Arquitectura de las Computadoras

Trabajo Práctico Nº 4

Assembly de la arquitectura Intel x86

Ing. Walter Lozano
Ing. Alejandro Rodríguez Costello

La arquitectura Intel x86 es de amplia difusión en el mercado pues constituye la base de las PC compatibles de escritorio. En sus inicios los registros tenían 16 bits y no son de uso general como en MIPS. De allí que se tome como al *word* con este tamaño.

Es evidente que en un laboratorio no podemos abordar profusamente esta arquitectura, pero a traves de conocimientos previos podemos intentar una práctica sencilla que revele las profundas diferencias con la arquitectura estudiada.

Hay una cantidad copiosa de literatura de gran calidad sobre Intel, pero la sobre abundancia puede producir un efecto negativo pedagógicamente. Por eso tomamos el libro "Lenguaje Ensamblador para PC" del Dr. Paul Carter[1] como referencia. Este se adapta correctamente a las necesidades del curso y de este trabajo en particular.

Abordaje de la literatura

Aunque el texto no es extenso, la cátedra considera adecuado sugerir la forma en que debiera abordarse el texto y en sus componentes escenciales. Siéntase libre de realizar dicha lectura como más le parezca conveniente.

Dentro del capítulo 1, en la sección 1.2 está la descripción básica de la arquitectura. Los puntos 1.2.4 al 1.2.8 son básicos para comprender las diferencias con el banco de registro de MIPS. En la sección 1.3 se encara específicamente el lenguaje ensamblador desde la perspectiva de nuestro compilador de assembly llamado **nasm**. En 1.3.4 tenemos algunos ejemplos de instrucciones básicas, en 1.3.5 se abordan las directivas, en 1.3.6 el manejo de I/O, mediante una interface creada específicamente por el autor y en 1.3.7 vemos como hacer depuración. Debemos recordar que no estamos en presencia de un emulador sencillo, sino utilizando una máquina virtual cuya complejidad es la misma de una máquina real, por lo cuál el entorno de trabajo es radicalmente diferente al **pcspim**. La sección 1.4 contiene la explicación de la mayoría de los comandos a utilizar y como operan. Su lectura no es necesaria para la realización del trabajo porque el mismo está guiado, pero es escencial para entender lo que ocurre y seguramente necesaria al momento de evaluación final.

El capítulo 2 y 3 deberían tomarse como material de consulta en caso de necesitar asistencia con el comportamiento de algunas instrucciones. Por ejemplo allí encontrará como se usan las instrucciones de multiplicación.

Finalmente en el capítulo 4 dedicado a subrutinas encontramos todo el material para abordar el ABI de nuestro entorno de trabajo. Su lectura es fundamental y excluimos solo la sección final 4.8 sobre código reentrante y subrutinas recursivas.

[1]Lamentablemente la traducción deja bastante que desear como pasa con mucha literatura pobremente traducida (nota del docente).

Espacio de trabajo

Para llevar adelante esta práctica utilizaremos una máquina virtual. Esta correrá mediante el virtualizador de Software Libre Oracle **VirtualBox**. El mismo está disponible para todos los sistemas operativos usuales. Es importante respetar esta consigna porque el entorno a utilizar es una distro de Linux de 32 bits "Debian 6.0 Squeeze" con un kernel 2.6.32 y todas las herramientas necesarias.

Para ello se ha dispuesto un arhico de aprox. 300 Mb llamado **mini.ova** que podrá importar desde el menú *File > Import Appliance* que le permitirá crear la máquina con todos los requisitos virtuales de hardware y software. Si necesita alguna aclaración se recomienda utilizar el foro correspondiente.

La consigna

Sintetizar la arquitectura x86 es un trabajo titánico que el Dr. Paul Carter ha llevado a cabo con sus limitaciones, motivo por el cuál la cátedra **no ha optado** por su uso para entender los aspectos prácticos de la programación de bajo nivel de procesadores. Sin embargo, dado la masividad de su presencia en el mercado y debido a las enormes diferencias que obstenta con MIPS R2000, se hace necesario establecer un **contraste**.

La primer parte de la consigna consiste en realizar una breve práctica usando los ejemplos provistos en la máquina virtual y en clase. Se le solicita que reconstruya las sesiones de trabajo con los comandos utilizados para compilar y linkear. Finalmente debe programar su propio *driver* en C, para invocar una rutina llamada *det* escrita en assembly de Intel x86 con el prototipo *int det(int a, int b, int c, int d)* que devuelva la resta de los productos cruzados a.c – b.d[2].

Luego se le pide que reevea con detalle **todo lo realizado** e informe las diferencias sustanciales entre arquitecturas por escrito. Es importante que en esta segunda parte no acopie el material del libro o internet sino que realice sus **propias conclusiones**. Si dicha tarea tuvo éxito, entonces deberá ser capaz de abordar conceptualmente los siguientes items:

- Diferencias en cantidad y uso obligatorio o no de los registros.
- Importancia del registro acumulador.
- Cantidad y forma de los operandos en las intrucciones.
- Diferencias en las instrucciones aritméticas.
- Diferencias en cantidad y modos de direccionamiento. Modo implícito.
- Complejidad del formato de las instrucciones.
- Importancia de los FLAGS y su análogo en MIPS (ALU).
- Diferencias de evaluación de condiciones y su implicación en los saltos.
- Diferencias en el manejo de pila.
- Diferencias en la gestión de subrutinas.

Sus conclusiones deben estar justificadas a traves de la ejemplificación mediante **código** de la práctica. A pesar su simplicidad, esta práctica tiene un alcance conceptual muy importante y vasto. Estos interrogantes bien pueden encontrarse en la extensa literatura que existe sobre el tema, pero sus respuestas debieran surgir solo del análisis del código utilizado y del libro de cátedra.

[2]A este número se le llama determinante de una matriz de 2x2.