PCS2056 – Linguagens e Compiladores

4ª parte do projeto – definição do ambiente de execução

Renan Martins Zomignani Mendes
Tiago Schelp Lopes

Introdução

O ambiente de execução provê suporte para executar o programa gerado pelo compilador. Esse ambiente de execução implementa abstrações incorporadas na definição da linguagem e que não são implementadas pelo código que está sendo executado. Gerenciam a comunicação com o sistema operacional, dispositivos de entrada e saída e outros programas, alocação de memoria e variáveis declaradas, passagem de parâmetro entre chamadas de função e funções especiais definidas.

Alguns exemplos de rotinas típicas de ambiente de execução incluem #NEWLINE, #CONVERT, #READ e #STOP.

#NEWLINE: Imprime uma linha no dispositivo de entrada/saída de acordo com os comandos definidos pela linguagem.

#CONVERT: Converte o conteúdo do acumulador para adequá-lo para a saída.

#READ: Faz leitura de dados, convertendo o valor lido para um valor adequado ao acumulador.

#STOP: Para o processamento do programa sendo executado e retorna o controle ao sistema operacional.

No nosso caso, o ambiente de execução do código é a maquina de von Neumann. O código será executado pela máquina virtual utilizada na disciplina PCS2024.

A máquina de von Neumann disponibiliza ações mais poderosas e ágeis que a máquina de Turing. Por se tratar de código armazenado em memória, existe considerável avanço em flexibilidade e poder computacional se comparado com máquinas de programa fixo, inclusive em tempo de execução, pois na máquina de von Neumann o programa e os dados são armazenados no mesmo espaço de memória. Essa memoria pode ser endereçável aleatoriamente.

Outras peculiaridades da Máquina de von Neumann serão exploradas adiante.

Instruções da linguagem de saída

Operação	Nome	Mnemônico
0	Jump	JP
1	Jump if Zero	JZ
2	Jump if Negative	JN
3	Load Value	LV
4	Add	+
5	Subtract	-
6	Multiply	*
7	Divide	/
8	Load	LD
9	Move to Memory	MM
A	Subroutine Call	SC
B	Return from Subroutine	RS
C	Halt Machine	HM
D	Get Data	GD
E	Put Data	PD
F	Operating System	OS

Pseudo-instruções da linguagem de saída

Símbolo	Pseudo-instrução	Exemplo
@	Origem absoluta do programa	`@ /50; indica /050 como origem do código`
#	Fim do programa	`# X;`
K	Área preenchida por uma constante de 2 bytes	`XYZ K /10; gera /10 na posição XYZ`
$	Bloco de memória com número especificado de bytes	`XYZ $ =30; reserva 30 bytes e o primeiro chama-se XYZ`
&	Origem relocável	`& /50; próximo código se localizará no endereço /050 relativo à origem do código`

Características gerais

O ambiente de execução do código objeto possui 4096 posições em memória e 7 registradores específicos.

Registrador Função
MAR Registrador de endereço de memória
MBR Registrador de dados da memória
IC Registrador de endereço de instrução
IR Registrador de instrução
OP Registrador de código de operação
OI Registrador de operando de instrução
AC Acumulador

Deve existir também uma posição em memória para guardar o endereço de retorno de chamada de função. Para que seja possível o encadeamento de subrotinas o valor desse endereço também é guardado na pilha da máquina virtual.

Os parâmetros tanto de chamada de função quanto de retorno ficam armazenados no AC. No caso de mais de um parâmetro pra chamada de função, esses valores ficam armazenados na pilha da máquina virtual.

A o espaço de memória é dividido em 3 partes:

- **Área de programa:** Armazena o código do programa.
- **Área de dados:** Área onde ficam armazenados os valores das variáveis declaradas no programa.
- **Área de pilha:** Área onde serão armazenados valores temporários auxiliares à execução do programa como no caso de chamadas de subrotinas.