Apéndice

C

Ensamblador

En este apéndice veremos qué es un ensamblador y como trabaja

Introducción

Un programa ensamblador (se llaman ensambladores porque lo que hacen es juntar varias rutinas en un único programa) de forma genérica es aquel que traduce un código en lenguaje ensamblador a lenguaje máquina (binario). En algunos casos se dice que es un programa traductor que convierte cada línea de ensamblador en una línea de código binario, pero esto no es del todo cierto sobre todo en programas ensambladores en los que se pueden construir macroinstrucciones.

Los lenguajes ensambladores se dividen normalmente (por su funcionamiento) en dos clases: ensambladores de un paso "*load-and-go* " y ensambladores de dos pasos, que primero determinan los símbolos del programa y después en el segundo paso los traducen adecuadamente.

Estos símbolos o esta información con que trabaja un ensamblador es de tres tipos:

1. Absoluta. Es independiente de la dirección donde vaya a estar cargado el programa, como por ejemplo cantidades.

2. Externa o global. Son variables que pertenecen a otros subprogramas y que no van a ser conocidos hasta que todo el programa sea enlazado.

3. Reubicable o localizable. En este caso la dirección dependerá de donde esté localizado el subprograma en sí, pero no de otros subprogramas, por lo que para hacer la dirección absoluta basta con sumar la dirección relativa (de dentro del programa) a la dirección de inicio.

A su vez, el ensamblador trabaja con varios tipos de instrucciones:

1. Simbólicas. Constan de varios campos: Etiqueta, añadida a una instrucción permite que se pueda referenciar lógicamente en el programa y su valor será la dirección que tome la instrucción; Código de operación, un símbolo obligatorio que indica la instrucción propiamente dicha que pertenece al conjunto de instrucciones de la CPU; Operando, es una dirección de datos cuya complejidad vendrá dada por el propio repertorio de instrucciones de la CPU, puede ser desde un número a un símbolo pasando por operaciones aritméticas; Comentario, algo que se pone de forma explicativa y que no es traducido a código máquina.

2. Pseudo-instrucciones. Son instrucciones que ayudan al proceso de ensamblado del programa, no traduciéndose a código máquina. Instrucciones típicas son las de comienzo y fin de programa o las de equivalencia de valores o símbolos.

3. Definición de constantes. Estas pseudo-instrucciones introducen una o varias constantes a partir de una posición de memoria que normalmente es referenciada con una etiqueta.

4. Reserva de espacio. Es otra pseudo-instrucción que sirve para reservar espacio en memoria. También la dirección de final o del principio suele tener asociada una etiqueta.

5. De subprogramas. Son pseudo-instrucciones que indican cuando una variable es local o global. Por norma sólo se utilizan con las variables globales, teniendo una forma para definirlas y otra para referenciarlas.

6. De origen. Indican al ensamblador donde el programa o los subprogramas se deben cargar en la memoria, si es que se sabe de antemano.

Proceso de traducción

El proceso de traducción se realiza de línea en línea, analizándose para cada una los campos introducidos y transformando éstos en códigos binarios. Para ello el ensamblador siempre tiene presentes dos tablas fijas:

1. Tabla de código de operación. Dónde están definidas toda las instrucciones de la CPU con sus correspondientes códigos binarios.

2. Tabla de pseudo-instrucciones. Aquí están definidas las instrucciones propias de ese programa ensamblador.

Además, el ensamblador construye una tabla variable que es la que define los símbolos utilizados en ese programa junto con su dirección.

Así, el proceso de traducción en el primer paso del ensamblador consistiría en:

1. Lectura de la sentencia.
2. Análisis (verificación de corrección)
    - Etiquetas
        - Localización en la tabla de símbolos y tomar dirección.
        - Si no está insertarla.
    - Código de operación
        - Localización en la tabla de códigos y escribir código.
        - Análisis del operando.
    - Pseudo-instrucción
        - Localización en la tabla de pseudo-instrucciones.
        - Tratamiento de la pseudo-instrucción.
        - Análisis del operando.

Como se ve el trabajo del primer paso del ensamblador (Analizador, reconocedor o scanner) es el de trabajar con tablas. Por lo que es fundamental que la búsqueda en las mismas esté muy optimizada, para ello se utilizan varias técnicas:

1. Búsqueda secuencial. Dada una tabla (formada por k símbolos que sirven de índice y sus k traducciones), se va comparando en ella el símbolo a encontrar con los valores índice de la tabla secuencialmente. El número de intentos medio será de N/2, donde N es la longitud de la tabla. Si se hace un ordenamiento de esta tabla con el número de recurrencias, el promedio puede disminuir.

2. Búsqueda logarítmica o binaria. En este caso se ordena la tabla con el valor binario del símbolo y en vez de hacer una búsqueda secuencial se empieza comparando con la mitad de la tabla para saber en que parte está, esto se repite con cada nueva mitad hasta encontrar el índice.

3. Indexación alfabética. No se tiene una tabla normal sino que ésta se divide en 26 subtablas una por cada letra del alfabeto, teniendo una tabla general con los orígenes de cada una de las subtablas. La búsqueda, en este caso, sólo se realizará en la subtabla correspondiente a la inicial del símbolo a encontrar.

4. Búsqueda en forma de árbol. Aquí la tabla tiene forma de árbol binario ordenado que permita la inserción ordenada y la búsqueda simultáneamente.

5. Búsqueda utilizando pilas. Se utiliza cuando un mismo índice o símbolo tiene varias traducciones y se utiliza únicamente la más reciente.

6. Búsqueda utilizando tablas de hash. Es parecido al de indexación alfabética, pero en este caso la división en subtablas se hace a través de una función de hash (desmenuzar).

Para completar la función del primer paso del ensamblador sólo hay que decir que también se encarga de decir de descubrir posibles errores en la introducción de los símbolos de las instrucciones (ver sí hay alguno que no existe).

Una vez verificado el programa y construidas las tablas de símbolos, se aplica la segunda fase del ensamblador que fundamentalmente lo que va a hacer en escribir el código binario resultante. Para ello la función que lleva a cabo se puede describir como:

1. Lectura del programa.
2. Tratamiento de las sentencias.
    - Operación
        - Búsqueda en la tabla de códigos de operación.
        - Obtención de código binario y longitud.
        - Actualizar contador de direcciones.
    - Pseudo-instrucción.
        - Localización en la tabla de pseudo-instrucciones.
        - Tratamiento de la pseudo-instrucción.
    - Análisis del operando.
        - Búsqueda si es símbolo en la tabla.
        - Obtención de dirección.
        - Si no es símbolo se obtiene el valor directamente.
3. Escritura de código objeto.

Tipos de ensambladores

Hasta aquí se ha visto el ensamblador básico más normal, pero existen otros tipos de ensambladores:

1. Ensambladores de un solo paso o de incremento. Se hacen las dos fases a la vez: la verificación y construcción de tablas; y su análisis y escritura del código. Sólo se utilizaban este tipo de ensambladores cuando la lectura del código era muy lenta (antiguas lectores de tarjetas).

2. Macroensambladores. La operación será la misma que la vista anteriormente para un ensamblador de dos pasos, con la diferencia que tendremos que construir una tabla para las macroinstrucciones (grupo de instrucciones con un nombre que puede ser llamado

desde otras partes del programa) que tienen como particularidad que su tamaño es variable, por lo que suele almacenar esta tabla en una lista encadenada.

3. Ensambladores cruzados. La única particularidad de éstos es que producen código objeto para otra máquina distinta a la que está ejecutando el ensamblador.