Part

Unidad 1.- Introducción al lenguaje ensamblador

Alumno: Oliver Garibo Urias

* 1. **Importancia de la programación en lenguaje ensamblador.**

La importancia del lenguaje ensamblador radica principalmente que se trabaja directamente con el microprocesador; por lo cual se debe de conocer el funcionamiento interno de este, tiene la ventaja de que en el se puede realizar cualquier tipo de programas que en los lenguajes de alto nivel no lo pueden realizar. Otro punto sería que los programas en ensamblador ocupan menos espacio en memoria.

**Ventajas y desventajas del Lenguaje Ensamblador**

**Ventajas**

1.- Como trabaja directamente con el microprocesador al ejecutar un programa, pues como este lenguaje es el más cercano a la máquina la computadora lo procesa más rápido.

2.- Eficiencia de tamaño: Un programa en ensamblador no ocupa mucho espacio en memoria porque no tiene que cargan librerías y demás como son los lenguajes de alto nivel

3. Flexibilidad: Es flexible porque todo lo que puede hacerse con una máquina, puede hacerse en el lenguaje ensamblador de esta máquina; los lenguajes de alto nivel tienen en una u otra forma limitantes para explotar al máximo los recursos de la máquina. O sea que en lenguaje ensamblador se pueden hacer tareas específicas que en un lenguaje de alto nivel no se pueden llevar acabo porque tienen ciertas limitantes que no se lo permite

**Desventajas**

1.- Tiempo de programación: Como es un lenguaje de bajo nivel requiere más instrucciones para realizar el mismo proceso, en comparación con un lenguaje de alto nivel. Por otro lado, requiere de más cuidado por parte del programador, pues es propenso a que los errores de lógica se reflejen más fuertemente en la ejecución.

2.- Programas fuente grandes: Por las mismas razones que aumenta el tiempo, crecen los programas fuentes; simplemente requerimos más instrucciones primitivas para describir procesos equivalentes. Esto es una desventaja porque dificulta el mantenimiento de los programas, y nuevamente reduce la productividad de los programadores.

2.- Peligro de afectar recursos inesperadamente: Que todo error que podamos cometer, o todo riesgo que podamos tener, podemos afectar los recursos de la máquina, programar en este lenguaje lo más común que pueda pasar es que la máquina se bloquee o se reinicialice. Porque con este lenguaje es perfectamente posible (y sencillo) realizar secuencias de instrucciones inválidas, que normalmente no aparecen al usar un lenguaje de alto nivel.

3.- Falta de portabilidad: Porque para cada máquina existe un lenguaje ensamblador; por ello, evidentemente no es una selección apropiada de lenguaje cuando deseamos codificar en una máquina y luego llevar los programas a otros sistemas operativos o modelos de computadoras.

**Perspectiva histórica**

Los lenguajes ensambladores fueron primero desarrollados en los años 1950, cuando fueron referidos como lenguajes de programación de segunda generación. Por ejemplo, el SOAP (Symbolic Optimal Assembly Program) era un lenguaje ensamblador de 1957 para el computador IBM 650. Los lenguajes ensambladores eliminaron mucha de la propensión a errores y del consumo de tiempo de la programación de los lenguajes de primera generación que se necesitaba con los primeros computadores, liberando a los programadores del tedio tal como recordar códigos numéricos y cálculo de direcciones. Una vez fueron ampliamente usados para todo tipo de programación. Sin embargo, por los años 1980 (1990 en los microcomputadores), su uso había sido en gran parte suplantado por los lenguajes de alto nivel, en la búsqueda de una mejorada productividad en programación. Hoy en día, aunque el lenguaje ensamblador es casi siempre manejado y generado por los compiladores, todavía se usa para la manipulación directa del hardware, acceso a instrucciones especializadas del procesador, o para resolver problemas de desempeño crítico. Los usos típicos son drivers de dispositivo, sistemas embebidos de bajo nivel, y sistemas de tiempo real.

Históricamente, un gran número de programas han sido escritos enteramente en lenguaje ensamblador. Los sistemas operativos fueron casi exclusivamente escritos en lenguaje ensamblador hasta la aceptación amplia del lenguaje de programación [C](http://es.wikipedia.org/wiki/Lenguaje_de_programaci%C3%B3n_C) en los años 1970 y principios de los 1980. También, muchas aplicaciones comerciales fueron escritas en lenguaje ensamblador, incluyendo una gran cantidad del software escrito por grandes corporaciones para mainframes de IBM. Los lenguajes COBOL y FORTRAN eventualmente desplazaron mucho de este trabajo, aunque un número de organizaciones grandes conservaran las infraestructuras de aplicaciones en lenguaje ensamblador hasta bien entrados los años 1990.

La mayoría de los primeros microcomputadores confiaron en el lenguaje ensamblador codificado a mano, incluyendo la mayoría de los sistemas operativos y de las aplicaciones grandes. Esto era porque estos sistemas tenían limitaciones severas de recursos, impusieron idiosincráticas arquitecturas de memoria y de pantalla, y proporcionaron servicios de sistema limitados y con errores. Quizás más importante era la falta de compiladores de primera clase de lenguajes de alto nivel adecuados para el uso en el microcomputador.

* 1. **El procesador y sus registros**

El procesador, también conocido como CPU o micro, es el cerebro del PC. Sus funciones principales incluyen, la ejecución de las aplicaciones y la coordinación de los diferentes dispositivos que componen un equipo. No puede existir por tanto una máquina rápida que no tenga en su interior un micro potente.

Físicamente, no es más que una pequeña pastilla de silicio. Se coloca sobre la placa base en un conector que se denomina socket, aunque en un laptop o portátil lo normal es encontrarlo soldado. La placa base se convierte así en la encargada de permitir la conexión con los restantes dispositivos del equipo, como son la memoria RAM, la tarjeta gráfica o el disco duro usando para ello un conjunto de circuitos y chips denominado chipse. El encapsulado define como el micro se conecta a la placa base. Existen tres modelos, PGA, LGA y BGA. El último caso esta soldado a la placa y por lo tanto te será imposible cambiarlo para actualizarlo. Es muy usado en los laptops, como te comente anteriormente, ya que disminuye el tamaño total del equipo.

El procesador es uno de los elementos del PC que más ha evolucionado a lo largo del tiempo. Gracias a las mejoras en la tecnología de fabricación se ha reducido el tamaño de los transistores que se encuentran en su interior permitiendo integrar un mayor número de ellos. Estos elementos no son más que pequeños ladrillos que unidos configuran la funcionalidad del micro. Como se realiza esta interconexión es lo que se denomina arquitectura.

Gracias a estas mejoras, sobre todo en cuanto a área ocupada, se pueden incluir más bloques funcionales en su interior. En un principio fue el controlador de memoria, después la tarjeta gráfica y en un futuro muy cercano, pasaremos del concepto de procesador a lo que se denomina SOC, es decir, un chip con todos los elementos de la placa base en su interior.

Ten en cuenta que no todos son iguales pero la mayoría de ellos incluyen entre otros elementos:

**Núcleos**. Un núcleo no es más que un procesador en miniatura. Al tener varios, dentro del mismo procesador, podrás trabajar con más de una aplicación al mismo tiempo y puedes acelerar ciertos tipos de aplicaciones y evitar bloqueos.

**Cache**. El sistema de memoria es muy importante y se divide en varios elementos. La memoria cache, es el más cercano al micro ya que se encuentra en su interior. Se usa para mejorar la velocidad de los accesos a la memoria RAM. Si un dato se encuentra en la cache no tendrá que buscarlo en toda la RAM y por lo tanto el procesado final es mucho más rápido.

La cache se encuentra, a su vez, organizada en varios niveles cada uno más lento y grande que el anterior. Será tarea del micro dejar los datos que más se usen lo más cerca posible del micro. No olvides que la memoria RAM almacena tanto los datos como las instrucciones de tus programas y utilidades.

Controlador de memoria. Este fue uno de los primeros elementos que se quiso integrar en el micro. Se hizo porque al incorporar el controlador de memoria en el interior del procesador y quitarlo de la placa base se consigue aumentar la velocidad de la memoria RAM. Esto tiene un inconveniente y es que sólo puedes usar el tipo de memoria para la que tu procesador esté preparado. Antiguamente la memoria dependía de la placa base y no era raro que esta estuviera preparada para poder funcionar con varios tipos de memoria RAM.

**Tarjeta gráfica**. Si integran este componente ya no hablamos de CPUs sino de APUs. Ya no estaríamos ante un micro convencional si no ante un hibrido entre procesador y tarjeta gráfica. En la actualidad y parece que en desarrollos futuros vamos a tener siempre este tipo de dispositivos.

La inclusión de este elemento es fundamental ya que las tarjetas gráficas están compuestas de pequeños núcleos en su interior que pueden usarse para acelerar cierto tipo de aplicaciones. Al pasar al interior del micro pueden hacer su trabajo de una forma mucho más eficiente.

**Otros elementos**. Los micros han incorporado aún mas funcionalidad que antes se encontraba sobre la placa base. Por ejemplo, el controlador de PCI Express, aumentando la velocidad con la que el micro es capaz de comunicarse, por ejemplo, con una tarjeta gráfica discreta.

**REGISTROS DEL PROCESADOR**

Los registros del procesador se emplean para controlar instrucciones en ejecución, manejar direccionamiento de memoria y proporcionar capacidad aritmética. Los registros son direccionables por medio de un nombre. Los bits por convención, se numeran de derecha a izquierda, como en:

... 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0

Los registros internos del procesador se pueden clasificar en 6 tipos diferentes:

1. Registros de segmento
2. Registros de propósito general
3. Registros de apuntadores
4. Registros de banderas
5. Registros de Puntero de instrucción
6. Registros de Pila

**Registros de segmento**: Un registro de segmento tiene 16 bits de longitud y facilita un área de memoria para direccionamiento conocida como el segmento actual.

**Registro CS**. El DOS almacena la dirección inicial del segmento de código de un programa en el registro CS. Esta dirección de segmento, mas un valor de desplazamiento en el registro apuntador de instrucción (IP), indica la dirección de una instrucción que es buscada para su ejecución.

**Registro DS**. La dirección inicial de un segmento de datos de programa es almacenada en el registro DS. En términos sencillos, esta dirección, mas un valor de desplazamiento en una instrucción, genera una referencia a la localidad de un byte especifico en el segmento de datos.

**Registro SS**. El registro SS permite la colocación en memoria de una pila, para almacenamiento temporal de direcciones y datos. El DOS almacena la dirección de inicio del segmento de pila de un programa en el registro SS. Esta dirección de segmento, más un valor de desplazamiento en el registro del apuntador de pila (SP), indica la palabra actual en la pila que está siendo direccionada.

**Registros ES**. Algunas operaciones con cadenas de caracteres (datos de caracteres) utilizan el registro extra de segmento para manejar el direccionamiento de memoria. En este contexto, el registro ES está asociado con el registro DI (índice). Un programa que requiere el uso del registro ES puede inicializarlo con una dirección de segmento apropiada.

**Registros FS y GS**. Son registros extra de segmento en los procesadores 80386 y posteriores.