10. CPU (CENTRAL PROCESSOR UNIT)

Como vimos no capítulo 8, um mesmo módulo funcional podia realizar uma multiplicação ou uma divisão função de um bit de entrada no módulo de controlo. Este facto, conduz-nos à ideia de conceber uma estrutura capaz de executar qualquer algoritmo. A estrutura deverá ser constituída por um módulo funcional contendo uma ALU, caminhos de dados para obtenção de operandos, e armazenamento de resultados numa memória de dados, um módulo de controlo em que o comportamento lhe seja determinado por instruções armazenadas numa memória de programa. O módulo de controlo terá duas fases, uma de preparação em que lê da memória de programa uma instrução e outra de execução, que realiza acções determinadas pelos bits contidos nessa instrução. O controlo repetirá estas duas fases de forma continuada até concluir o algoritmo.
Assim sendo, esta estrutura será constituída pelos seguintes elementos:

- Memória de programa onde se armazenam as várias instruções a serem executadas;
- Memória de dados onde se armazenem operandos e resultados das operações;
- Módulo funcional contendo uma ALU capaz de realizar operações aritméticas, relacionais e lógicas;
- Módulo de controlo contendo um registo denominado PC que, lhe indique na fase de preparação, qual o endereço de memória de onde vai ser lida a instrução a ser executada e que desta forma estabelece o estado de execução do programa.

É esta estrutura formada pelo módulo funcional e pelo módulo de controlo, que denominamos por processador ou CPU (*Central Processor Unit*) e cujo possível diagrama de blocos está representado na Figura 10-1.

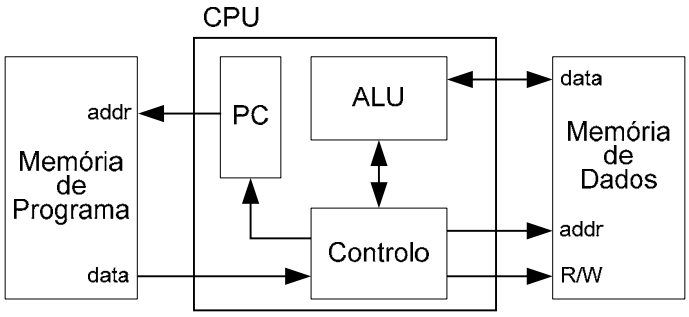

Figura 10-1 – CPU, diagrama de blocos

Como se pode observar na Figura 10-1, o modelo de controlo é muito idêntico ao do controlo micro programado.

10.1 Conjunto das instruções

Desde o início do desenvolvimento dos processadores que existem várias linhas de concepção no sentido de procurar a melhor relação entre complexidade e eficiência. Entre estas linhas de desenvolvimento, destacam-se as arquitecturas CISC (*Complex Instruction Set Computer*) seguidas em grande parte pela INTEL, e as arquitecturas RISC (*Reduced Instruction Set Computer*) seguidas pela ARM e MIPS. O grande desenvolvimento que as arquitecturas RISC tiveram nos últimos anos deve-se essencialmente ao facto do conjunto das instruções ser uniforme, regular e ortogonal.

Como consequência, estas arquitecturas podem ser mais facilmente implementadas em estruturas *pipeline*, melhorando desta forma o seu desempenho.
A especificação de um processador passa por definir alguns elementos constituintes, um conjunto de instruções a que este deve obedecer, a dimensão e o formato destas. Após definir estes elementos estamos em condições de desenhar uma arquitectura que melhor dê resposta à especificação estabelecida. O desenho do processador, como se constatará adiante, será de certa forma iterativa, ou seja, serão feitos reajustes ao longo do processo caso a arquitectura pontualmente se complique demasiado para responder à especificação de uma instrução não essencial.
A arquitectura do conjunto das instruções, também denominado ISA (*Instruction Set Architecture*), constitui uma parte fundamental na definição da arquitectura do processador. Por esta razão deveremos começar por definir que tipo de instruções é que o CPU deverá executar, tal que através delas, se possa sintetizar com eficiência uma determinada gama de algoritmos, isto porque o ISA de um processador de uso genérico será com certeza diferente do ISA de um processador de imagem ou de um processador de controlo.
Um ISA tem três componentes: a especificação da instrução; o formato da instrução; e a fonte de armazenamento dos operandos. A especificação está ligada a gama de algoritmos, o formato à complexidade da arquitectura e a fonte de armazenamento ao tipo de memória.
No caso de um processador genérico, podem-se definir três tipos de instrução:

Transferência	lê da memória de dados ou da memória de programa os valores que constituem os operandos da ALU e escrevem na memória de dados os resultados produzidos pelas operações efectuadas pela ALU;
Processamento	estabelece qual a operação a ser realizada pela ALU;
Controlo de Fluxo	por avaliação do resultado produzidos pelos operadores relacionais altera o estado de execução do programa.

Concretizado o processador, ou seja, a arquitectura e o ISA, o programador exprime o algoritmo que pretende realizar através de uma lista de instruções ao qual dá o nome de programa, e no qual especifica as operações a realizar, os operandos e a sequência com que as instruções serão processadas.
No sentido de diminuir a dimensão da memória de programa, as várias instruções podem ser codificadas (comprimidas) numa série de bits, o qual passaremos a denominar por código máquina. Esta série de bits organizados em palavras, será posteriormente armazenado na memória de programa a que por esta razão é usualmente referida como memória de código.
Dado que a execução de um algoritmo é concretizado pela execução de uma sequência de instruções, então por simplicidade colocaremos a sequência de instruções que compõem o programa em endereços consecutivos da memória de programa.
Uma vez que o código do programa está armazenado de forma sequencial, e como já foi anteriormente referido, o módulo de controlo do processador dispõe de um registo usualmente denominado por PC (***P**rogram **C**ounter*) ou IP (***I**nstruction **P**ointer*) que referencia a instrução que vais ser executada, então este registo, por cada instrução executada é incrementado de uma unidade de instrução. No entanto a instrução a ser executada, pode ser do tipo controlo de fluxo, ou

seja, que tem como objectivo a alteração do valor do registo PC. Esta acção, referida normalmente por salto (*jump* ou *branch*) permite a execução condicionada de um troço de programa.

10.2 Estrutura interna

A arquitectura de um processador como já foi referido anteriormente consiste numa estrutura contendo registos, operadores, caminhos para os dados e um sistema de controlo que determina a sequência de acções a realizar sobre registos e operadores que compõem a estrutura.

O elemento principal do CPU é uma ALU, capaz de executar operações aritméticas: soma, subtracção, etc., bem como operações lógicas básicas. Como a ALU executa operações unárias e binárias, são necessários dois registos para conter os dois operandos, e um registo onde é armazenado o resultado. Além dos operadores aritméticos e lógicos, a ALU implementa operadores relacionais booleanos, tais como igualdade, maior que, menor que, informação de que as operações aritméticas produziram excesso etc. São o resultado destes últimos operadores, normalmente denominados por *flags*, que fornecem informação às instruções de controlo de fluxo no sentido de realizar execução condicionada. As *flags* produzidas durante uma operação lógica ou aritmética são armazenadas num registo denominado PSW (***P**rogram **S**tatus Word*), permitindo a avaliação das *flags* nas instruções subsequentes. As *flags* mais comuns são: a **Z** (Zero) que, quando activa, indica que o resultado da operação lógica ou aritmética foi igual a zero; a de **CY** (*carry*) com dupla funcionalidade que, quando activo, indica que a adição produziu arrasto (*carry*) e a subtracção deficit (*borrow*); a **GE** (***G**reater or **E**qual*) que, indica que a operação aritmética produziu um resultado positivo, interpretados os operandos, como inteiros com sinal em código dos complementos para dois.

10.3 Formato das instruções

Quanto ao formato das instruções estas podem ser constituídas unicamente pelo código da instrução ou pelo código da instrução e um ou mais parâmetros. É óbvio que quanto maior for o número de bits que dispusermos para codificar a instrução mais estruturada será a codificação e por conseguinte mais simples será o descodificador das micro-instruções.

O formato mais natural seria constituído por três parâmetros: as referências de memória dos dois operandos e a referência de memória onde colocar o resultado. Este formato tem como vantagem um menor número de instruções para executar um algoritmo, mas tem como desvantagem a dimensão de cada instrução. O aumento da dimensão de cada instrução implicará maior dimensão do bus de dados que liga o CPU à memória ou o aumento do tempo de preparação da instrução por acessos sucessivos à memória caso esta fosse seccionada e repartida por várias posições da memória.

Outro formato mais económico seria instruções com um só operando. Este formato tem como vantagem a diminuição da largura do bus de dados mas tem como desvantagem a menor eficiência em termos do número de instruções necessárias para executar um algoritmo. Para cumprir o mesmo objectivo de uma instrução com três parâmetros seriam necessárias instruções para transportar individualmente cada um dos operandos e o resultado.

Este equilíbrio não é fácil de obter pois são muitas as variáveis em jogo, quando se pensa na implementação e na eficiência de execução e na dimensão do programa.

Outro item que é necessário definir é a referência aos operandos, ou seja o tipo de parâmetro que cada instrução manipula. Este item é normalmente denominado por tipo de endereçamento. São conhecidos vários tipos de endereçamento sendo os mais comuns: imediato, directo, registo, indirecto e o indexado.

Imediato o parâmetro é o próprio valor.

Directo o parâmetro constitui o endereço de memória onde reside o operando.

Registo o parâmetro é a referência ao registos que contém o operando

Indirecto o parâmetro é a referência a um registo cujo conteúdo serve de endereço da memória onde reside o operando.

Indexado constituído por dois parâmetros: a referência a um registo base e um índice. O conteúdo do registo referenciado servirá como endereço base que, somado com o índice, define o endereço de memória onde se encontra o operando. O índice poderá ser uma constante ou o conteúdo de um registo.