Capítulo 1

Introdução aos Microprocessadores

1.1. Introdução

Um microprocessador (µP), ou simplesmente processador, é uma das principais inovações tecnológicas da eletrônica desde a aparição do transistor em 1948. Esse dispositivo não somente se estabeleceu no processo revolucionário da eletrônica digital, mas também está presente em quase todas as esferas da vida humana. As aplicações dos microprocessadores vão desde controladores de processos industriais, smartphones, calculadoras de bolso, máquinas fotográficas, *wearables*, equipamentos médico-hospitalares, satélites, radares, robôs, instrumentos de medição, aeronaves, veículos automotores, equipamentos de áudio e vídeo, eletrodomésticos, equipamentos de supervisão, videogames a até brinquedos. É, portanto, imperativo para todo engenheiro, especialmente o engenheiro de computação, conhecer as potencialidades dos microprocessadores.

Todo projetista, ou analista de produtos eletrônicos digitais, precisa conhecer as peculiaridades do mundo dos microprocessadores. Mesmo que ele não tenha planos imediatos de usar um microprocessador, ele deve ter conhecimento do assunto para poder planejar, de maneira inteligente, seus futuros projetos e fazer julgamentos adequados de engenharia.

Nesse capítulo são apresentados os conceitos fundamentais do mundo dos microprocessadores, o que ele pode fazer, como se aplica em um sistema computacional, além de obter uma ideia geral da arquitetura interna desse importante dispositivo eletrônico.

Um microprocessador é um dispositivo multifuncional programável, sendo uma das unidades estruturais de um sistema computacional. Ele busca por instruções armazenadas em uma memória, decodifica e executa essas instruções que operam sobre dados digitais, também armazenados na memória. Ele executa, essencialmente, operações lógicas e aritméticas e controla a temporização e a operação geral de um sistema computacional. Também recebe dados de dispositivos de entrada e envia os resultados do processamento para dispositivos de saída, sendo, portanto, capaz de se comunicar com o ambiente externo. Por tudo isso, um microprocessador também é chamado de Unidade Central de Processamento (*CPU – Central Processing Unit*).

Atualmente, no mercado existe uma variedade muito grande de processadores para diferentes finalidades e com capacidades computacionais distintas. Neste mercado, são encontrados desde processadores de uso geral, como os mostrados na Figura 1, utilizados em desktops domésticos, até processadores de uso específico, como aqueles voltados para aplicações profissionais em grandes *mainframes*, processadores digitais de sinais analógicos (*DSP – Digital Signal Processors*), processadores de vídeo (*GPU – Graphic Processor Unit*), entre outros.

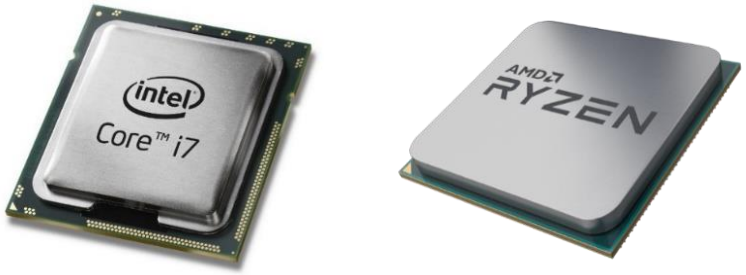

Figura 1 – Processadores comerciais de uso geral Intel core I7 e AMD RYZEN.

Podemos fazer uma analogia da função de um processador em um sistema computacional com o cérebro humano, pois todas as decisões são tomadas pela *CPU,* conforme suas instruções programadas e armazenadas na memória. Todas as outras partes de um sistema computacional também são controladas por esta unidade. Assim, diz-se que o processador é o cérebro do sistema computacional.

Um microprocessador é conectado à memória, onde se encontram armazenados os dados a serem processados e as instruções (programa), e aos dispositivos de entrada e saída (dispositivos de *I/O – Input/Output*) para formar o que convencionalmente chamamos de microcomputador, como mostrado na Figura 2.

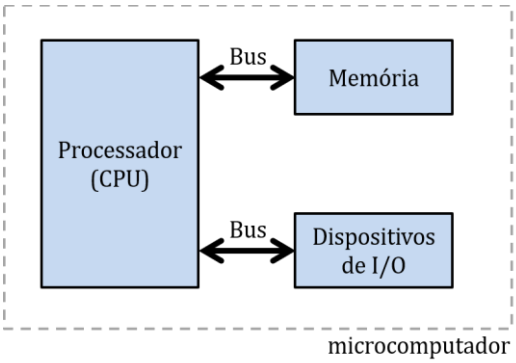

Figura 2 – Arquitetura de um microcomputador.

A conexão do processador com a memória e com os dispositivos de entrada e saída é feita mediante o uso de conjuntos de linhas (fios) que carregam informações e características similares. A esses conjuntos de linhas damos o nome de barramento (*Bus*).

Um microprocessador é caracterizado pelo "tamanho da palavra", que define a sua arquitetura. O termo "tamanho da palavra" significa a quantidade máxima de bits de dados que são processados simultaneamente pelo microprocessador, o que também especifica a largura do barramento de dados. Por exemplo, um processador pode ter tamanho de palavra de 4 bits, 8 bits, 16 bits, 32 bits, 64 bits, etc. Dessa forma, um microprocessador de 8 bits executa várias operações em dados de 8 bits. Atualmente, os processadores utilizados em desktops domésticos e smartphones utilizam, majoritariamente, uma arquitetura de 64 bits. Falaremos mais sobre isso adiante.

Os dispositivos de entrada são usados para fornecer dados à CPU. Exemplos desses dispositivos são teclado, mouse, botões, conversor analógico-digital, *scanner*, leitor de cartões, etc. Os dispositivos de saída são usados para fornecer os resultados do processamento ao mundo exterior. Exemplos são diodos emissores de luz (*LEDs*), displays e monitores em geral, conversor digital-analógico, impressora, plotter, etc. Os dispositivos de entrada e saída também são comumente chamados de periféricos, pois estão logicamente conectados no entorno da CPU.

1.2. Histórico da evolução dos microprocessadores

Os microprocessadores tornaram possível o advento do microcomputador em meados da década de 1970. Inicialmente, as *CPUs* eletrônicas eram tipicamente construídas com dispositivos discretos de comutação lenta, barulhentos e volumosos, como os relés eletromecânicos. Posteriormente, os relés foram gradativamente sendo substituídos pelas válvulas eletrônicas, que eram muito mais rápidas que os relés, não faziam barulho, mas, ainda assim, consumiam uma grande quantidade de energia. É dessa época o desenvolvimento do ENIAC (*Electronic Numerical Integrator and Computer*), o primeiro computador digital eletrônico de grande escala, usado pelo exército norte-americano para calcular trajetórias táticas que exigiam conhecimento substancial em matemática com maior agilidade, mas que só se tornou operacional após o final da segunda guerra mundial.

Em meados da década de 50, os transistores passaram a substituir com sucesso as válvulas, consumindo muito menos energia e aumentando a velocidade de processamento. Mais tarde, circuitos completos foram integrados em uma única pastilha de silício (*chip*) e foram usados para projetar as *CPUs*. Ao integrar o processador em um único chip (contendo o equivalente a milhares de transistores discretos), o custo, o tamanho e o consumo de energia dos processadores foram bastante reduzidos e a velocidade de processamento passou a aumentar significativamente.

O termo 'microprocessador' surgiu em 1971, quando a *Intel Corporation of America* desenvolveu o primeiro processador integrado em um único *chip*, o Intel 4004, mostrado na Figura 3, que é um microprocessador com arquitetura de 4 bits, que operava a uma frequência de clock de 750 kHz e era constituído por 2300 transistores. Ele tinha a capacidade de executar operações aritméticas e lógicas simples, como adição, subtração, comparação, AND e OR. Também possuía uma unidade que podia executar várias funções de controle, como buscar uma instrução na memória, decodificá-la e gerar sinais de controle para executá-la.

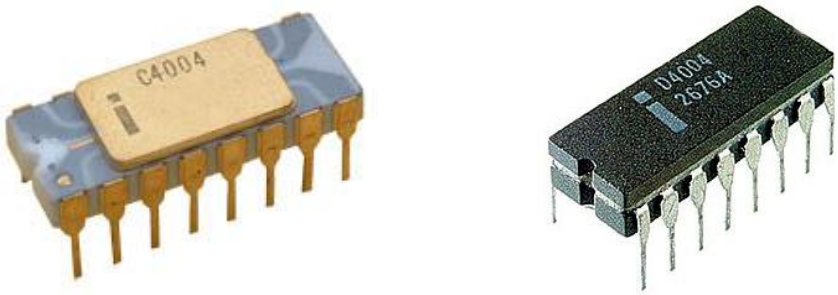

Figura 3 – Microprocessador Intel 4004 em dois tipos de encapsulamentos.

Para podermos perceber como esses dispositivos tiveram uma enorme evolução em menos de meio século de desenvolvimento, hoje podemos facilmente encontrar dispositivos com arquiteturas de 64 bits (ou até mais), que operam com frequências da ordem dos 5 GHz, constituídos por mais de 3,5 bilhões de transistores.

Na Figura 4, são apresentadas as etapas pelas quais a evolução do microprocessador passou até atingir o estado de desenvolvimento atual. Da esquerda para a direita, são apresentados: A válvula eletrônica, que fazia a comutação da corrente nos primeiros processadores digitais; O primeiro transistor; A integração dos circuitos do processador no chip do Intel 4004 e a integração no chip do Intel Core I7.

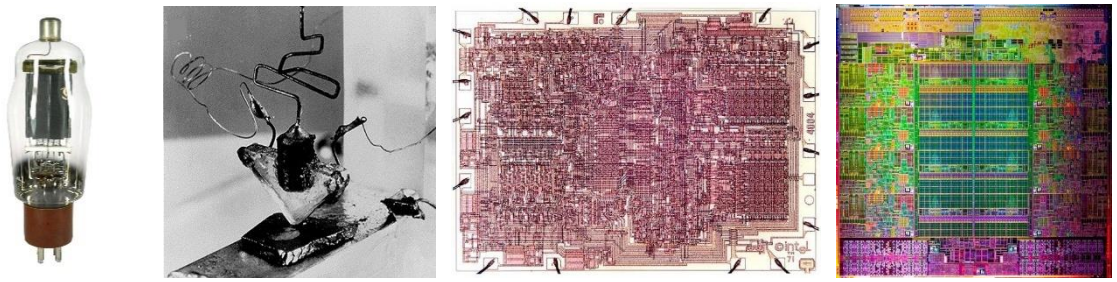

Figura 4 – Etapas de evolução do processador (da esquerda para a direita): válvula eletrônica; primeiro transistor; integração do circuito do processador no die do Intel 4004; integração no die do Intel Core I7.

Depois do 4004, foi desenvolvido o microprocessador Intel 4040 de 4 bits, uma versão aprimorada do seu antecessor. Muitas outras empresas como a Motorola, Fairchild, Texas Instruments, Zilog, National Semiconductor, entre outras, também entraram no mercado dos processadores e desenvolveram seus próprios *chips* com arquitetura de 4 bits.

Em 1972, a Intel lançou o primeiro microprocessador de 8 bits, o Intel 8008, que usava a tecnologia de fabricação de transistores de efeito de campo conhecida como PMOS. Os microprocessadores que usavam essa tecnologia eram lentos e não eram compatíveis com a lógica TTL, que eram circuitos que operavam com transistores bipolares e tensão de 5V. Em 1973, a Intel introduziu um microprocessador com tecnologia NMOS de 8 bits mais poderoso e mais rápido, o Intel 8080. Foi nessa época que a acirrada disputa comercial entre a Intel e a AMD teve início. Os microprocessadores de 8 bits ainda são amplamente utilizados até hoje, principalmente como núcleo de processamento de microcontroladores de baixo custo e baixo consumo de energia.

Em 1978, a Intel introduziu um microprocessador de 16 bits, o Intel 8086. Outros microprocessadores de 16 bits produzidos nessa época foram: Intel 80186, 8088, 80188, 80286; Zilog Z8000; Motorola 68000, 68010, 68012; Fairchild 9440, TMS 9900 da Texas Instruments, entre outros.

Após 1980, muitos fabricantes lançaram microprocessadores de 32 bits. O Motorola MC6800 foi um dos primeiros processadores dessa geração. Foi inicialmente utilizado em sistemas de alto-custo, incluindo microcomputadores multiusuários, estações de trabalho e terminais gráficos. A partir da metade da década de 1980, o MC68000 passou a ser empregado em computadores pessoais e domésticos, começando com o Apple Lisa e Macintosh, sendo precedido do Commodore Amiga, Atari ST e Sharp X68000. Diversos fabricantes de videogames passaram a utilizar esse processador como núcleo de diversos arcades e consoles de jogos eletrônicos, como as placas Sega System 16, Capcom CPS-1 e CPS-2 além do Neo-Geo da SNK. Os microprocessadores de 32 bits ainda são amplamente utilizados em computadores desktop, portáteis, notebooks, estações de trabalho e servidores.

Ao mesmo tempo, a Intel lançou o seu primeiro microprocessador de 32 bits, o iAPX 432, que não se tornou popular e acabou sendo descontinuado. A Intel, então, apresentou o poderoso 80386, seguido pelo 80486. Foram esses processadores que popularizaram a famosa arquitetura x86 da Intel que forma a base de muitos computadores pessoais e estações de trabalho até hoje.

Os microprocessadores com arquitetura de 64 bits foram introduzidos pela Intel em 1993 com o Pentium.

Com tecnologias melhores e mais avançadas, a velocidade dos microprocessadores aumentou consideravelmente. O antigo Intel 8085 de 1977 executava meio milhão de instruções por segundo (0,5 MIPS), enquanto o 80486, de 1989, executava 54 milhões de instruções por segundo. Outros processadores, como os processadores gráficos (*GPUs*) possuem ainda arquiteturas com barramento de dados de 128, 256 ou 512 bits, para permitir processamentos ainda mais rápidos.

A evolução da capacidade de processamento dos processadores foi possível devido à conjunção de várias técnicas de aperfeiçoamento de hardware, a saber: O aumento do tamanho da palavra, permitindo que dados com maiores quantidades de bits pudessem ser processados simultaneamente, ou fazendo com que uma instrução pudesse processar um número maior de dados básicos de cada vez; Redução do tamanho e aumento da quantidade de transistores no chip, expandindo a capacidade de processamento com o advento de novas unidades funcionais implementadas diretamente no hardware; Maior velocidade do clock, aumentando a quantidade de instruções executadas por segundo.

O aumento na velocidade do clock dos processadores foi uma estratégia de hardware utilizada durante muito tempo para garantir o aumento na capacidade de processamento. Entretanto, o aumento da quantidade de transistores confinados em áreas cada vez menores, associado ao aumento do clock, trouxe como efeito colateral um aumento da dissipação de energia em forma de calor, o que exigia sistemas de refrigeração mais robustos.

A obtenção de alto desempenho computacional não depende somente da utilização de dispositivos de hardware mais rápidos, mas também de melhorias na arquitetura dos processadores. Arquiteturas avançadas de computadores estão baseadas no conceito de processamento paralelo.

O processamento paralelo pode ser encontrado em diversos níveis na arquitetura de computadores. O nível mais baixo é conhecido como *pipeline*. A técnica de *pipeline* possibilita a execução paralela das microinstruções dentro do processador. Ou seja, ela permite uma sobreposição temporal das fases de execução de uma instrução. Diante do antigo paradigma onde cada instrução era executada separadamente, passando por todas as fases de execução no processador (por exemplo: leitura da memória principal, carregamento nos registradores, utilização da ULA para processamento, escrita no registrador de destino e escrita na memória principal), a utilização do *pipeline* divide esta execução em etapas permitindo a serialização no processamento das instruções. Em resumo, *pipeline* é o processo pelo qual uma instrução do processador é subdividida em etapas e, uma vez que cada uma destas etapas é executada por uma porção específica da *CPU*, pode-se colocar mais de uma instrução em execução simultânea. Isto traz um uso mais racional da capacidade computacional, com ganho substancial de velocidade. Entre os problemas enfrentados com essa técnica, estão a dependência de instruções anteriores e desvios no fluxo de execução, que dificultam o processo, bem como a diferença de complexidade de instruções que fazem com que as mesmas possam levar um tempo variável para execução.

O nível seguinte de paralelismo é o nível do chip, onde é possível ter mais de um núcleo de processamento, ou unidade funcional, realizando operações simultaneamente. Essa técnica é adotada nos processadores *Multicore*.

O multiprocessamento é o terceiro nível de paralelismo. Consiste no uso de duas ou mais unidades centrais de processamento dentro de um mesmo sistema computacional. O termo também se refere à capacidade de um sistema suportar mais de um processador ou à capacidade de alocar tarefas entre eles. Difere da multitarefa, pois esta simula a simultaneidade, utilizando-se de vários recursos, sendo o principal o compartilhamento de tempo de uso do processador entre vários processos. Nessa arquitetura, há um único espaço de endereçamento de memória que é gerenciado e compartilhado entre os processadores.

O último nível de paralelismo é a computação descentralizada, que se utiliza de computadores paralelos nos quais cada *CPU* tem sua própria memória privada, que não pode ser acessada diretamente por qualquer outra *CPU*. Essa arquitetura é chamada de *cluster* e é bastante utilizada nos supercomputadores para processamento de grandes conjuntos de dados, como em previsões meteorológicas, estudos econômicos e científicos.

1.3. Concepção da organização interna dos microprocessadores

Consideremos inicialmente o exemplo no qual desejamos processar a função:

$$fatorial(N) = N! = N * (N-1) * (N-2) * \ldots * 1$$

Um trecho de código em linguagem C que implementa uma solução para esse problema é mostrado a seguir, onde o resultado é fornecido pela variável `a`.

```
int a=1;
int b=N;
while(b!=0){
  a=a*b;
  b=b-1;
}
```

O trecho de código anterior pode ser descrito pela máquina de estados finitos apresentada na Figura 5. Uma máquina de estados finitos (*FSM - Finite State Machine*), ou autômato finito, é um modelo matemático usado para representar programas de computadores ou circuitos lógicos. O conceito é concebido como uma máquina abstrata que deve estar em um de um número finito de estados. A máquina de estar em apenas um estado de cada vez. Este estado é chamado de "estado atual". Um estado armazena informações sobre o passado, isto é, ele reflete as mudanças desde a entrada em um determinado estado, no início da operação do sistema, até o momento presente. Uma transição indica uma mudança de estado e é descrita por uma condição que precisa ser satisfeita para que a transição ocorra. Uma ação é a descrição de uma atividade que deve ser realizada num determinado momento.

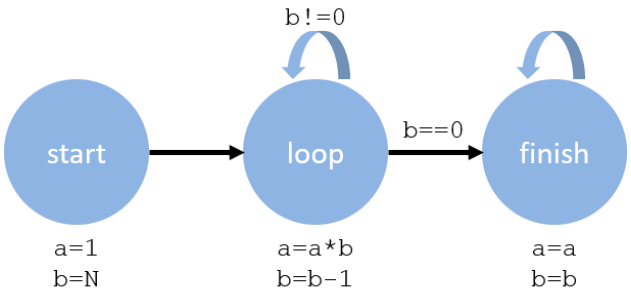

Figura 5 – Modelagem por máquina de estados finitos para o problema do processamento da função fatorial(N).

A máquina de estados da Figura 5 possui 3 estados: *start*, que é o estado inicial, onde as variáveis a e b são inicializadas; *loop*, onde as multiplicações e subtrações são executadas até que a condição na qual b é igual a zero seja encontrada; *finish*, onde o valor de a apresentará o resultado do processamento.

Podemos implementar esse autômato com o circuito lógico da Figura 6. O circuito é composto por um caminho onde circulam os dados (*Datapath*) e uma unidade que controla a transição entre os estados da máquina a cada pulso de clock recebido.

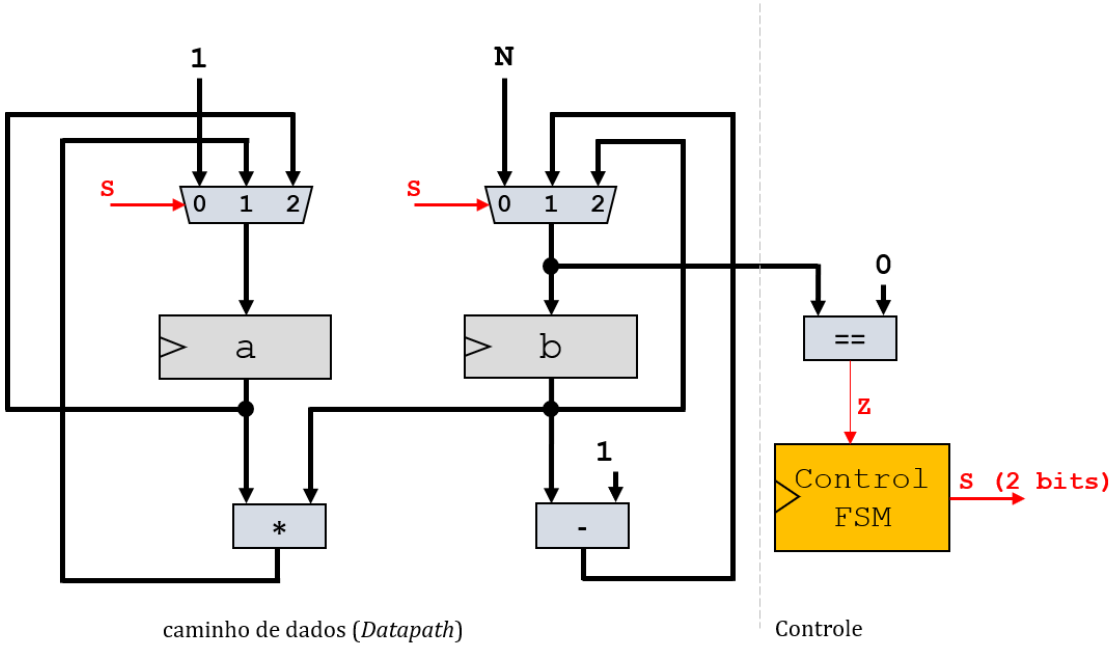

Figura 6 – Datapath e unidade de controle para a máquina de estados do problema do processamento da função fatorial(N).

O *datapath* é composto por 2 registradores, *a* e *b*, usados para armazenar temporariamente os dados durante o cálculo do fatorial, por duas unidades que realizam as multiplicações e as subtrações, dois multiplexadores que selecionam qual o próximo dado que será armazenado nos registradores *a* e *b* a partir dos sinais de seleção S (2 bits), além dos barramentos que interligam esses elementos.

Na seção de controle, há uma unidade de comparação que seta o bit de status Z (Z=1) se o próximo valor a ser armazenado no registrador *b* for zero e mantém esse bit resetado (Z=0) para outros casos. Esse bit é enviado a um circuito de controle (*Control FSM*) que gera os bits de seleção *S* dos multiplexadores do *datapath*.

O circuito de controle pode ser projetado tendo como entradas o bit Z e dois bits internos que especificam o estado atual (E_a), uma vez que a máquina possui um total de 3 estados. As saídas desse circuito serão os bits internos que indicam o próximo estado (E_f) além dos bits S de seleção dos multiplexadores, que indicam ao *datapath* o que deve ser feito para permitir o cálculo do fatorial. Como os multiplexadores possuem 3 entradas, são necessários 2 bits de seleção. A tabela-verdade mostrada a seguir é usada para projetar o circuito de controle.

Tabela-verdade para o circuito de controle do cálculo de *fatorial(N)*

Entradas		Saídas	
E_a	Z	E_f	S
00	0	01	00
00	1	10	00
01	0	01	01
01	1	10	01
10	0	10	10
10	1	10	10

Analisando a tabela, verificamos que o comportamento do circuito ocorre da seguinte forma:

1. No estado inicial (E_a=00 e Z=0 ou Z=1), os bits S=00 selecionam a entrada 0 dos multiplexadores, fazendo com que o registrador *a* venha a receber o valor 1 e o registrador *b* venha a receber o valor de N no próximo pulso de clock. Se o bit Z estiver resetado, isto é, se o valor a ser armazenado em *b* for diferente de 0, o estado futuro será o segundo estado (E_f=01). Caso contrário, se o bit Z estiver setado, o sistema fará uma transição para o último estado (E_f=10);
2. No segundo estado (Ea=01), os bits S=01 selecionam a entrada 1 dos multiplexadores, fazendo com que o registrador *a* venha a receber o resultado de *a*b* e o registrador *b* venha a receber o resultado de *b-1* no próximo pulso de clock. Se o bit Z estiver resetado, isto é, se o valor a ser armazenado em *b* for diferente de 0, o estado futuro será o estado atual (E_f=01). Caso contrário, se o bit Z estiver setado, o sistema fará uma transição para o último estado (E_f=10);
3. No terceiro estado (E_a=10), os bits S=10 selecionam a entrada 2 dos multiplexadores, fazendo com que os registradores *a* e *b* venham a receber seus próprios valores atuais, respectivamente, permanecendo assim indefinidamente e encerrando o cálculo do fatorial do número N.

Dessa forma, sabemos até agora, com tudo o que fizemos, como construir um hardware que executa especificamente o cálculo da função *fatorial(N)*. Devemos, portanto, buscar uma solução de hardware de uso geral para calcular qualquer tipo de função ou programa. É o que será feito nos próximos parágrafos.

Existem várias abordagens para construir um processador de uso geral. A mais empregada nos processadores modernos é aquela na qual o processador é constituído basicamente por duas grandes unidades funcionais: A unidade operacional, também chamada de seção de dados, e a unidade de controle, também chamada de seção de controle, como mostrado na Figura 7.

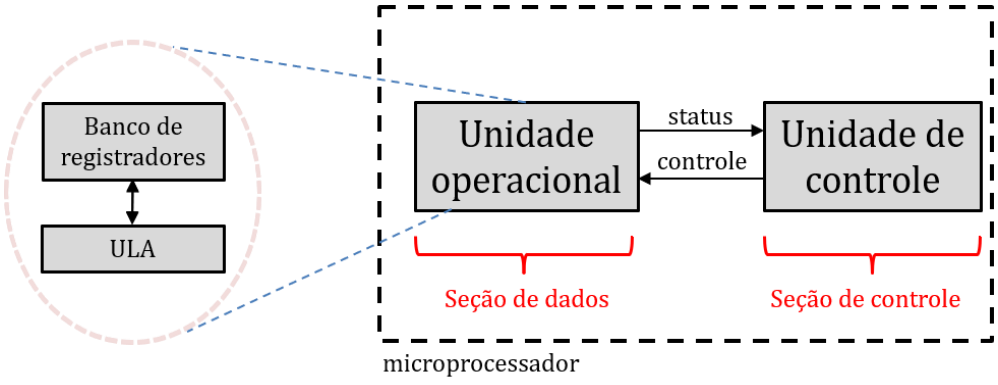

Figura 7 – Unidades funcionais de um microprocessador moderno de uso geral.

A unidade operacional é constituída pelos circuitos digitais que realizam o processamento dos dados e onde também são armazenados temporariamente os dados para serem processados. Essa unidade é, por sua vez, subdividida em duas outras: A Unidade Lógica e Aritmética (ULA), onde as instruções são efetivamente executadas, e um conjunto de registradores de uso geral, também chamado de banco de registradores, onde são armazenados os dados enquanto eles estão sendo manipulados.

No exemplo anterior, a unidade operacional é o *datapath,* que é capaz de executar multiplicações e subtrações e o banco de registradores possui apenas dois registros, *a* e *b*.

O *datapath* pode, agora, ser definido como a interconexão, realizada por meio de barramentos internos, da ULA com o banco de registradores e outros circuitos auxiliares (como os multiplexadores do exemplo da Figura 6) para permitir o fluxo interno dos dados e o processamento, que ocorre quando os dados dos registradores do banco são selecionados e direcionados à ULA e o resultado do processamento é redirecionado para um dos registradores do banco. Fazendo uma analogia com o corpo humano, o *datapath* funciona como os músculos, fazendo todo o trabalho duro e realizando os cálculos, e a unidade de controle seria o cérebro, informando ao *datapath* como realizar corretamente sua tarefa ativando corretamente os sinais de controle.

Além dos registradores de uso geral, as *CPUs* também podem conter outros registradores que têm funções especiais, como o registrador de instruções, registradores acessíveis apenas pela unidade de controle e/ou microprogramas de hardware e registradores de *pipeline*. Esses registradores não são diretamente acessíveis ao programador. Um registrador muito importante nesse contexto é o registrador de status do processador (*Processor Status Register - PSR*). Ele contém bits de status da operação do processador. Entre esses bits, temos o bit C (*carry*), ou bit de transporte, N (*negative*), V (*overflow*) e Z (zero), que refletem os estados de operação da ULA. Esses bits são usados para tomar decisões que determinam o fluxo de execução do programa, com base nos resultados da ULA ou no conteúdo dos registradores, como será visto no próximo capítulo. Os bits de status também são chamados de códigos de condição ou sinalizadores (*flags*).

O bit de transporte (geralmente indicado como C) é usado para indicar, quando setado, um transporte ou empréstimo aritmético gerado a partir do bit mais significativo da ULA. Ele permite que números com uma largura maior que a quantidade de bits dos registradores sejam adicionados/subtraídos transportando (adicionando) um dígito binário de uma adição/subtração parcial para o bit menos significativo de uma palavra mais significativa.

O bit sinalizador negativo (geralmente indicado como N), também chamado de bit de sinal, é usado para indicar se o resultado da última operação matemática produziu um valor no qual o bit mais significativo foi setado. Na representação de números com sinal em complemento de 2, esse bit, quando setado, indica que a última operação resultou em um valor negativo. Na verdade, o bit de sinal é o próprio bit mais significativo do barramento de saída da ULA.

O bit de overflow, ou transbordamento (geralmente indicado como V) é usado para indicar quando ocorreu um estouro aritmético em uma operação, indicando que o resultado representado em complemento de dois sinalizado não caberia na quantidade de bits usados para a operação.

O bit zero (geralmente indicado por Z) é usado para verificar se o resultado da última operação da ULA resultou em zero. Esse é o mesmo bit presente no circuito da Figura 6.

A unidade de controle fornece os sinais digitais que controlam o fluxo dos dados para a execução das microoperações realizadas no *datapath* e em outros componentes do sistema, como memórias externas, por exemplo. Além disso, como uma máquina de estados finita, ela também controla sua própria operação, determinando transições de estados dependendo dos resultados das microoperações atuais e passadas, o que pode ser indicado pelos bits de status, como o bit Z do exemplo anterior. Em um processador mais complexo, normalmente existem várias unidades de controle e vários caminhos de dados. Os sinais de controle são acionados dependendo das instruções que o processador deve executar, o que é expresso a partir de um programa armazenado em memória.

Para executar um programa, o microprocessador "lê" cada instrução da memória, "interpreta" essa instrução e depois a "executa". Esse modelo de execução de instrução é conhecido como "*Modelo de Execução de Instrução de Três Ciclos*". Os ciclos recebem os seguintes nomes: Ciclo de busca, Ciclo de decodificação e Ciclo de execução. Esses ciclos se repetem até que todas as instruções tenham sido executadas, conforme mostrado na Figura 8.

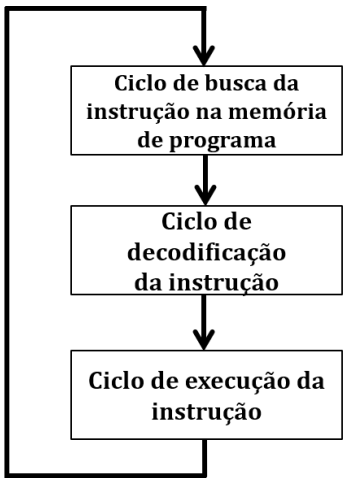

Figura 8 – Modelo de execução de instrução de 3 ciclos.

Um exemplo prático de implementação da estrutura mostrada na Figura 7 é apresentado na Figura 9.

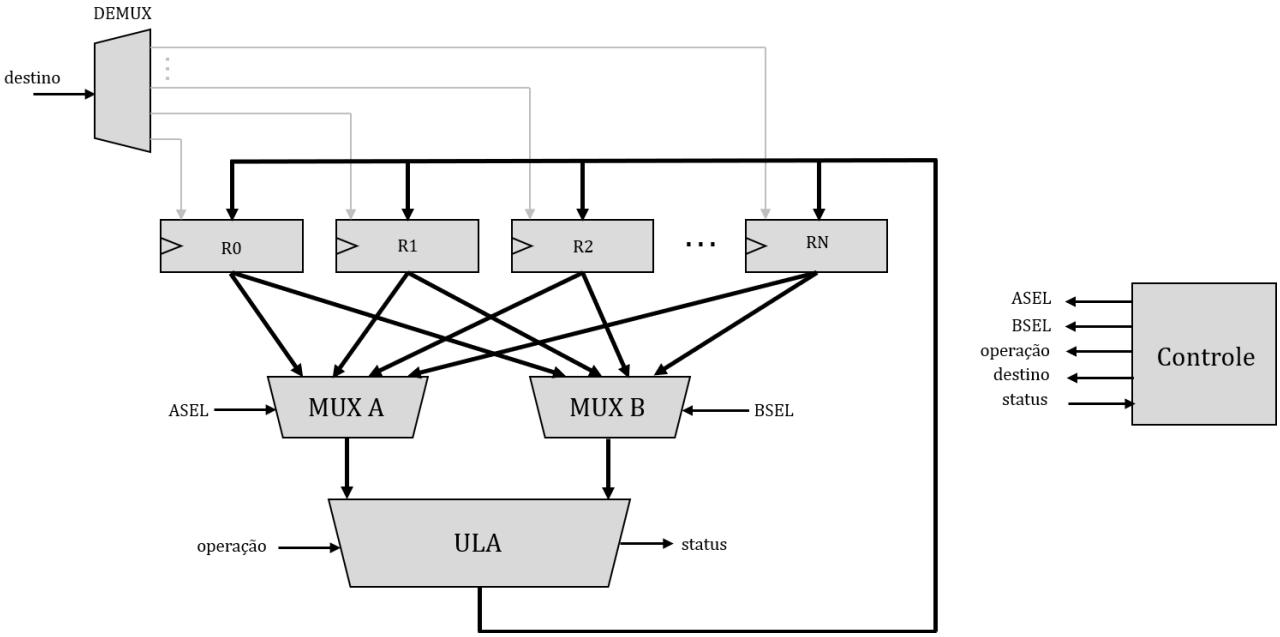

Figura 9 – Implementação prática da estrutura da figura 7.

Nessa figura, é possível identificar um conjunto com vários registradores (R0, R1, R2, ..., RN) cujas saídas alimentam, paralelamente, dois multiplexadores, MUX A e MUX B. Cada multiplexador seleciona a saída de um entre os vários registradores e a direciona para uma das entradas da unidade lógica e aritmética (ULA) onde são realizadas as diferentes operações. Cada entrada da ULA é considerada como um operando. O resultado da ULA retorna para as entradas dos registradores, mas é armazenado em apenas um deles, selecionado pelo demultiplexador de destino. Os sinais de controle, nesse caso, são os sinais que selecionam os operandos da ULA (ASEL e BSEL), que selecionam a operação a ser executada (operação), que seleciona o registrador de destino do resultado (destino), além dos bits de status da ULA (status). Os bits de status fornecem informações à unidade de controle permitindo, por exemplo, que uma instrução seja executada apenas se determinada condição for satisfeita.

Sabendo que a ULA é um circuito combinacional, toda a operação de transferência de dados dos registradores de origem, através da ULA, para o registrador de destino é realizada durante um único pulso de clock. As operações de deslocamento de bits geralmente são executadas em uma unidade separada, mas às vezes essas operações também são implementadas na ULA, e também ocorrem em um único pulso de clock.

Nesta arquitetura, todos os dados das operações executadas pela ULA são provenientes dos registradores. Outras fontes de dados, como a memória externa ao processador, só podem acessadas mediante o uso de duas instruções especiais: **LOAD**, para ler da memória e armazenar em um registrador e **STORE** para escrever o conteúdo de um registrador na memória. Obviamente, devem ser fornecidos barramentos auxiliares para os dados externos entrarem ou saírem do banco de registradores. Assim, toda instrução deverá indicar os registradores que serão utilizados na operação e o endereço de memória se a instrução for um LOAD ou um STORE. Outros exemplos de arquitetura de processadores incluem a baseada em acumulador, baseada em pilha e a baseada em memória-registrador.

A complexidade da unidade lógica e aritmética define a diversidade de operações básicas que o processador é capaz de executar. Essas operações formam o seu Conjunto de Instruções, ou Set de Instruções, que é a interface entre o software e o hardware do microprocessador.

Uma maior quantidade de registradores, por sua vez, permite o armazenamento temporário de mais dados no processador, não sendo necessário fazer acesso à dados externos, oriundos de memórias externas ou dispositivos de entrada e saída, que normalmente são mais lentos que o próprio processador, o que reduz o desempenho do sistema. Entretanto, isso exige uma maior complexidade no circuito de controle para permitir a correta seleção dos vários registradores e das várias operações.

Quando a ULA é simples, a quantidade de instruções é pequena e elas são projetadas para reduzir o seu tempo de execução. Elas consistem em instruções básicas que suportam vários formatos e tipos de dados, utilizam modos de endereçamento simples e são de comprimento fixo. Em geral, cada instrução requer apenas um pulso de clock para fornecer resultados em tempo de execução uniforme. Há mais linhas de código por programa, portanto, mais memória é necessária para armazenar as instruções. O compilador também precisa trabalhar mais

para converter instruções de linguagem de alto nível nas operações básicas da ULA. Tudo isso faz com que o processador seja classificado como uma máquina *RISC* (*Reduced Instruction Set Computer*), utilizado em dispositivos portáteis devido ao seu baixo consumo e alta eficiência energética, como por exemplo os processadores utilizados em sistemas embarcados.

Quando a ULA é complexa, o processador possui uma grande quantidade de instruções complexas projetadas para minimizar o número de instruções por programa, ignorando o número de pulsos de clock por instrução. A ênfase está na implementação de instruções complexas diretamente no hardware. O compilador tem pouco trabalho para converter uma linguagem de alto nível em código de máquina porque o tamanho do código é relativamente curto, portanto, é necessária pouca memória para armazenar as instruções. Possui uma grande variedade de modos de endereçamento, comprimento variável de formatos de instrução, podem ser necessários vários pulsos de clock para executar uma instrução, a lógica de decodificação da instrução é complexa. Tudo isso faz com que o processador seja classificado como uma máquina *CISC* (*Complex Instruction Set Computer*), como por exemplo os processadores comerciais utilizados em desktops e notebooks.

1.4. Sistemas baseados em microprocessadores

O microprocessador sozinho não serve a nenhum propósito útil, a menos que seja suportado pela memória externa e portas de entrada e saída. A combinação de memória e portas de I/O com o microprocessador é conhecida como um sistema baseado em microprocessador, ou sistema microprocessado.

Conforme discutido anteriormente, o microprocessador executa o programa armazenado na memória e transfere dados de e para o mundo externo através das portas de I/O. O microprocessador é interconectado à memória e às portas de I/O por meio de três barramentos: O barramento de dados, o barramento de endereços e o barramento de controle. Um barramento é basicamente um link de comunicação entre a unidade de processamento e os dispositivos periféricos, como mostrado na Figura 10.

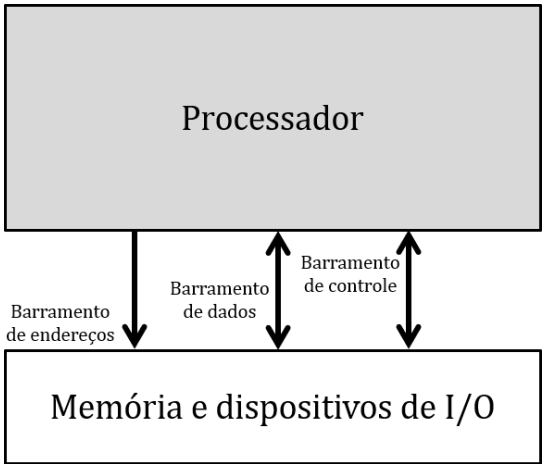

Figura 10 – Arquitetura de um sistema microprocessado.

O barramento de endereços é unidirecional e deve ser usado pela CPU para enviar o endereço do local da memória a ser acessado. Também é usado pela CPU para selecionar uma porta de entrada ou saída específica. Pode consistir em 8, 12, 16, 20 ou até mais linhas paralelas (bits). O número de bits no barramento de endereços determina a quantidade de endereços que podem ser acessados na memória. Um barramento de endereço de 16 bits, por exemplo, pode acessar 2^{16} (65536 – 64K) endereços. As linhas são rotuladas como $A_0\ A_1 \ldots A_{n-1}$, onde n é a largura do barramento, em bits.

O barramento de dados é bidirecional, ou seja, o fluxo de dados pode ocorrer da CPU para os periféricos, ou memória, e vice-versa. As linhas do barramento são rotuladas como: $D_0\ D_1 \ldots D_{n-1}$, onde n é a largura do barramento, em bits.

Um microprocessador é caracterizado pela largura do seu barramento de dados, conhecido como comprimento da palavra de dados, fornecido em n bits, onde n pode ser 4, 8, 16, 32 ou 64. Um microprocessador de 8 bits, por exemplo, pode processar dados de 8 bits por vez. Se os dados consistirem em mais de 8 bits, o processador coletará os 8 primeiros bits e fará seu processamento. Depois, os próximos grupos de 8 bits de dados são coletados, um a um, para processamento. Sua ULA é projetada para processar dados de 8 bits. Seus registradores de uso geral são de 8 bits. Da mesma forma, um microprocessador de 64 bits processa dados de 64 bits por vez, sua ULA processa dados de 64 bits e os registros de uso geral retêm dados de 64 bits.

Em linhas gerais, um microprocessador com palavra de dados de *n* bits pode ser usado para executar operações em dados maiores, mas, essas operações geralmente serão mais lentas que as operações de *n* bits. Por exemplo, o Intel 8085 (8 bits) pode ser usado para executar a adição de dois números de 32 bits. No entanto, isso deve ser feito sob o controle de um programa. Assim, em vez de executar a adição com apenas uma instrução, o 8085 pode ser programado para adicionar dois números de 32 bits, executando uma sequência de quatro adições de pares de números de 8 bits. Por outro lado, o Intel 80386 (32 bits) ou o Motorola 68030, pode adicionar dois números de 32 bits em apenas uma operação de adição. Portanto, os microprocessadores de 32 bits são muito mais rápidos que os de 16 ou 8 bits. Um processador com tamanho de palavra maior é mais poderoso e pode processar dados em velocidade mais rápida em comparação com um processador com tamanho de palavra menor.

O barramento bidirecional de controle contém as várias linhas individuais carregando sinais de sincronização e status. Esse barramento envia e recebe sinais de controle e status de e para a memória, portas de I/O e outros dispositivos periféricos para garantir a sua correta operação. Por exemplo, se for desejado ler o conteúdo de um determinado local de memória, a CPU envia primeiro o endereço desse local no barramento de endereços e um sinal de controle '*Memory Read*' no barramento de controle. A memória responde produzindo dados armazenados no local de memória endereçado no barramento de dados para que o processador possa fazer a leitura.

Relativamente à conexão do processador com a memória e com os dispositivos de I/O, podemos dizer que há dois tipos de arquitetura computacional: Arquitetura de von Neumann e arquitetura de Harvard, como mostrado na Figura 11.

Na arquitetura de Von Neumann, as instruções e os dados estão na mesma memória física. Há apenas um barramento de endereços e apenas um barramento de dados. Quando o conteúdo endereçado na memória é um dado, ele é direcionado para o banco de registradores do *datapath*. De outra forma, quando o conteúdo endereçado é uma instrução, ele é direcionado para a unidade de controle para ser decodificado e posteriormente executado. Apenas um acesso (dado ou instrução) pode ser realizado por vez. É uma arquitetura relativamente barata e simples. Essa arquitetura é a adotada em processadores *CISC*.

Na arquitetura de Harvard, a memória de instruções, normalmente uma ROM não volátil, e a memória de dados, normalmente uma RAM, são dois dispositivos de memória separados. Há dois conjuntos de barramentos de dados e endereços, um para cada dispositivo, o que permite o acesso simultâneo a ambas as memórias. Devido a isso, oferece um processamento mais rápido, considerando a mesma velocidade de clock da arquitetura von Neumann. Sob as mesmas exigências de performance, tende a ser mais eficiente no consumo de energia, visto que é necessária uma velocidade de clock menor. Essa arquitetura é a adotada em processadores *RISC*.

Nas aplicações da arquitetura de Harvard, as memórias de dados e instruções são suficientemente pequenas para caberem no mesmo espaço de endereçamento. Se não houver sobreposição entre os endereços dos dados e das instruções, as memórias podem compartilhar o mesmo barramento de endereços.

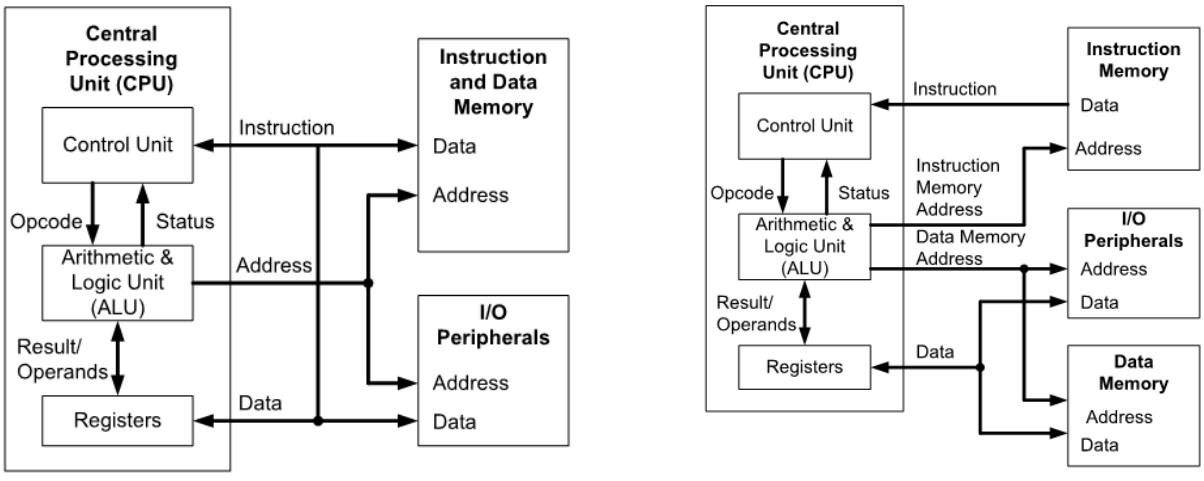

Figura 11 – (a) Arquitetura de von Neumann. (b) Arquitetura de Harvard. ■