Practica 12.- “Cargador Absoluto”.

3.1. Cargador Absoluto.

OBJETIVO DEL TEMA. Diseñar el pseudocódigo del cargador absoluto.

Introducción:

CLASES DE CARGADOR

Un cargador es un programa de sistema que realiza principalmente el alojamiento de memoria y la "carga" de un programa en código objeto (originalmente almacenado en disco) en la propia memoria. Opcionalmente el cargador puede pegar o unir otros recursos al código objeto para que éste sea completamente funcional desde el intérprete de ordenes del sistema operativo. Los recursos a su vez pueden ser bloques de memoria de datos o de código, dispositivos de E/S, inclusive otros archivos ejecutables. La ligadura de tales recursos puede ser de naturaleza estática o dinámica. Esto nos lleva al siguiente proceso:

1. carga: lleva el programa objeto a la memoria para su ejecución.

2. relocalización: modifica el programa objeto de forma que puede cargarse en una dirección diferente de la localidad especificada originalmente.

3. ligado: combina 2 o mas programas objeto independientes y proporciona la Información necesaria para realizar referencias entre ellos.

Clasificación de cargadores de memoria.

Cargador absoluto: Código con direccionamientos directos (absolutos).
Cargador relativo: Código de posición independiente (re localizable).

La función principal de un cargador es colocar un programa objeto en la memoria e iniciar su ejecución. El cargador se denomina cargador absoluto porque carga el programa ejecutable en una posición fija de memoria... siempre. Aunque es posible determinar la posición inicial de carga al examinar el formato de archivo objeto (repasar el módulo 2). No es muy deseable dotar a nuestro sistema de un cargador absoluto pues limitaríamos mucho sus capacidades, por ejemplo la multitarea no sería posible. Para ello es mejor un cargador re localizable, relativo pues. Lo anterior significa que podemos cargar no sólo más de un programa sino que además en diferentes partes de la memoria del sistema. El cargador absoluto es simple y eficiente, sin embargo este esquema tiene varias desventajas potenciales. Una de las mas grandes es que el programador necesita especificar la dirección real a la que se cargara en memoria, si se trata de un computador muy sencillo, con poca memoria esto no crea muchas dificultades ya que de cualquier manera solo hay espacio para ejecutar un programa a la vez. Pero en un computador más avanzado a menudo se requieren ejecutar varios programas independientes compartiendo la memoria (y otros recursos del sistema). Esto significa que no se conoce con anterioridad donde será cargado el programa, para un uso compartido eficiente de la maquina es necesario escribir programas re localizables en lugar de programas absolutos. El uso de este tipo de cargador también dificulta el uso eficiente de subrutinas de biblioteca. La mayoría de estas bibliotecas contienen muchas más subrutinas de las que puede utilizar cualquier programa. Para hacer uso eficiente de la memoria es importante poder seleccionar y cargar con exactitud las

subrutinas necesarias. Esto no podría hacerse de una forma efectiva si todas las subrutinas tuvieran direcciones absolutas previamente asignadas.
El cargador relativo también llamado cargador re localizador, en este se utiliza un registro de modificación para describir cada parte del código objeto que ha de cambiar al relocalizar el programa, hay un registro de modificación para cada valor que se ha de cambiar durante la relocalización, cada registro especifica la dirección inicial y la longitud del campo cuyo valor se va a alterar y después describe la modificación a realizar. El esquema de registro de modificación es un medio conveniente para especificar la relocalización de programas, sin embargo no es muy adecuado para utilizar con todas las estructuras de la maquina. En una maquina que utiliza básicamente direccionamiento directo y tiene un formato de instrucción fijo, a menudo es más eficiente especificar la relocalización por medio de una técnica diferente.
Si el bit de relocalización correspondiente a una palabra de código objeto se iguala a 1, La dirección del programa se suma a esta palabra al relocalizar el programa. Un valor de un bit 0 indica que no es necesaria ninguna modificación. Si un registro de texto contiene menos de 12 palabras de código objeto, los bits correspondientes a las palabras no utilizadas se igualan a 0. De esta forma, la máscara de bis FFC (1111 1111 0100) del primer registro de texto específico que las 10 palabras de código objeto han de ser modificadas durante la relocalización También tenemos otra opción: la segmentación. Un programa segmentado también utiliza una estructura de árbol, sin embargo puede haber más de 3 niveles, y cada segmento puede tener varios puntos de entrada. El programador en cualquier caso puede utilizar un lenguaje de mandatos para asignar explícitamente programas a los segmentos de la estructura del árbol, en caso de no hacerse, el cargador asignara programas a segmentos analizando las referencias externas entre ellos. Los segmentos se cargan automáticamente bajo el control de un cargador residente, si un segmento no está cargado en un momento determinado, todas las referencias a los puntos de entrada de ese segmento se reemplazan por el cargador residente. Algunos computadores proporcionan una posibilidad de relocalización por hardware que elimina en parte la necesidad de que el cargador efectué la relocalización de programas. Por ejemplo:

Se consideran todas las referencias a memoria como relativas al inicio del área de memoria asignada al usuario. La conversión de esas direcciones relativas en direcciones reales se realiza al ejecutar el programa, pero el cargador todavía debe manejar la
Relocalización de subprogramas en relación con el ligado.

Tablas y lógica de un cargador ligador:
Ahora ya se puede presentar un algoritmo de un cargador ligador (y relocalizador), se Utilizan registros de modificación para que las funciones de relocalización y ligado se realicen por medio de este mecanismo.

El algoritmo de un cargador-ligador es mucho más complicado que el del cargador absoluto, la entrada de este cargador consta de un conjunto de programas objeto o secciones de control que se van a ligar. Una sección puede (y es común que lo haga) hacer una referencia externa a un símbolo cuya definición aparece más adelante en este flujo de entrada. En este caso, la operación de ligado requerida no se puede realizar hasta haber asignado una dirección al símbolo externo implicado.
La principal estructura de datos necesaria para el cargador ligador es una tabla de

símbolos externos TABSE (análoga TABSIM), se usa para almacenar el nombre y la dirección de los símbolos externos en el conjunto se secciones de control que se está cargando, la tabla también suele indicar en que sección de control se define el símbolo, entre otras variables importantes podemos encontrar a DIRPROG (dirección y carga del programa) que es la dirección inicial de la memoria donde se va a cargar el programa ligado, y DIRSC (dirección de la sección de control) que contiene la dirección inicial asignada a la sección de control que está examinando el cargador.

RELACION CARGADOR-SISTEMA OPERATIVO

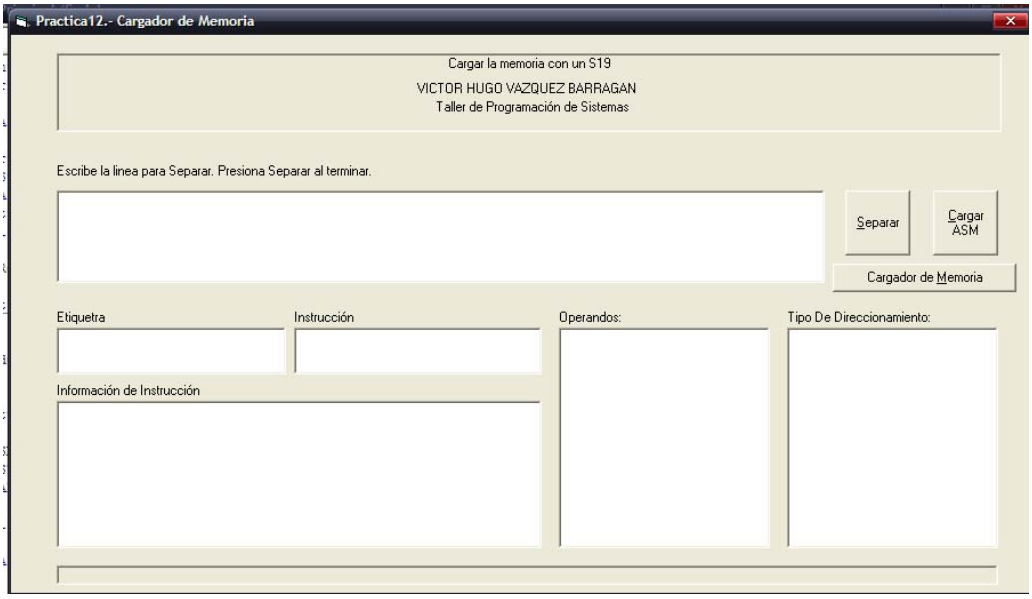
Un sistema operativo es un programa que actúa como intermediario entre el usuario y el hardware de un computador y su propósito es proporcionar un entorno en el cual el usuario que pueda ejecutar programas. El objetivo principal de un sistema operativo es, entonces, lograr que el sistema de computación se use de manera cómoda, y el objetivo secundario es que el hardware del computador se emplee de manera eficiente. Un sistema operativo es una parte importante de casi cualquier sistema de computación. Un sistema de computación puede dividirse en cuatro componentes: el hardware, el sistema operativo, los programas de aplicación y los usuarios. El principal objetivo de un sistema operativo es proporcionar una interfaz entre el equipo y los programas. Para lograr este objetivo, el sistema operativo debe administrar los recursos del sistema de cómputo, de manera tal que su utilización sea lo más sencilla y eficiente posible. El hardware (unidad central de proceso (UCP), memoria y dispositivos de entrada y salida (E/S)) proporciona los recursos de computación básica básicos. Los programas de aplicación (compiladores, sistemas de bases de datos, juegos de vídeo y programas para negocios) definen la forma en que estos recursos se emplean para resolver los problemas de computación de los usuarios. Puede haber distintos usuarios (personas, máquinas, otros computadores) que intentan resolver problemas diferentes; por consiguiente, es posible que haya diferentes programas de aplicación. El sistema operativo controla y coordina el uso del hardware entre los diversos programas de aplicación de los distintos usuarios. Los sistemas operativos son ante todo administradores de recursos; el principal recurso que administran es el hardware del computador: los procesadores, los medios de almacenamiento, los dispositivos de entrada/salida, los dispositivos de comunicación y los datos. Los sistemas operativos realizan muchas funciones, como proporcionar la interfaz con el usuario, permitir que los usuarios compartan entre sí el hardware y los datos, evitar que los usuarios interfieran
Recíprocamente, planificar la distribución de los recursos, facilitar la entrada y salida,
Recuperarse de los errores, contabilizar el uso de los recursos, facilitar las operaciones en paralelo, organizar los datos para lograr un acceso rápido y seguro, y manejar las
Comunicaciones en red. No existe una definición universal aceptada de qué forma parte de un sistema operativo y qué no. Una perspectiva sencilla considera como tal todo lo que envía un vendedor cuando se ordena la adquisición del "sistema operativo". Sin embargo, los requisitos de memoria y las características incluidas varían en forma considerable de un sistema a otro. Algunos requieren menos de un megabyte de espacio e incluso carecen de un editor de pantalla, mientras que otros necesitan cientos de megabytes de espacio e incluyen revisores ortográficos y sistemas de ventanas. Una definición más común es que el sistema operativo es el programa que se ejecuta todo el tiempo en el computador (conocido usualmente como núcleo), siendo programas de aplicación todos los demás. La segunda definición es la más común y es la que utilizamos en general.

La necesidad de un cargador relocalizador es una consecuencia de la rápida evolución de las computadoras, más poderosas y con más memoria, de tal manera que puede alojar no uno sino varios programas y entonces el sistema operativo (según la política de gestión de procesos) puede hacer que se ejecuten varias tareas. Pero desde luego primero se tiene que cargar el sistema operativo, por lo tanto necesitamos un cargador que desde el inicio o arranque de la computadora, aloje memoria para el S.O. Tal cargador es el llamado "Cargador de Arranque" o Bootstrap, el cual comúnmente es un cargador absoluto. Ese cargador aloja memoria para el cargador relativo del sistema operativo y una parte de éste, el conmutador de tareas... finalmente se carga todo el sistema operativo y el control de carga se pasa al cargador relativo incluido en éste.
El arrancador Bootstrap, se encuentra en una memoria de sólo lectura (ROM), la cual Tiene decodificadas direcciones fijas, lo que define al cargador absoluto. En dicha memoria ROM, se encuentran además otras rutinas pertinentes al arranque en frío de la computadora.

DESARROLLO DE LA PRÁCTICA

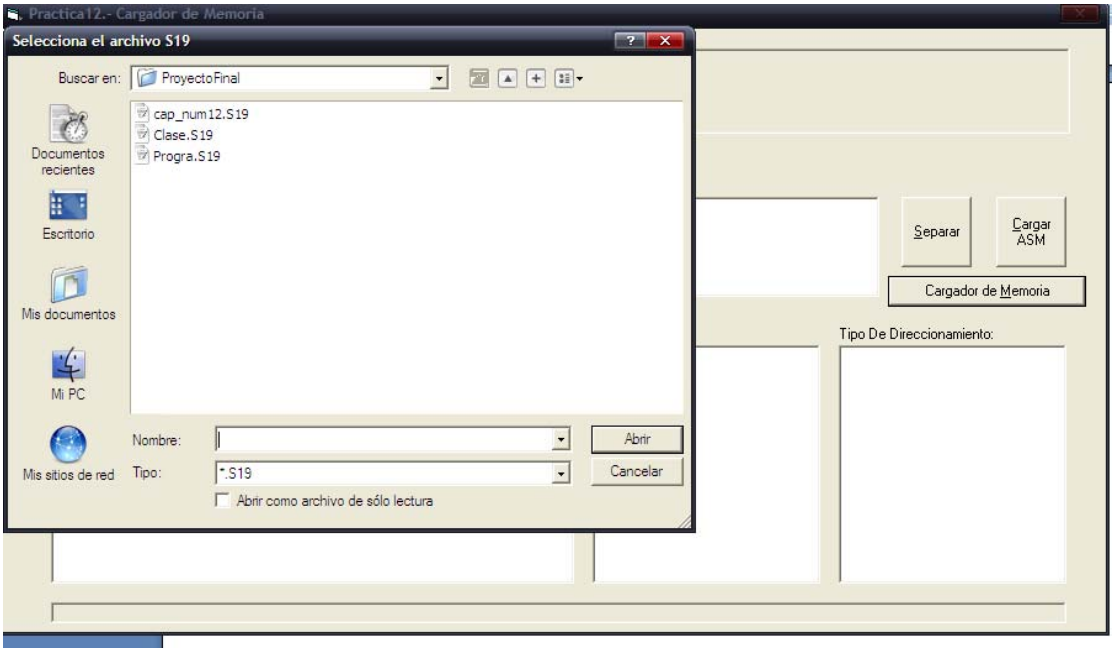
Esta práctica se desarrollo solo como un simulador de carga , y se comenzó desarrollando una función que si simule esto, la función se declaro como void asi es que no regresa ni ningún valor , en la cual declaramos un estructura en la cual se declaraba un arreglo de 1000 localidades , y esta estructura se declaro como un arreglo de la misma de 65535 localidades en la cuales se simulara la memoria de 64k , en esta función se lee del archivo s19 y solo los registros s1 son lo que nos interesa , descartamos las primeras cuatro letras, que corresponden al tipo de registro y al tamaño del registro, después obtenemos la dirección de localidad donde empieza el registro , y esta la convertimos en entero, de esta manera sirve como índice para el arreglo de estructuras que sirve de simulador de memoria. Después se imprime en pantalla cada localidad de memoria.

Utilicé Visual Basic 6 para desarrollar la aplicación. El diseño de la aplicación se muestra a continuación.



Se agrega un botón a la aplicación antes creada con el título “Cargador de Memoria”. Este botón pide un archivo de tipo S19 Para llenar un arreglo de 64K que simula la memoria del HC12 de Motorola y carga en ella los códigos máquina descritos en el S19.

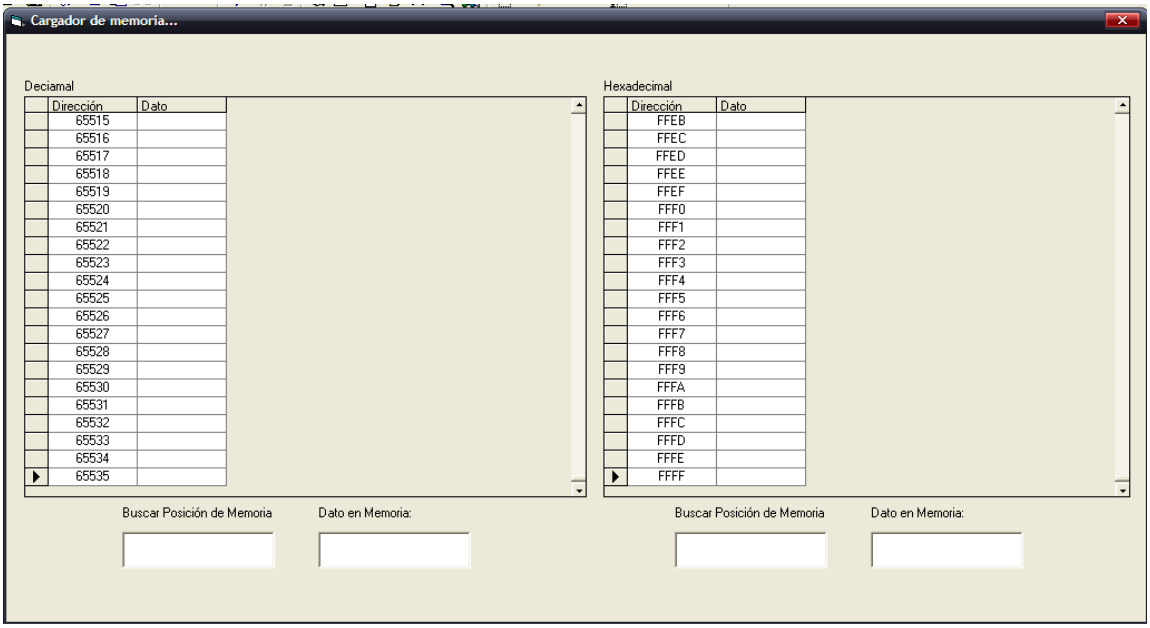
Un ejemplo de una corrida simple del programa.



Selección de archivo S19

Después de seleccionar el archivo, una función se encarga de recorrer el archivo y poner en cada dirección de memoria, que en el caso del arreglo que lo simula representaría su índice, los códigos máquina correspondientes.

El programa lanzará una ventana como la siguiente una vez que haya terminado de cargar los arreglos:



En la parte izquierda, se simula una memoria con direcciones en base decimal, mientras que a la derecha se simula lo mismo pero con direcciones de memoria en formato hexadecimal.

Además se agregaron cuadros de texto para realizar búsquedas en memoria de forma decimal o hexadecimal, según la que se seleccione.