Resumen

Este documento tiene un resumen de la unidad 4.

rEPorte unidad 4

Jesus Oviedo Ovalle #13480497

**Generación de código objeto.**

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.

**Distribución**

La distribución es el proceso en el que el programa generado puede ejecutarse en otras máquinas. Con respecto al ensamblador, la mayoría del direccionamiento se hace relativo para que el programa sea re localizable por un programa llamado cargador.

El uso de registros puede dividirse en dos sub-problemas:

* Durante la reserva de registros (allocation), se seleccionan el conjunto de variables que viviría en registros en un punto del programa.
* Durante la (posterior) asignación de registros (assignation), se elige el registro específico para cada variable.

**Asignación**

El ensamblador se encarga de ajustar el tamaño de los segmentos tomando como base el número de bytes que necesita cada instrucción que va ensamblando, ya que sería un desperdicio de memoria utilizar los segmentos completos. Por ejemplos, si un programa únicamente necesita 10kb para almacenar los datos, el segmento de datos únicamente será de 10kb y no de los 64kb que puede manejar.

**Lenguaje ensamblador**

El ensamblador es un traductor de un código de bajo nivel a un código, ejecutable directamente por la máquina para la que se ha generado.  
Fue la primera abstracción de un lenguaje de programación, posteriormente aparecieron los compiladores.

**Características**

El programa lee un archivo escrito en lenguaje ensamblador y sustituye cada uno de los códigos mnemotécnicos por su equivalente código máquina.  
Los programas se hacen fácilmente portables de máquina a máquina y el cálculo de bifurcaciones se hace de manera fácil.

**Clasificación**

* **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.

**Almacenamiento**

Una de las principales ventajas del uso del ensamblador, es que se encarga de administrar de manera transparente para el usuario la creación de memoria, las bifurcaciones y el paso de parámetros. Además, nos permite acceder directamente a los recursos de la máquina para un mejor desempeño.

**Lenguaje máquina**

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

**Características**

El lenguaje máquina realiza un conjunto de operaciones predeterminadas llamadas micro operaciones.  
Las micro operaciones sólo realizan operaciones del tipo aritmética (+,-,\*,/) lógicas (AND, OR, NOT) y de control (secuencial, de control y repetitiva).

Algunos microprocesadores implementan más funcionalidades llamado **CISC**, pero son más lentos que los **RISC** ya que estos tienen registros más grandes.

**Ventajas**

* Mayor adaptación al equipo.
* Máxima velocidad con mínimo uso de memoria.

**Desventajas**

* Imposibilidad de escribir código independiente de la máquina.
* Mayor dificulta 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.

**Direccionamiento**

Es la forma en cómo se accede a la memoria. Recordar que un programa no puede ejecutarse sino se encuentra en memoria principal.  
La forma de acceder a la memoria depende del microprocesador, pero en general existen dos tipos de direccionamiento:

* Directo: También recibe el nombre de direccionamiento absoluto y el acceso a las direcciones se hace de manera directa.
* Indirecto: También recibe el nombre de direccionamiento relativo y se basa a partir de una dirección genérica, generalmente el inicio del programa.

**Administración 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 memoria. Los lenguajes más recientes controlan el uso de punteros y tienen un programa denominado recolector de basura que se encarga de limpiar la memoria no utilizada mejorando el desempeño.

* **Reubicación:** La técnica de multiprogramación requiere que varios programas ocupen la memoria al mismo tiempo. Sin embargo, no se sabe con anticipación donde será cargado cada programa por lo que no es práctico usar direccionamiento absoluto de memoria.
* **Organización física:** Debido al costo de una memoria principal rápida, éste se usa en conjunto con una memoria secundaria mucho más lenta (y por consiguiente, barata) a fines de extender su capacidad.
* **Organización lógica:** Aunque la mayor parte de las memorias son organizadas linealmente con un direccionamiento secuencial, esto difícilmente concuerde con el camino seguido por el programa, debido al uso de procedimientos, funciones, subrutinas, arreglos, etc.

**Bibliografía**

M. en C. Pérez Pérez Isaías y L.S.C. Monroy Cedillo Jair Jonathan. (2013). Generador de Código objeto. 03/05/2018, de Universidad Autónoma del estado de Hidalgo Sitio web: <http://cidecame.uaeh.edu.mx/lcc/mapa/PROYECTO/libro32/autocontenido/autocon/137_generador_de__cdigo_objeto.html>

Generación de código objeto. 03/05/2018, de wordpress.com Sitio web: <https://ingarely.files.wordpress.com/2012/11/unidad-viii.pdf>

Mark Francisco. (2016). Generación de código objeto. 03/05/2018, de Blogspot Sitio web: <http://acaurio.blogspot.mx/2016/11/unidad-4-generacion-de-codigo-objeto.html>