Aula 9: A Linguagem de Máquina: Operandos em Memória

Ao contrário dos programas nas linguagens de alto nível, os operandos das instruções aritméticas são restritos, precisam ser de um grupo limitado de locais especiais, embutidos diretamente no hardware, chamados *registradores*. Os registradores são os tijolos da construção do computador: os registradores são primitivas usadas no projeto de hardware que também são visíveis ao programador quando o computador é completado.

Uma diferença importante entre as variáveis de uma linguagem de programação e os registradores é o número limitado de registradores, normalmente 32 nos computadores atuais. O MIPS possui 32 registradores, detalhados na Tabela 1.

Nome	Número	Utilização
zero	0	Constante 0
at	1	Reservado para o Assemblador (“Assembler temporary”)
v0 v1	2 3	Usado para retornar valores de funções
a0 a1 a2 a3	4 5 6 7	Usado para transferir os primeiros 4 argumentos para funções (os restantes usam a “stack”)
t0 t1 t2 t3 t4 t5 t6 t7	8 9 10 11 12 13 14 15	Temporários (não é necessário que sejam preservados pelas funções)
s0 s1 s2 s3 s4 s5 s6 s7	16 17 18 19 20 21 22 23	“Saved temporary” (as funções devem salvaguardar e restaurar os valores nestes registos)
t8 t9	24 25	Temporários (não é necessário que sejam preservados pelas funções)
k0 k1	26 27	Reservados para o sistema operativo
gp	28	Ponteiro para a área global
sp	29	Ponteiro para “Stack” (“Stack Pointer”)
fp	30	Ponteiro para “frame”
ra	31	Endereço de retorno (“Return Adress”)

Tabela 1: Registradores de uso geral.

O motivo para o limite dos 32 registradores pode ser encontrado no segundo dos quatro princípios de projeto básicos da tecnologia de hardware:

Princípio de Projeto 2: menor significa mais rápido.

Uma quantidade muito grande de registradores pode aumentar o tempo de ciclo do clock simplesmente porque os sinais eletrônicos levam mais tempo quando precisam atravessar uma distância maior.

Compilando uma atribuição de operandos em memória

Vamos supor que A seja uma sequência de 100 words e que o compilador tenha associado o endereço inicial da sequência de A, ou o seu endereço base em $s3. Compile esta instrução de atribuição em Python:

g = h + A[3]

Embora haja uma única operação nesse instrução de atribuição, um dos operandos está na memória, de modo que primeiro precisamos transferir A[3] para um registrador. O endereço desse elemento da sequência é a soma da base da sequência A, encontrada no registrado $s3, com um número para selecionar o elemento 4. Os dados devem ser colocados em um registrador temporário, para uso na próxima instrução. Com base na Figura 1, a primeira instrução compilada é

lw $t0, 12($s3) # Registrador temporário $t0 recebe A[3]

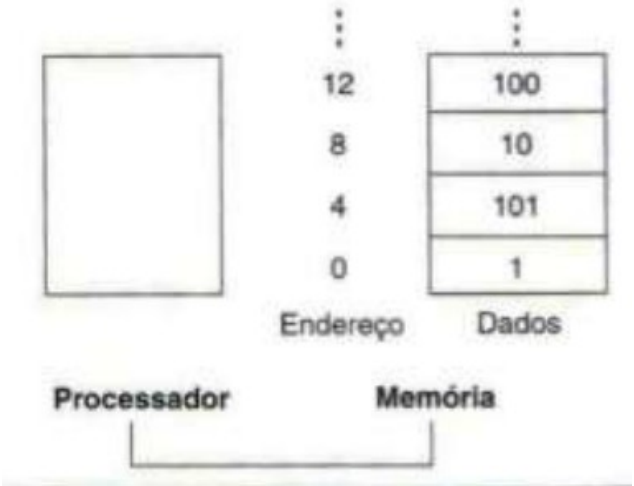

Figura 1: Endereços de memória reais do MIPS e conteúdos da memória para essas words.

A seguinte instrução pode operar sobre o valor em $t0 (que é igual a A[3]), já que está em um registrador. A instrução precisa somar h (contido em $s2) com A[3] ($t0) e colocar a soma no registrador correspondente a g (associado a $s1). No MIPS, words precisam começar em endereços que sejam múltiplos de 4.

Add $s1, $s2, $t0 # g = h + A[3]

Exercícios

1) Traduza para o assembly do MIPS os códigos Python a seguir e calcule a quantidade de variáveis e quantidade de instruções necessárias em MIPS:

   a) a = (c + (g - h)) - T[2])

   b) p = h - b

      k = (G[1] + (h - (p - b)))

   c) g = (a + b) - d

      d = (g - H[1])

      e = (g - d) – w)

2) Faça o processo inverso agora:

   a) lw $t0, 4($s3)

      add $t0, $s2, $t0