento” é feito através da incidência de luz ultravioleta em uma janela de cristal existente no chip. Para apagar e gravar a memória, o chip deve ser retirado do circuito.
- **EEPROM - Electrically Eraseable Programmable Read Only Memory:** Memória somente de leitura, programável, que pode ser apagada eletricamente. O “apagamento” é feito através de pulsos elétricos. A programação pode ser feita sem retirar o chip do circuito. Possui menor densidade do que a EPROM, além de custo mais elevado. Os microcontroladores possuem esse tipo de memória de programa para gravar dados do programa.
- **FLASH EEPROM:** Memória somente de leitura programável, com gravação e apagamento através de pulsos elétricos, no próprio circuito (in-circuit). Possui densidade e custos próximos da EPROM. São chamadas de flash devido ao apagamento e programação em blocos. A cada 512 Bytes se apaga ou se grava. É o tipo usado para memória de programa nos microcontroladores. Presentes em SSD, pendrive e cartão de memória.

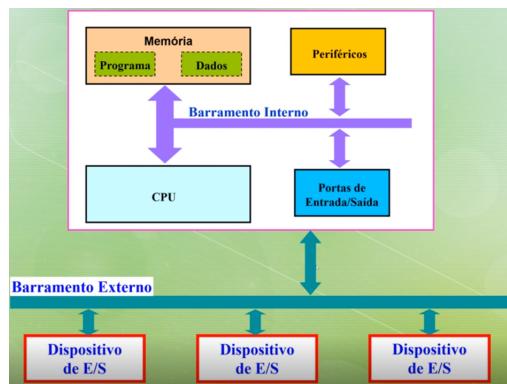
- **FRAM - Ferroelectric Random Access Memory:** Alta Velocidade de escrita, mil vezes mais rápido que a Flash EEPROM. Baixo consumo energético. Maior Confiabilidade com mais de cem trilhões de ciclos.

## Memória de dados

- **RAM - Random Access Memory:** Memória de trabalho, utilizada para armazenar temporariamente os dados utilizados pelo processador durante o tempo de execução do programa. É uma memória volátil, ou seja, os dados são perdidos quando a alimentação é retirada. Nos microcontroladores de um modo geral, são da ordem de 1 kbytes a 16 kbytes.

## 2.2 Barramentos e periféricos

Barramento é o meio físico ao qual trafega bits. O número de bits ou largura do barramento, determina a arquitetura do MCU. A interface entre o barramento interno do MCU e o externo é feito pelas **portas** de entrada e saída de um determinado microcontrolador. Para o Atmega328P, temos o **PORTC**, **PORTB** e **PORTD**, onde cada um tem seus respectivos periféricos "pendurados" ao qual o desenvolvedor pode usar em seu projeto.



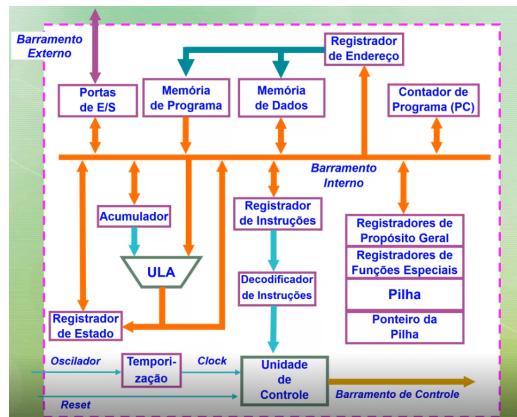
A principal vantagem dos periféricos é proporcionar funcionalidades sem consumir ciclos de máquina. Segue uma lista dos principais periféricos:

- Temporizadores/Contadores;
- Watchdog Timer: Contador que reseta o microcontrolador em caso de funcionamento irregular. O chamado "cão de guarda" que possuí clock a parte da CPU. Dentro do meu programa eu devo 'limpar' a contagem com certa frequência para não ocorrer o estouro do watchdog timer de maneira indevida;
- Conversores A/D;
- Conversores D/A;
- Osciladores (geradores de clock) internos;
- Controladores de LCD;

- Comparadores Analógicos;
- Relógio de Tempo Real (RTCC);
- Controladores de PWM (Pulse Width Modulator);
- Controladores de Interface Serial;
- UART (Universal Asynchronous Receiver / Transmitter): USB (Universal Serial Bus); CAN (Controller Area Network); SPI (Serial Peripheral Interface); I2C (Inter Integrated Circuit);

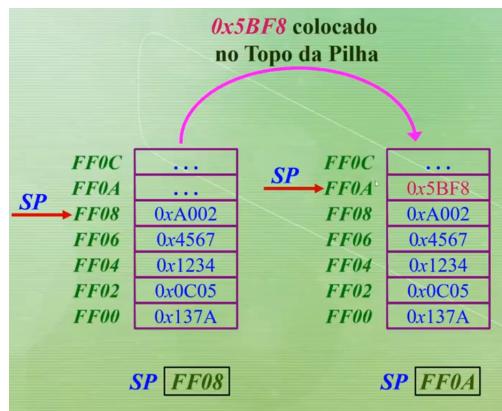
## 2.3 CPU

A CPU é o núcleo cordenador dos demais componentes do sistema embarcado/processado que atua em função do firmware. Cada microcontrolador usa uma CPU de diferente arquitetura a depender da família do mesmo, porém, os componentes e funcionamento básico pode ser generalizado. Note que todos os componentes mostrados na imagem a seguir estão encapsulados: define microcontrolador.



- **Memória de programa:** Tipo não volátil, onde se armazena o firmware. No atmega328P existe 32 kBytes. Em termos de microcontralrdores, é um valor elevado, pois a programação é feita em função dos **registradores**, não existindo um sistema operacional de interface entre aplicação e hardware.
- **Memória de dados:** Tipo volátil, onde valores de variáveis são armazenadas em tempo de execução. No atmega328P existe 2kBytes.
- **Registrador de endereços:** Contém o endereço de memória de programa ou de dados a ser lido ou escrito. Esse endereço trabalha junto ao barramento de endereço e com os sinais de controle da memória envolvida. O conteúdo é jogado do barramento interno de dados para a região de memória do endereço, ou o conteúdo do endereço daquela região de memória é jogado para o barramento interno de dados, a depender do sinal **WE** ou **OE**.
- **Portas I/O:** Interface entre o "mundo externo" e a CPU. No atmega328P temos 23 pinos I/O, distribuídos entre PORTB, PORTC e PORTD. Observação: podemos notar que na relação **pin-out** do arduino, não temos disponível os pinos 'PB6' e 'PB7', pois ambos estão sendo usados para fonte de clock externo de 16 MHz e o PC6 como reset.

- **Unidade lógica aritmética e acumuladores:** Conjunto de circuitos combinacionais que tem o propósito de realizar operações à nível digital. Sua memória de trabalho de alta velocidade são os acumuladores, que são registradores de 8 ou 16 bits. No atmega328P são totalizados em 32 registradores, que também desempenham papel de registradores de propósito geral. As etapas de uma operação que a ULA realiza, depende do armazenamento nesses registradores.
- **Registrador de estado:** O *status register* é um registrador de importância para o desenvolver. Esse registrador possui Flags que indicam o estado da ultima operação realizada pela ULA. Um exemplo é o bit 7, que habilita/desabilita a chamada de interrupções. Outro exemplo é o bit 0, que indica o resultado de uma comparação: se os valores forem iguais a flag é zero, ou seja, o bit está 'setado' como um.
- **Registrador de instrução e decodificador de instrução:** O registrador de instrução armazena de forma 'crua' a instrução qual foi lida da memoria de programa. É a instrução executada pelo processador no momento. O decodificador, decodifica o código da instrução que será executada no instante.
- **Unidade de controle:** Após a instrução ser decodificada, ela irá alimentar a unidade de controle, bloco que cobra a execução, disparando os sinais de controle necessários para o processador. Resumindo, a instrução armazenada, proveniente do firmware, só pode ser interpretada pela unidade de controle após ser decodificada. É nessa etapa que o processador responde ao código-fonte desenvolvido.
- **Temporizador:** Sincroniza os componentes do processador.
- **Pilha:** Componente importante em caso de chamada de subrotina, que podem ser interrupções via hardware. Nessas situações, o endereço de retorno para rotina principal é armazenado no topo da pilha, para quando a subrotina encerrar-se, a execução possa voltar a sequência natural do programa. A pilha é implementada na memória SDRAM do Atmega328P, isso influencia como o ponteiro da pilha atua.
- **Ponteiro da pilha:** O **stack pointer** ou ponteiro da pilha, contém o endereço do topo da pilha. Notamos que o endereço da proxima instrução da rotina natural é guardada no topo da pilha quando ocorre a chamada de subrotina, que por sua vez, culmina no incremento do stack pointer. Interessante ressaltar que o endereço para qual o stack pointer aponta, tem como conteúdo o endereço da proxima instrução a ser executada na rotina natural da CPU, ou seja, a próxima instrução que deverá ser decodificada para que a unidade de controle dispare os sinais que vão ordenar a execução da instrução, quando a subrotina se encerrar.



- **Registradores de função específica:** São registradores de configuração dos periféricos.
- **Registradores de propósito geral:** São os registradores do acumulador, para a arquitetura do Atmega328P.
- **Contador de programa:** Armazena o endereço da próxima instrução a ser executada. Ao se resetar o microcontrolador, o **PC**, **Program Counter ou contador de programa** é carregado o **Entry Point**, que é um endereço específico para cada arquitetura, no caso do Atmega328P, é o 0x0000.

### 3 Ciclo de máquina

O ciclo de máquina é dividido em três etapas: **ler, decodificar e executar a instrução**. Um ciclo de máquina pode consumir uma ou mais **ciclos de clock**, que no caso do Atmega328P é de 16 Mhz, quando alimentado com um cristal de clock externo. Para o atmega328P, um ciclo de máquina consome um ciclo de clock para quase todas as instruções.



As etapas do ciclo de máquina:

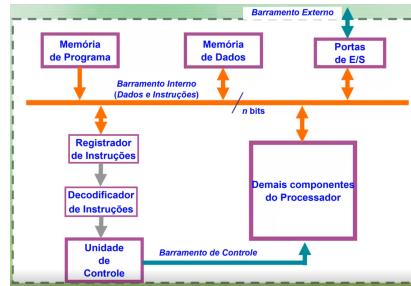
- Endereço do Program Counter é carregado para o barramento interno.
- Do barramento, a instrução é carregada para o registrador de endereço, ficando disponível no barramento de endereço de programa.
- O conteúdo do endereço da memória de programa é carregado no barramento interno da CPU, que por sua vez, é carregado no registrador de instrução. O código de operação da instrução a ser executada é transferida para o decodificador de instrução.
- O contador de programa é incrementado.

# 4 Arquiteturas

Existem dois tipos gerais de arquitetura.

## 4.1 Von-Neuman

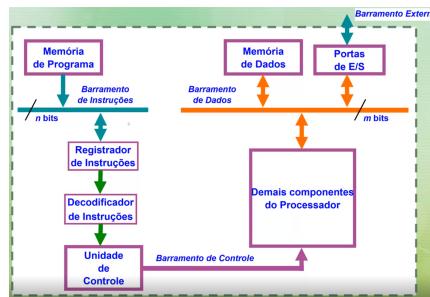
Para a arquitetura de Von-Neuman, existe somente um único barramento para dados e instruções que é compartilhado com os componentes básicos que constituem a CPU.



Devido ao barramento compartilhado, o ciclo de busca, decodificação e execução é feita sem sobreposição, ou seja, as etapas são processadas uma de cada vez.

## 4.2 Harvard

Para a arquitetura de Harvard, existe um barramento para instrução e outro distinto para dados.



Devido a quantidade de barramentos, o ciclo de busca, decodificação e execução é feita com sobreposição, ou seja, as etapas são processadas ao mesmo tempo. Sobreposição nesse contexto é chamado de **pipeline**.