Introducción

Año tras año, los avances tecnológicos en el área de los sistemas de computación permiten la disminución en los costos y el aumento en el rendimiento y la capacidad de los mismos. Los avances en el hardware le han permitido a los programadores crear programas de gran utilidad. Los programadores éxitosos siempre han estado interesados en el rendimiento de sus programas. En las décadas de los 60 y 70, una restricción importante era el tamaño de la memoria. Los programadores buscaban minimizar el espacio de memoria para hacer programas más rápidos. Ahora los programadores interesados en rendimiento necesitan entender la naturaleza e implicaciones de la jerarquía de memoria (que reemplazó el modelo simple de los 60´s) y de los procesadores paralelos. Por lo tanto, los programadores que deseen construir versiones competitivas de compiladores, sistemas de operación, bases de datos y aplicaciones necesitan incrementar su conocimiento de la organización del computador.

Si bien es cierto que actualmente los precios de los computadores personales y los componentes nos han permitido contar cada vez con mas recursos, también lo es el hecho de que cada día surgen nuevos dispositivos como PDAs, teléfonos móviles, tarjetas inteligentes, etc, que debido a su naturaleza y su bajo costo cuentan con recursos de cómputo, almacenamiento y presentación (E/S) restringidos por lo que para poder aprovechar todas sus prestaciones se hace necesario un adecuado conocimiento de su arquitectura y el funcionamiento e interacción de sus componentes.

En este curso tendremos la oportunidad de ver qué sucede dentro del computador, desenmarañar el software que se encuentra por debajo de nuestros programas y el hardware que se encuentra dentro de la caja del computador.

¿ Qué hay debajo de nuestros programas ?

Un computador digital es una máquina que puede resolver problemas ejecutando instrucciones que recibe de las personas. La secuencia de instrucciones que describe cómo realizar cierta tarea se llama programa. Los circuitos electrónicos de un computador pueden reconocer y ejecutar directamente un conjunto limitado de instrucciones sencillas, y todos los programas tienen que convertirse en una serie de instrucciones sencillas que el computador puede ejecutar.

Las instrucciones sencillas o primitivas de un computador constituyen un lenguaje que permite a las personas comunicarse con el computador. Por ejemplo, el hombre para comunicarse con sus semejantes utiliza palabras que están conformadas por letras del alfabeto. Cuando requerimos comunicarnos con máquinas electrónicas lo vamos a hacer por medio de instrucciones sencillas pero para transmitirle estas instrucciones debemos enviar señales eléctricas. Las máquinas interpretan más fácilmente las señales *on* y *off*, pues es más sencillo interpretar la presencia o la ausencia de voltaje que rangos de voltaje. Por lo tanto, el alfabeto de las máquinas está constituido por dos símbolos que son representados por el 1 y el 0 respectivamente. A cada una de estos símbolos se le conoce como dígito binario o bit (**bi**nary digi**t**), y sobre este alfabeto se construyen los comandos o instrucciones con los cuales nos comunicamos. Las instrucciones que ejecutan los computadores son colecciones de bits que pueden ser vistos como números. Por ejemplo, el siguiente patrón de bits:

1000110010100000

le indica al computador que debe sumar dos números. A este lenguaje que permite comunicarnos con el computador se le conoce como lenguaje de máquina.

Los primeros programadores se comunicaban con los computadores por medio de números binarios. Este tipo de comunicación resultaba tediosa y muy difícil de manejar. Rápidamente inventaron otras notaciones intermedias entre la forma de trabajar de los

computadores y la forma de pensar de los humanos. En un principio, esta nueva notación era traducida a números binarios de forma manual. Posteriormente se desarrollaron programas, conocidos como ensamblador, que se encargaban de realizar la traducción de forma automática.

Por ejemplo, el programador escribía

add A, B (lenguaje ensamblador)

y el ensamblador traducía esta notación en

1000110010100000 (lenguaje de máquina)

La introducción del lenguaje ensamblador produjo mejoras en cuanto a la programación pero aún asi seguía siendo tedioso el uso de este tipo de lenguaje simbólico. El lenguaje ensamblador obliga al programador a escribir una línea para cada instrucción que debe ejecutar la máquina forzando al programador a pensar como la máquina.

Con el objetivo de expresar los programas en un lenguaje que esté más cercana a la forma de pensamiento de los programadores y a las estrategias de resolución de los problemas más que a los esquemas de funcionamiento de las máquinas surgen los lenguajes de alto nivel. Para que los computadores puedan procesar estos nuevos tipos de lenguajes se desarrollaron otros programas que traducen o interpretan programas escritos en un lenguaje de alto nivel a lenguaje ensamblador. En un proceso de traducción de un programa escrito en el lenguaje L2 al lenguaje L1, se sustituyen todas las instrucciones del programa en L2 por instrucciones de L1, para luego ejecutar el programa que se obtuvo en L1. Esta traducción no es necesariamente una sustitición 1 a 1. En el caso de interpretación, un programa escrito en L2 es ejecutado por medio de un programa escrito en L1 que toma como dato las instrucciones del programa en L2.

En la figura 1 se muestra un ejemplo de un programa escrito en un lenguaje de alto nivel (C) que es traducido a un programa en lenguaje ensamblador por medio del compilador.

Este nuevo programa en lenguaje ensamblador es traducido a lenguaje de máquina por medio del ensamblador.

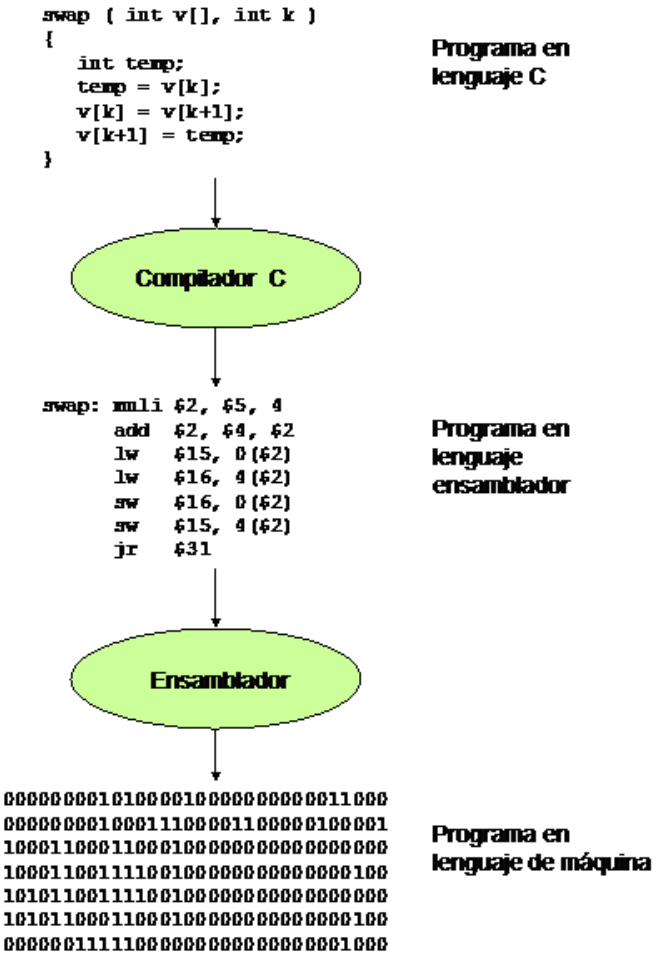

Figura 1. Programa en C compilado en lenguaje ensamblador y después ensamblado a lenguaje de máquina

El uso del lenguaje ensamblador le permite al programador indicarle al computador exactamente cómo llevar a cabo una tarea específica usando la menor cantidad de instrucciones. Aún cuando el código generado por los compiladores con opción de optimización es eficiente, la optimización manual puede resultar en una mejora sustancial en términos de rendimiento y consumo de memoria. El lenguaje ensamblador es usualmente utilizado en las siguientes circunstancias:

- Mejorar la eficiencia de una rutina específica que se ha transformado en un cuello de botella

- Obtener acceso a funciones de bajo nivel del procesador para realizar tareas que no son soportadas por los lenguajes de alto nivel
- Escribir manejadores de dispositivos para comunicarse directamente con hardware especial tales como tarjetas de red
- Trabajar en ambientes con recursos limitados puede requerir el uso del lenguaje ensamblador pues el código ejecutable puede ser menor que el generado por el compilador

Cuando se diseña una nueva máquina se debe decidir qué instrucciones serán incluidas en su lenguaje de máquina. Generalmente, se busca que las instrucciones primitivas sean lo más simple posible a fin de reducir la complejidad y el costo de los circuitos requeridos. Casi todos los lenguajes de máquina son tan simples que para las personas resulta difícil y tedioso usarlos.

Para resolver este problema, se diseña un nuevo conjunto de instrucciones que para las personas es más fácil de usar que el conjunto de instrucciones de la máquina original. Estas nuevas instrucciones conforman el lenguaje L1. Para que la máquina pueda ejecutar programas en este nuevo lenguaje es necesario llevar a cabo la traducción del programa en L1 a un programa en lenguaje de máquina o la interpretación del programa escrito en L1. En lugar de pensar en términos de traducción o interpretación, es más fácil imaginar la existencia de un computador hipotético o máquina virtual cuyo lenguaje de máquina es L1. Este proceso se puede repetir y tener varios niveles de máquinas virtuales.

Generalmente, los computadores se diseñan como una serie de niveles, cada uno construido sobre sus predecesores. Cada nivel representa una abstracción distinta y contiene objetos y operaciones. Una máquina multinivel tendría definidas varias

máquinas virtuales y un mecanismo de traducción o interpretación entre máquinas virtuales de niveles adyacentes.

Las abstracciones son usadas constantemente en la vida real. Un mapa es una abstracción y podemos encontrar varios niveles de abstracción en los mismos. Desde mapas del planeta que muestran los límites entre países, masas importantes de agua, ciudades importantes hasta mapas que muestran detalles físicos tales como montañas, desiertos, lagos, ríos. Un mapa de carreteras es otra abstracción muy usada.

Cuál es la utilidad de estas abstracciones? Un modelo abstracto permite ignorar detalles mientras captura la esencia del problema. Asociado al modelo encontramos un conjunto de primitivas u operaciones elementales que pueden ser aplicadas. En el área de nuestro interés, el uso de abstracciones nos ayuda a entender los sistemas. Un sistema de computación es complejo y consiste en millones de componentes. Una serie de modelos de abstracción han sido elaborados para ayudarnos por ejemplo a predecir cómo se comportará el sistema en una situación dada. Es por esto que los ingenieros agregan capas de software para transformar sistemas de circuitos electrónicos en una herramienta de apoyo y de esta forma se le presenta al usuario una interfaz o máquina virtual que es más fácil de entender y programar.

Un modelo de máquina multinivel que se adapta a la mayoría de los sistemas computacionales actuales se muestra en la figura 2.

El nivel 0 corresponde al hardware de la máquina. Sus circuitos ejecutan los programas en lenguaje de máquina del nivel 1. Los objetos que integran este nivel se conocen como compuertas. Se construyen con componentes analógicos pero pueden modelarse como dispositivos digitales. Cada compuerta tiene una o más entradas digitales (señales que representan 0 o 1) y para generar su salida calcula alguna función sencilla de dichas entradas, como AND u OR. Podemos combinar compuertas para formar una memoria

de un bit capaz de almacenar un 0 o un 1. A su vez, las memorias de un bit pueden combinarse para formar registros. Cada registro puede contener un solo número binario menor que un cierto valor límite. Las compuertas también pueden combinarse para formar la máquina calculadora principal.

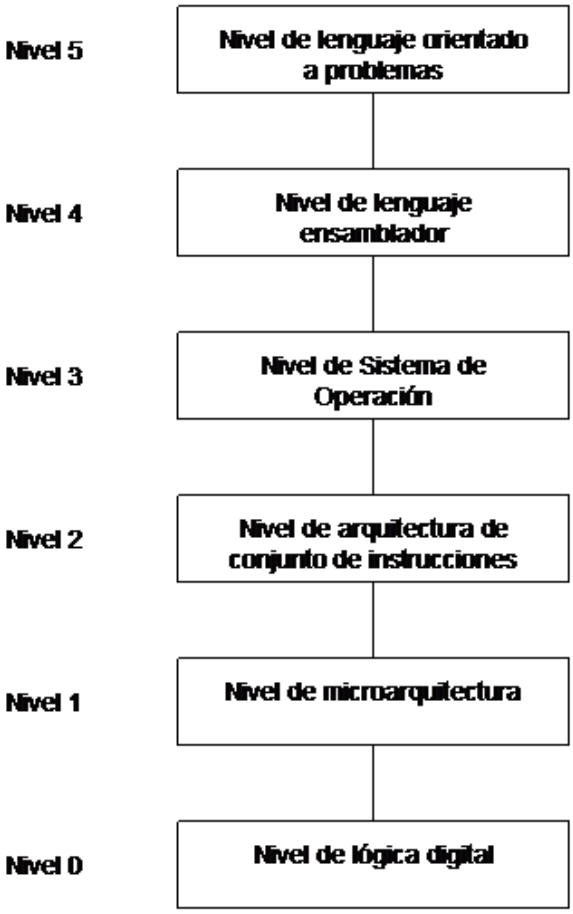

Figura 2. Modelo de máquina multinivel

En el nivel de microarquitectura encontramos una colección de registros que forman una memoria local y un circuito llamado ALU (Unidad Lógica Aritmética) que puede efectuar operaciones aritméticas y lógicas sencillas. Los registros se conectan a la ALU para formar una trayectoria de datos por donde fluyen los datos (datapath). La operación

básica de la trayectoria de datos consiste en seleccionar uno o dos registros, hacer que la ALU opere con ellos y almacenar el resultado en algún registro.

En algunas máquinas un programa conocido como microprograma controla la operación de la trayectoria de datos. En otras máquinas, la trayectoria de datos está bajo el control directo del hardware. El microprograma es un intérprete de las instrucciones en el nivel 2: obtiene, examina y ejecuta las instrucciones una por una utilizando la trayectoria de datos.

En el nivel 2 tenemos el nivel de arquitectura del conjunto de instrucciones o nivel ISA (Instruction Set Architecture). Este nivel incluye el conjunto de instrucciones que el microprograma o los circuitos de ejecución en hardware ejecutan.

El nivel 3 suele ser un nivel híbrido. Casi todas las instrucciones de su lenguaje se encuentran en el nivel ISA. Además hay un nuevo conjunto de instrucciones, una organización diferente de la memoria, capacidad para ejecutar dos o más programas al mismo tiempo.

El nivel 4 o nivel de lenguaje ensamblador, es en realidad una forma simbólica de uno de los lenguajes subyacentes. Los programas en lenguaje ensamblador primero se traducen a un lenguaje de nivel 1, 2 o 3 y luego son interpretados por la máquina virtual o real apropiada. El programa que realiza la traducción se llama ensamblador.

El nivel 5 está conformado por los lenguajes diseñados para ser usados por los programadores de aplicaciones. Tales lenguajes suelen llamarse lenguajes de alto nivel.

¿ Qué hay dentro del computador ?

Los cinco componentes clásicos de un computador son: Entrada (Input), Salida (Output), Memoria, Camino de datos (datapath) y Control (Figura 3).

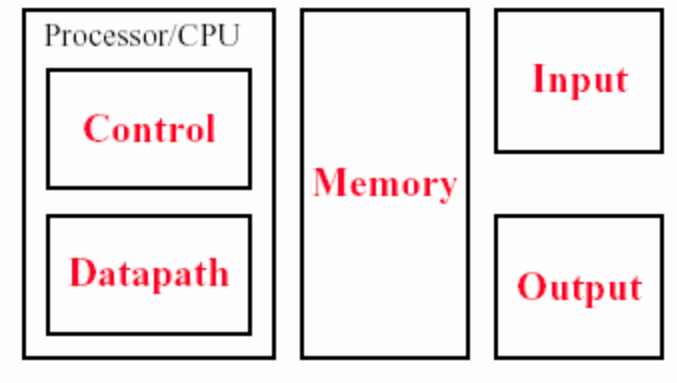

Figura 3. Componentes de un computador

Cuando trabajan con un computador, habrán notado que generalmente disponemos al menos de dispositivos de entrada como el teclado y el ratón y dispositivos de salida como el monitor. Con los dispositivos de entrada alimentamos el computador y a partir de los dispositivos de salida obtenemos los resultados.

Si abrimos la caja del computador, encontraremos una tarjeta (tarjeta madre) cubierta con docenas de rectángulos grises o negros. Los pequeños rectángulos contienen circuitos integrados o chips. La tarjeta está conformada por tres piezas importantes: la pieza que conecta con los dispositivos de E/S (bus), la memoria y el CPU.

La memoria es donde se almacenan los programas y los datos necesarios mientras se ejecutan los programas. El procesador es la parte activa de la tarjeta. Adiciona números, compara, envía señales a los dispositivos de E/S etc. El procesador comprende dos componentes importantes: camino de datos y el control. El camino de datos lleva a cabo

las operaciones y el control indica que se debe hacer para llevar a cabo la ejecución de una instrucción del programa.

Referencias Bibliográficas

- Tanenbaum, Andrew S. Organización de computadoras. Un Enfoque Estructurado. 4ta Edición. Prentice Hall
- Stalling, William. Organización y Arquitectura de Computadoras. Diseño para optimizar prestaciones. 5ta Edición Prentice Hall.
- Patterson, D.A., Hennessy, J.L. Computer Organization & Design: The Hardware/Software Interface. Morgan-Kaufmann

Revisado y modificado por el Prof. Ricardo González (Septiembre 2004)