INSTITUTO TECNOLÓGICO DE NUEVO LEÓN

Unidad #4 Generación de código objeto

Nombre: Eduardo Rodriguez Rodriguez

13480062

Lenguajes y autómatas 2

3 de mayo de 2018

Introducción

La fase final de un compilador es la generación de código objeto, que por lo general consiste en código de máquina relocalizable o código ensamblador. Las posiciones de memoria se seleccionan para cada una de las variables usadas por el programa. Después, cada una de las instrucciones intermedias se traduce a una secuencia de instrucciones de máquina que ejecuta la misma tarea. Un aspecto decisivo es la asignación de variables a registros.

El generador de código objeto puede considerarse como la penúltima fase de un compilador, la cual se encarga de tomar como entrada el código intermedio generado por el front-end, y producir código objeto de la arquitectura target para luego entrar en la fase de optimización de código.

Capítulo: 1 Registros

Los registros son la memoria principal de la computadora. Existen diversos registros de propósito general y otros de uso exclusivo. Algunos registros de propósito general son utilizados para cierto tipo de funciones. Existen registros acumuladores, puntero de instrucción, de pila, etc.

Los registros son espacios físicos dentro del microprocesador con capacidad de 4 bits hasta 64 bits dependiendo del microprocesador que se emplee.

La UCP o CPU tiene 14 registros internos, cada uno de ellos de 16 bits (una palabra). Los bits están enumerados de derecha a izquierda, de tal modo que el bit menos significativo es el bit 0.

Los registros se pueden clasificar de la siguiente forma:

Registros de datos:

AX: Registro acumulador. Es el principal empleado en las operaciones aritméticas.

BX: Registro base. Se usa para indicar un desplazamiento.

CX: Registro contador. Se usa como contador en los bucles.

DX: Registro de datos.

Registros de segmentos:

CS: Registro de segmento de código. Contiene la dirección de las instrucciones del programa.

DS: Registro segmento de datos. Contiene la dirección del área de memoria donde se encuentran los datos del programa.

SS: Registro segmento de pila. Contiene la dirección del segmento de pila. La pila es un espacio de memoria temporal que se usa para almacenar valores de 16 bits (palabras).

ES: Registro segmento extra. Contiene la dirección del segmento extra. Se trata de un segmento de datos adicional que se utiliza para superar la limitación de los 64Kb del segmento de datos y para hacer transferencias de datos entre segmentos.

Registros punteros de pila:

SP: Puntero de la pila. Contiene la dirección relativa al segmento de la pila.

BP: Puntero base. Se utiliza para fijar el puntero de pila y así poder acceder a los elementos de la pila.

Capitulo: 2 Lenguaje Ensamblador

El lenguaje Assembly es un tipo de lenguaje de bajo nivel utilizado para escribir programas informáticos, y constituye la representación más directa del código máquina específico para cada arquitectura de computadora.

Este lenguaje fue usado ampliamente en el pasado para el desarrollo de software, pero actualmente sólo se utiliza encontradas ocasiones, especialmente cuando se requiere la manipulación directa del hardware o se pretenden rendimientos inusuales de los equipos.

Se clasifican en

Ensambladores básicos: Son de muy bajo nivel, y su tarea consiste básicamente, en ofrecer nombres simbólicos a las distintas instrucciones, parámetros y cosas tales como los modos de direccionamiento

Ensambladores modulares, o macro ensambladores: Descendientes de los ensambladores básicos, fueron muy populares en las décadas de los 50 y los 60, fueron antes de la generalización de los lenguajes de alto nivel. Un macroinstrucción es el equivalente a una función en un lenguaje de alto nivel.

Las operaciones básicas en un lenguaje ensamblador son la suma la resta la multiplicación y la división y Necesitara un poco más de información sobre la arquitectura y SO para el cual programas.

Pero la idea básica es:

Definir qué parámetros tendrá la función.

Hacer el programa, propiamente dicho, en assembler.

```
-u 100 1a
0CFD:0100 BA0B01      MOV     DX,010B
0CFD:0103 B409        MOV     AH,09
0CFD:0105 CD21        INT     21
0CFD:0107 B400        MOV     AH,00
0CFD:0109 CD21        INT     21
-d 10b 13f
0CFD:0100                                       48 6F 6C 61 2C
0CFD:0110  20 65 73 74 65 20 65 73-20 75 6E 20 70 72 6F 67
0CFD:0120  72 61 6D 61 20 68 65 63-68 6F 20 65 6E 20 61 73
0CFD:0130  73 65 6D 62 6C 65 72 20-70 61 72 61 20 6C 61 20
0CFD:0140  57 69 6B 69 70 65 64 69-61 24
```

Capitulo: 3 Lenguaje Maquina

Es el sistema de códigos directamente interpretable por un circuito microprogramable, como el microprocesador de una computadora o el microcontrolador de un autómata. Este lenguaje está compuesto por un conjunto de instrucciones que determinan acciones a ser tomadas por la máquina.

Es el que proporciona poca o ninguna abstracción del microprocesador de un ordenador. El lenguaje máquina solo es entendible por las computadoras. Se basa en una lógica binaria de 0 y 1, generalmente implementada por mecanismos eléctricos. En general el lenguaje maquina es difícil de entender para los humanos por este motivo hacemos uso de lenguajes más parecidos a los lenguajes naturales.

Se denomina lenguaje máquina a la serie de datos que la parte física de la computadora o hardware, es capaz de interpretar. El lenguaje máquina fue el primero que empleo el hombre para la programación de las primeras computadoras.

Una instrucción en lenguaje máquina puede representarse de la siguiente forma: 011011001010010011110110. Esta secuencia es fácilmente ejecutada por la computadora, pero es de difícil interpretación, siendo aún más difícil la interpretación de un programa escrito de esta forma.

Ventajas

Mayor adaptación al equipo.

Máxima velocidad con mínimo uso de memoria.

Desventajas

Imposibilidad de escribir código independiente de la máquina.

Mayor dificultad en la programación y en la comprensión de los programas.

El programador debe conocer más de un centenar de instrucciones.

Es necesario conocer en detalle la arquitectura de la máquina.

Capitulo: 4 Administrador de Memoria

La administración de la memoria es un proceso hoy en día muy importante, de tal modo que su mal o buen uso tiene una acción directa sobre el desempeño de memoria. En general un ensamblador tiene un administrador de memoria más limitado que un compilador; en la mayoría de los lenguajes de programación el uso de punteros no estaba vigilado por lo que se tienen muchos problemas con el uso de memoria.

La memoria principal puede ser considerada como un arreglo lineal de localidades de almacenamiento de un byte de tamaño. Cada localidad de almacenamiento tiene asignada una dirección que la identifica

Se distinguen los siguientes propósitos del sistema de administración de memoria:

Protección.

Si varios programas comparten la memoria principal, se debería asegurar que el programa no sea capaz de cambiar las ubicaciones no pertenecientica él. Aunque una acción de escritura puede tener efectos más graves que una de lectura, esta última tampoco debería estar permitida, para proporcionar algo de privacidad al programa.

Características

Los sistemas de administración de memoria de sistemas operativos multitarea normalmente tratan con las siguientes tareas.

Reasignación

En los sistemas con memoria virtual, los programas durante su ejecución pueden salir por un tiempo de la memoria y luego regresar, de modo que no pueden colocarse en el lugar que ocupaban previamente. Por ello debe ser posible que residan en diferentes partes de la memoria en diferentes momentos

Protección

Los procesos no deberían poder referenciar la memoria de otros procesos sin permiso, para evitarlo existe la protección de memoria, que evita que código malicioso o erróneo de un programa interfiera con la operación de otros programas en ejecución.

Memoria compartida

Aunque la memoria utilizada por diferentes procesos suele estar protegida, algunos procesos puede que sí tengan que compartir información y, para ello, han de acceder la misma sección de memoria.

Organización lógica

Los programas a menudo están organizados en módulos, algunos de los cuales pueden ser compartidos por diferentes programas, algunos son de solo-lectura y otros contienen datos que se pueden modificar. Se escriben y se compilan independientemente.

Conclusiones

Los temas de esta unidad 4 nos hablan de la programación de bajo nivel que es el lenguaje ensamblador y el lenguaje máquina, que hoy en día no son muy usados pero es una base para cualquier programador

Bibliografías

http://acaurio.blogspot.mx/2016/11/unidad-4-generacion-de-codigo-objeto.html

https://es.wikipedia.org/wiki/Lenguaje_ensamblador

https://lignux.com/lenguaje-maquina/