Índice

Sentencias de asignación

Área de Arquitectura y Tecnología de Computadores – Versión 1.0.886, 10/02/2022

Objetivos

Este bloque tiene como objetivo comprender la traducción de programas escritos en lenguaje de alto nivel al lenguaje del Computador Teórico. En esta sesión se presentará el simulador del Computador Teórico que se usará a lo largo del bloque práctico y se utilizará para estudiar la traducción de sentencias de asignación y aritmético-lógicas desde un lenguaje de alto nivel que es C++.

Conocimientos y materiales necesarios

Para aprovechar adecuadamente esta sesión de prácticas, el alumno necesita:

- Conocer los sistemas de codificación binario natural y complemento a 2, así como tener soltura en la conversión a hexadecimal.
- Conocer el lenguaje de programación C++.
- Conocer el lenguaje ensamblador del Computador Teórico.
- Conocer conceptos básicos de traducción de sentencias de alto nivel, tales como sentencias de asignación y aritmético-lógicas, a lenguaje máquina.
- Durante la sesión se plantearán una serie de preguntas que puedes responder en el correspondiente cuestionario en el Campus Virtual. Puedes abrir el cuestionario en otra pestaña del navegador pinchando en el enlace mientras mantienes pulsada la tecla `Ctrl`.

Desarrollo de la práctica

A lo largo de esta sesión, se presentarán pequeños fragmentos de código en C++, junto con indicaciones de cómo se almacenarán las variables declaradas en esos fragmentos (si se almacenan en un registro, en cuál sería, o si por el contrario se almacenan en memoria, en qué dirección estarían). Se te pedirá que hagas el papel de un compilador y escribas un corto fragmento de código en lenguaje ensamblador del Computador Teórico que sea equivalente en su funcionalidad al fragmento de C++ propuesto.

Para cada ejercicio propuesto deberás crear un fichero llamado `Ejercicio-N.txt`, siendo `N` el número del ejercicio, y con un editor de texto escribir en ese fichero la solución al ejercicio. Cualquier editor es válido, como por ejemplo el Bloc de Notas de Windows, u otro que prefieras, como `Notepad++`.

Para comprobar si la solución que has ideado es correcta, accederás a la siguiente dirección web:
http://www.atc.uniovi.es/tools/CPU/simulador.php

En esta página encontrarás un simulador del Computador Teórico que te permitirá introducir el código que has escrito (copiando y pegando desde el editor) y ejecutarlo para comprobar que, tras la ejecución, todos los registros tienen el valor esperado. Antes de ejecutar el programa puede ser necesario cargar ciertos valores iniciales en los registros, o en ciertas posiciones de memoria. Si así fuera, el ejercicio lo indicará en su momento.

> **Importante:** todos los valores iniciales que se especifiquen para registros o memoria a través de la interfaz web, se interpretan en hexadecimal, siendo la `h` final opcional. Sin embargo, los datos inmediatos que aparezcan en el listado ensamblador de la interfaz web ***sí*** deberán llevar la `h` final, o de lo contrario se entenderá que están en base 10.

Por defecto, este simulador mostrará en pantalla los valores de todos los registros antes y después de que se ejecute el código introducido, coloreando en rojo aquellos que han cambiado de valor. Es posible asimismo indicar ciertas posiciones de memoria cuyo contenido se quiere observar también. Para ello basta escribir la dirección de memoria deseada bajo la columna "Dirección", y marcar la casilla "Mirar" de esa dirección. También aquí se puede especificar un valor inicial para el contenido de esa dirección.

Si el resultado no es el que se esperaba, habrá que localizar la causa y corregirla. Para este cometido puede ser muy útil activar la opción "Obtener trazado de cada instrucción". Cuando esta opción está activa, no sólo se muestran los estados inicial y final, sino también el estado de todos los registros tras la ejecución de cada una de las instrucciones. De este modo puedes comprobar en qué momento el resultado deja de ser el que se esperaba y comprender mejor las causas del fallo.

1. Asignación y expresiones simples

- **Ejercicio 1.** Traduce al ensamblador del Computador Teórico la siguiente asignación en C++. Cada variable se almacena en un registro, indicado en el comentario.

```
a=b;    // Con a en r3,  b en r5
```

  Para probar tu código, pon en el simulador web un valor inicial de 7 al registro `r5`. Comprueba que, tras ejecutar el código, el registro `r3` adquiere también el valor 7.

- **Ejercicio 2.** Convierte al ensamblador del Computador Teórico el siguiente fragmento de código, suponiendo que la variable `c` está almacenada en el registro `r1`.

```
c=65;   // Con c en r1
```

  Observa que en el código C++ el dato introducido se da en base 10. A la hora de escribirlo en ensamblador puedes darlo también en base 10, o en hexadecimal, como prefieras. En este segundo caso deberás añadir una `h` al final del dato en el listado ensamblador. Sin embargo, el simulador web te hará el volcado de los registros siempre en hexadecimal. Comprueba que tras la ejecución del programa, el registro `r1` contiene el valor `0041h` (hexadecimal).

- **Ejercicio 3.** Repite el primer ejercicio, pero ahora considera que la variable `b` está almacenada en la dirección de memoria `0500h`, es decir:

```
a=b;    // Con a en r3,  b en memoria (0500h)
```

  Si necesitas registros auxiliares puedes utilizar cualquiera de los que no estén siendo usados para otra cosa. Para comprobar el correcto funcionamiento, usa el formulario web para indicar que la dirección de memoria `0500h` contiene el valor inicial `1234h` (puedes marcar también "Mirar" para comprobar el valor de esa dirección durante la ejecución del programa). Verifica que tras ejecutarse tu programa, el registro `r3` (que es el que contiene la variable `a`) termina con el valor `1234h`, que ha tomado de esa dirección de memoria.

- **Ejercicio 4.** Considerando, al igual que en el ejercicio anterior, que la variable `b` está almacenada en `0500h`, traduce el siguiente fragmento C++ al ensamblador del Computador Teórico.

```
b=27;    // Con b en memoria, en la dirección 0500h
```

  Si necesitas registros auxiliares puedes utilizar cualquiera de los que no estén siendo usados para otra cosa. Usa el formulario web para "Mirar" la dirección de memoria `0500h`, que debe comenzar con el valor `0000h`, y comprueba que al terminar el programa tiene el valor `001Bh` (que es 27).

- **Ejercicio 5.** Traduce al ensamblador del Computador Teórico la siguiente asignación, que contiene una sencilla expresión. Supón que la variable `a` se almacena en la dirección de memoria `0501h`, mientras que las variables `b` y `c` lo hacen en los registros `r1` y `r2`, respectivamente.

```
a = (a + b) - (c + 2);    // Con a en memoria (0501h), b en r1, c en r2
```

  Para comprobarlo, fijemos los siguientes valores iniciales (que damos aquí en base 10): `a=16`, `b=5`, `c=7`. Según estos valores ¿cuál habría de ser el valor final de `a`? Responde en el cuestionario: pregunta 1.

  Inicializa en el simulador web los registros `r1` y `r2` con los valores iniciales de `b` y `c`, respectivamente, y la dirección de memoria `0501h`, donde se guarda `a`, con el valor `0010h` (que es 16 en hexadecimal). Comprueba que tras ejecutarse tu código, la dirección `0501h` contiene el valor que habías predicho para `a`. Observa asimismo que la variable `a` es la única que resulta modificada en la expresión. Las variables `b` y `c` (es decir, los registros `r1` y `r2`) han de conservar sus valores iniciales. Si necesitas registros auxiliares para operaciones intermedias puedes usar `r3`, `r4` y `r5`.

  Ya que esta expresión, a pesar de su aparente sencillez, da lugar a varias líneas de código en ensamblador, puede resultar útil activar la opción "Obtener trazado de cada instrucción" en el simulador web.

2. Ejercicios adicionales

Intenta realizar ahora los siguientes ejercicios que tienen algo más de dificultad:

- **Ejercicio 6.** Repite el ejercicio 5, pero en esta ocasión las tres variables estarán en memoria, en concreto, `a` estará en la dirección `0501h`, `b` en `0502h` y `c` en `0503h`:

```
a = (a + b) - (c + 2);    // Con a, b y c en memoria (0501h, 0502h y 0503h, resp)
```

  Para comprobar el funcionamiento, inicializa las direcciones antes enumeradas de forma que los valores iniciales de las variables sean `a=9`, `b=8`, `c=3`. Con estos valores, ¿cuál sería el resultado de la expresión? Responde en el cuestionario: pregunta 2.

  Ejecuta tu programa "vigilando" la dirección `0501h` que corresponde a la variable `a`, para verificar que al finalizar el mismo esa dirección contiene el resultado que has predicho. Vigila también las direcciones de las otras variables para comprobar que no cambian de valor.

- **Ejercicio 7.** Traduce a lenguaje máquina las siguientes instrucciones del lenguaje C. Como en el ejercicio anterior, las variables `a` y `b` están en las direcciones de memoria `0501h` y `0502h` respectivamente.

```
a++;
b-=a;
```

  Si antes de ejecutar el fragmento de código anterior `a` vale 2 y `b` vale 4, ¿cuáles serán sus valores tras la ejecución? Responde en el cuestionario: pregunta 3. Inicializa las direcciones de memoria correspondientes `a` y `b` con los valores 2 y 4 y verifica que al finalizar la ejecución de tu programa ambas contienen los valores predichos.