Práctica 6

El procesador

Material: PC y Visual Studio 2013

Duración: 2 horas

Lugar: Laboratorios de prácticas (Laboratorio de Redes-Hardware)

La herramienta que vamos a utilizar para el desarrollo de las prácticas de C será Microsoft Visual Studio 2013 que permite la creación, compilación y ejecución de programas escritos en lenguaje C/C++ y C#. Se recuerda al alumnado que, para la mejor comprensión y asimilación de los conocimientos y habilidades desarrollados durante la práctica, deberá haberse estudiado previamente el material docente disponible. Durante la práctica se realizarán diferentes actividades que serán objeto de evaluación.

Desarrollo de la práctica:

1. Operaciones con bits

En esta primera parte de la práctica procederemos a realizar y resolver los ejercicios planeados mediante un proyecto de consola y escritos en el lenguaje C/C++, de la misma forma en la que fue resuelta la práctica 1.

Ejercicios:

1) Escribir un programa en C que declare una variable valor de tipo *unsigned char*, escriba la expresión en C que fije a 1 los 2 bits de mayor peso y a 0 los 2 bits de menor peso, e imprima el resultado por pantalla en binario.

2) Escribir un programa en C que declare una variable de tipo short *int* y la inicialice a un valor concreto. Los bits 10, 9, 8 y 7 de dicha variable deberán ser almacenados en una variable de tipo *unsigned char*. El resultado de dicha variable se imprimirá por pantalla en binario.

3) Escribir un programa en C que componga, en una variable de tipo *unsigned char*, un dato de 8 bits, a partir de los 4 bits de menor peso contenidos en las variables de tipo *unsigned short int* n_low y n_high. Los valores de las variables n_low y n_high deberán ser leídos por teclado, y el resultado se mostrará por pantalla en binario.

Nota. Para la realización de los tres ejercicios propuestos en esta práctica deberá implementarse una función llamada *convertir_a_binario()*, que recibirá, mediante un parámetro por valor, una variable de tipo *unsigned char* con el valor a convertir; y devuelva al programa o función principal, mediante la sentencia *return*, un puntero a la dirección de memoria en donde se encuentre almacenada la cadena de caracteres (formada por ceros y unos) resultado de la conversión. No podrán emplearse más parámetros para la implementación de la función que los que han sido indicados.

2. Introducción a la programación en ensamblador.

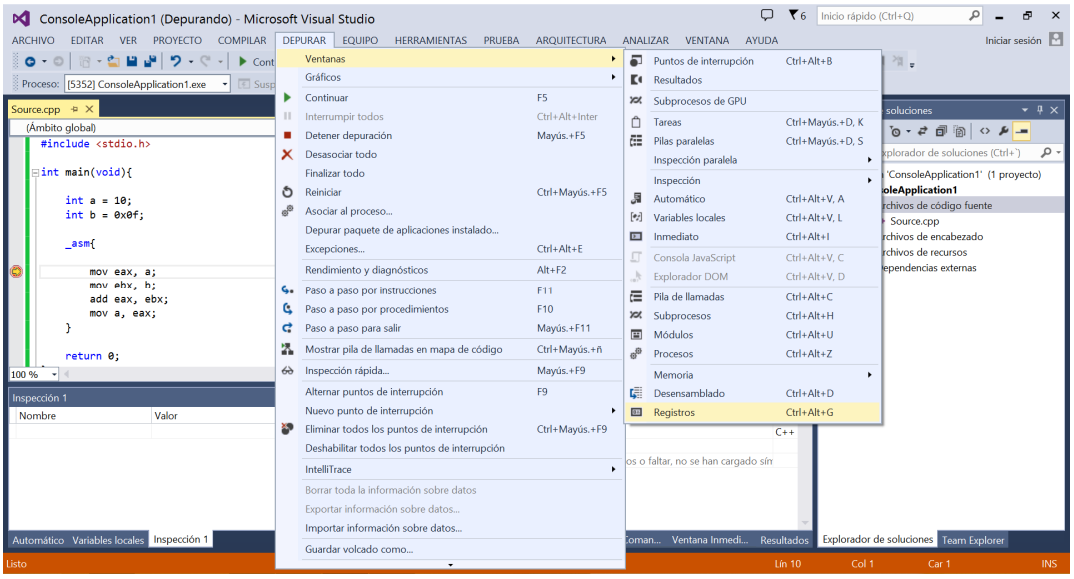
En esta segunda parte de la práctica la declaración e inicialización de variables se hará empleando el lenguaje de programación C. Posteriormente, cuando se pase a desarrollar el código en ensamblador requerido para cada ejercicio, se cargarán los valores de dichas variables en los registros de la máquina según corresponda. Finalmente, la visualización de resultados, si fuera necesario, será realizada empleando nuevamente el lenguaje C, trasladando previamente los resultados obtenidos de los registros de la máquina a variables declaradas en el programa principal.

Nota. Recuerda que puedes emplear las herramientas de depuración del Visual Studio 2013 para depurar el programa; herramientas que incluyen la inspección de los registros de la máquina y de las direcciones de memoria. Dicha opción se encuentra disponible en el Visual Studio cuando se depura el programa. Ejemplo:

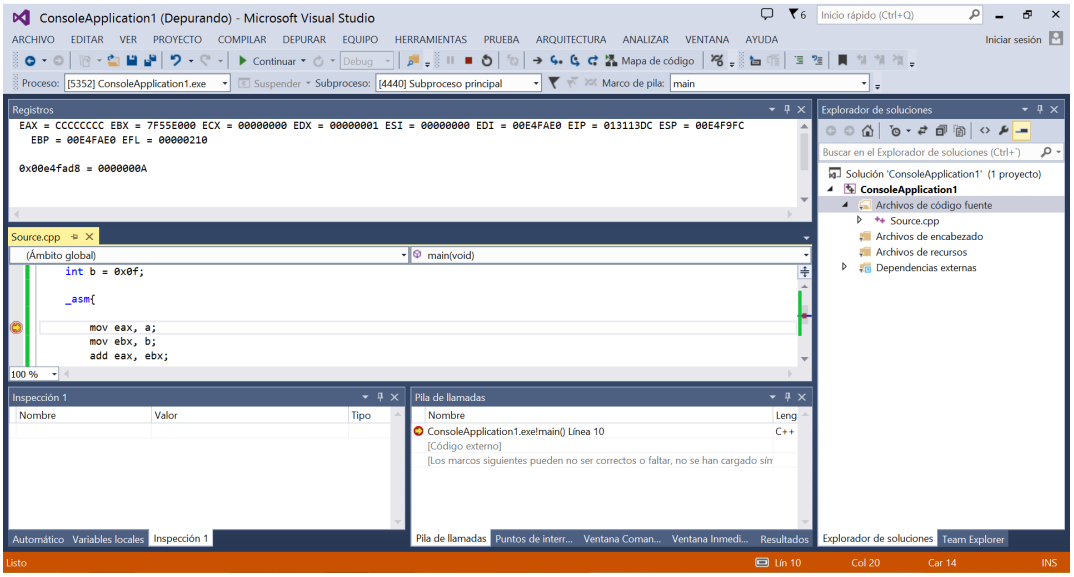
Dado el código:

```
#include <stdio.h>
int main(void){
    int a = 10;
    int b = 0x0f;
    _asm{
       mov eax, a;
       mov ebx, b;
       add eax, ebx;
       mov a, eax;
     }
     return 0;
}
```

Coloca un punto de interrupción en la línea marcada con color amarillo y ejecuta el programa con la opción *Iniciar Depuración*. Una vez que el programa se detenga en el punto de interrupción, iremos al menú Depurar -> Ventanas -> Registros y pulsaremos dicha opción:



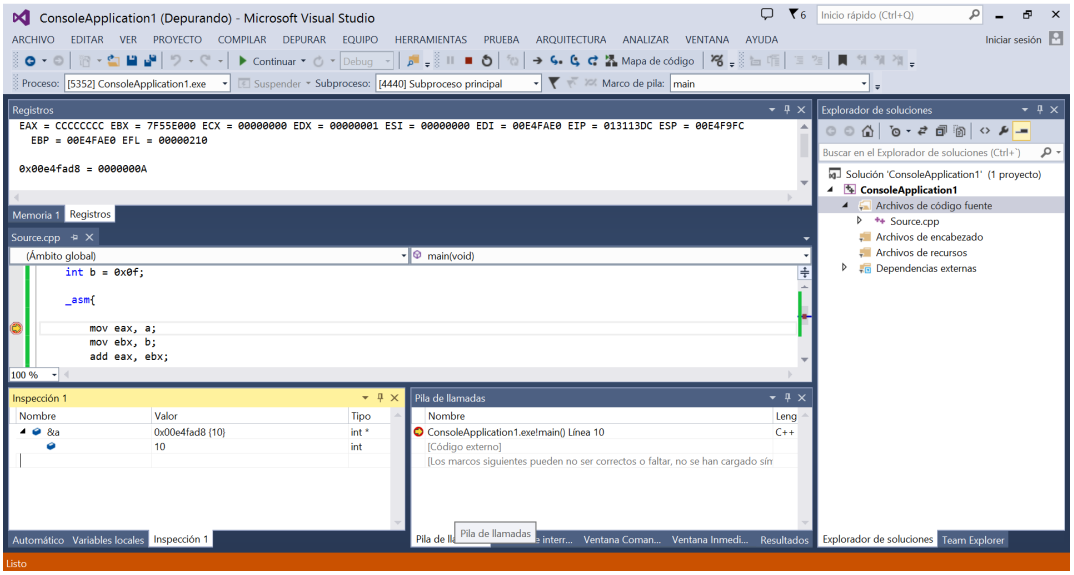
Veremos que se nos abre una nueva ventana con información relativa a los registros de la máquina, que podremos utilizar para comprobar cómo operan nuestros programas sobre la arquitectura interna de la máquina:



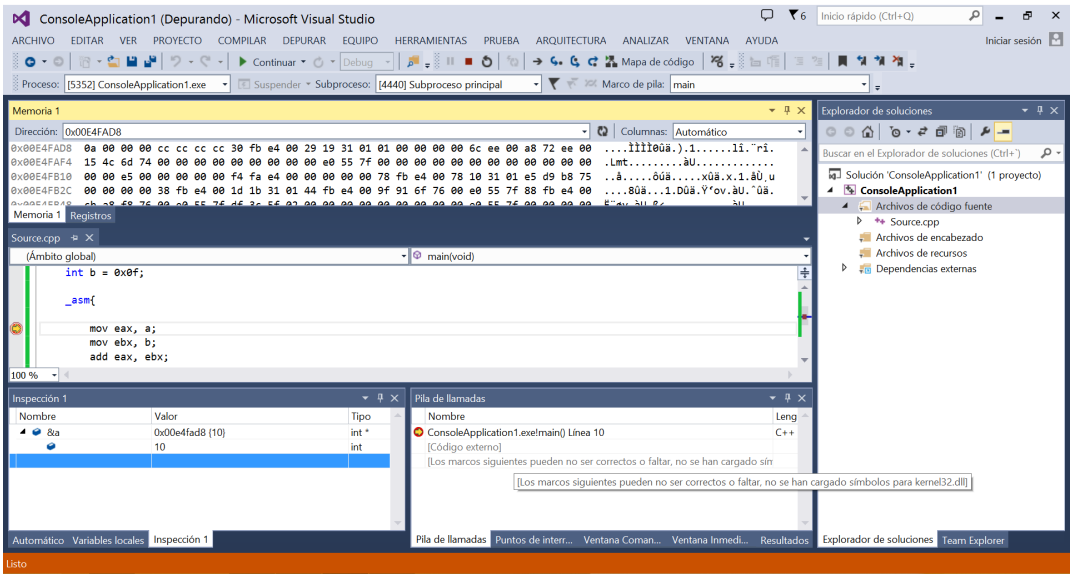
También podemos hacer esto con la memoria, es decir, podemos inspeccionar el contenido de las posiciones de memoria de la máquina. Para ello, seleccionamos Depurar -> Ventanas ->

Memoria 1. En principio, la opción de Memoria 1, al igual que las otras, nos da libertad de explorar la misma. Ejemplo:

Escribe en el inspeccionador de variables del depurador *&a*, para obtener cuál es la dirección de memoria en la que se encuentra almacenada la variable.



Cogeremos dicha dirección y la escribiremos en el campo editable de la pestaña de Memoria 1:



En el ejemplo que se muestra, corresponde a los cuatro bytes contados a partir de la posición 0x00e4fad8, almacenados en formato *little endian*. Puedes probar con otros valores y con variables de distintas longitudes, como un *float* o un *double*.

Ejercicios:

4) Hacer un programa en ensamblador que sume tres números enteros leídos por teclado mostrando el resultado final por pantalla.

5) Escribir un programa que realice el complemento a uno de un número entero leído por teclado. Posteriormente, deberá fijar a 1 el bit de mayor peso. Mostrar ambos resultados por pantalla.

6) Escribir un programa que cuente el número de unos y el número de ceros de un número entero leído por teclado. Mostrar ambos resultados por pantalla.

7) Realizar un programa en ensamblador que inicialice un contador en un registro a 0x06 y se vaya decrementando de uno en uno hasta llegar a 0. Deberá emplearse sentencias de salto en ensamblador para resolver correctamente el problema.

8) Reutilizando el programa en ensamblador realizado en el problema anterior, realiza las operaciones que realicen el decremento del valor del registro contador mediante una subrutina.