Trabajo Práctico
Mini–Assembler

R-222 Arquitectura del Computador

Consultas: fedebergero@gmail.com

Introducción

El objetivo de este trabajo es que el alumno aprenda los detalles de la codificación binario de instrucciones de máquina i386 y que codifique algunas de ellas.

Metodología

Para realizar el trabajo el alumno debe resolver los siguientes puntos:

- El alumno deberá primero generar programas assembler que resuelvan los problemas de tests. No debe utilizarse segmento de datos.
- Luego deberá codificar una a una las instrucciones con sus operando correspondientes en opcodes binarios para i386. Aquí puede obviar los marcos de activación.
- Deberá comentar cada opcode explicando instrucción correspondiente y argumentos.

  Por ejemplo: `\x8b\x44\x24\x04` que equivale a un `movl 4(%esp),%eax`:
  0x8b: MOV mem32,reg32
  0x44: Guarda lo apuntado por un registro+offset en %eax
  0x24: Utiliza %esp como registro de indirección
  0x04: Offset igual a 4
- Utilizando la función provista `generate(void *code,int size)` deberá generar objetos elf y probarlos ejecutándolos. Esta función genera un objeto elf llamado `foo.o` que define una función `fun` que debe ser linkeada junto con los programas de ejemplo.
- Los saltos a etiquetas son traducidos por el assembler en saltos relativos a la dirección actual. Por ejemplo en:

```
 ...
 0x00000000 addl %eax,%ecx
cont:
 0x00000002 cmpl $0,%ecx
 0x00000004 je cont
```

el `je` se traduce en un salto en -4, por lo cual agregar o quitar instrucciones modifican los saltos relativos. Por ello se sugiere traducir el programa inicialmente sin instrucciones de salto e introducir dos instrucciones `nop` y luego una vez terminado el programa, reemplazarlas por los saltos correspondientes.

Problemas de tests

- El `test1` debe implementar una función que retorne el valor absoluto de un `short`.
- El `test2` debe implementar una codificación XOR de la string dada como argumento, con el código que se pasa como segundo argumento.
- El `test3` debe implementar una función que cuente los bits de un entero.
- El `test4` debe implementar una función sume los enteros de un arreglo dado como argumento.

Características adicionales

El alumno puede extender el trabajo (opcionalmente) con las siguientes mejoras:

- Generar un nuevo caso de ejemplo que calcule el discriminante de una ecuación cuadrática utilizando instrucciones de punto flotante. Esto es, dados tres `double a,b,c` calcule b*b-4*a*c.
- Extender el generador de código para que permita generar etiquetas y saltos a estas etiquetas.

Entrega del Trabajo

El trabajo será evaluado por la cátedra mediante una presentación en computadora. El alumno debe entregar un informe de al menos dos páginas incluyendo datos académicos (integrantes del grupo, legajos, fechas) y reportando problemas y soluciones encontradas durante la realización del trabajo y posibles extensiones al mismo

Material y Referencias

X86 Opcode and Instruction Reference
Manuales Intel i386
SYSTEM V APPLICATION BINARY INTERFACE