Prácticas 7, 8 y 9 de *Estructura de los Computadores*

Eduardo Espuch

Universitat d'Alacant
Universidad de Alicante

Resumen

Documento que reúne los entregables de las prácticas 7, 8 y 9 de la asignatura *Estructura de los Computadores* . El documento tendrá adjunto tres carpetas para los ejercicios entregables.

1. Entregables práctica 7

1. Considerando que en el primer entregable se da el código principal y únicamente debemos hacer el código de la función sum, que pondremos al final ya que este hace incisión en mitad del código principal con una función jal.

   Con esto, sabemos que en $a1 tenemos la longitud del vector de numeros y, ademas, tenemos que en $v0 y en $v1 se obtendrán al finalizar la función el sumatorio de positivos y negativos respectivamente. Para ello, sabemos que debemos de hacer un loop que lea cada posición del vector que se inicia en la dirección de vector (cuya dirección en el código principal se ha guardado en $a0) y que cada elemento del vector esta a 4 bytes de distancia (o una palabra).

   Sabiendo esto, podemos crear un bucle 'loop' en el que tenemos un registro $t1 que sera la dirección dada por $a0 desplazada una cantidad de bytes indicado por $t0, el cual depende del indice o iteracion actual por la cantidad de bytes a desplazarse por iteracion (4 bytes o una palabra). El indice ira incrementando por uno en cada iteracion, estando este guardado en $t3 e iremos guardando la palabra en la dirección de $t1 en el registro $t2, el cual se comprobara si es positivo o negativo y según su valor se le sumara su valor al registro $v0 o al $v1.

   Finalmente, incrementamos el valor del desplazamiento de bytes ($t0) respecto al indice, recordando que son múltiplos de 4 (desplazamiento a la izquierda de 2 bits) y haremos una comprobación para saber si la iteracion o el indice del vector es igual a la longitud de este (dado por $a1), si no lo es hará loop, si lo es devolverá a la función principal, teniendo los sumatorios positivos y negativos en $v0 y $v1.

   Este ejercicio se puede comprobar con el archivo ej1p7.asm en la carpeta p7.

2. Se nos plantea el obtener la suma de la diagonal de una matriz que vamos introduciendo nosotros. El planteamiento para la resolución en este código consiste usar un bucle for anidado dentro de otro for y así ir guardando los elementos (separados por columnas) de cada fila hasta obtener el máximo numero de filas.

   Para desarrollar el código, generalizando, se ha:

a) Modularizado la suma de los elementos de la diagonal cuando los indices de la fila y la columna son iguales. El valor total se mantiene guardado en un valor de paso global y al acabar el doble bucle for se imprimirá.

b) Modularizado el pedir los valores para cada posición.

c) Avanzado en 4 bytes para acceder a las posiciones de cada elemento en una fila. La dirección final era ajena a la dirección base, por lo tanto vamos sumándole múltiplos de 4.

d) Avanzado en 16 bytes desde la dirección base para acceder a la siguiente fila y a la vez reiniciar el indice de elementos de la fila. Una opción hubiera seguido sumando 4 ya que hubiera accedido a la posición siguiente respecto a la dirección final pero se quería demostrar el salto de fila.

e) Considerar que las condiciones de salida para cada bucle for es que el indice alcance el máximo de elementos en una fila o en una columna. Si alcanza el máximo de elementos en una fila, sumara 1 en el indice de columnas y reiniciara el bucle pasando a la siguiente, si alcanza el maximos de filas (elementos en una columna) finaliza el bucle entero.

En el archivo ej2p7.asm se puede ver la solución planteada, acompaña de comentarios que facilitaran interpretar el código escrito.

2. Entregables práctica 8

1. Se nos da el código, solo falta completar la porción que obtiene la longitud ($2 * \pi * r$) y el área ($\pi * r^2$), ademas de emitir por pantalla el resultado (usaremos la función 2 del syscall, teniendo en $f12 el valor en coma flotante que queramos imprimir.

   Ambas formulas exigen obtener $\pi * r$, con lo cual guardamos en un registro, por ejemplo $f2 en este caso, esta operación (en la que usamos la instrucción mul.s). Para la longitud, si hacemos $f2+$f2 (usando add.s), esto también seria decir $f2∗2 y por lo tanto, guardamos este valor en $f12 y lo imprimimos directamente por pantalla.

   Por otro lado, el área consideramos el radio, que por el código ya dado se guarda en $f0, y lo multiplicamos usando otra vez mul.s por $f2, tal que tendríamos como resultado $\pi * r * r = \pi * r^2$. Con el área obtenida, imprimimos por pantalla el mensaje correspondiente para responder el valor del área (se le ha añadido un \n para hacer un salto de linea) y a continuación volver a usar la función 2 del syscall para imprimir el valor en coma flotante.

   Si quisiera que fuese en doble precision, cuando se nos pedían los valores se debería de haber usado la función 7, y no la 6, ya que esta es para coma flotante. Si se decidiese usar este formato, deberíamos cambiar las funciones 6 por 7, 2 por 3 y los mul.s y add.s por mul.d y add.d.

2. Matemáticamente hablando, se nos pide $S = \sum_{i=1}^{n} v_i$, con $v_i \in$array y ademas obtener la media, que seria $\overline{M} = \frac{S}{n}$. Para ello, debemos leer en bucle, aumentando el indice de desplazamiento en la memoria e ir convirtiendo la palabra a coma flotante para sumarle este valor a un registro que hará de S. Cuando haya leído todo los elementos, la longitud la pasamos a coma flotante, obteniendo n, y hacemos una división para obtener $\overline{M}$.

   Se reserva una posición de memoria para suma, con lo cual intuimos que es para guardar el valor de S ahí, con lo cual pasamos de coma flotante a palabra y lo introducimos. Por otro lado, se nos pide imprimir el resultado por consola, para ellos pasamos por orden los valores al registro $f12 y usamos la función 2 del syscall para imprimir coma flotante.

3. Entregables práctica 9

1. Es exactamente como el ejercicio 2 de la practica 8, únicamente varia en el hecho de no tener que convertir de palabra (complemento a la base) a IE^3 cada elemento del array salvo la longitud a la hora de obtener la media. Este ejercicio exigía conocer como trabajar de forma directa sobre

el coprocesador, usando comando como lwc1 o swc1, que actuaria como el lw y sw solo que en sobre el coprocesador.

2. Se nos da gran parte del codigo, únicamente se espera que hagamos la multiplicación sucesiva para obtener la n-ésima potencia de un valor dado al inicio. Es tan sencillo como un bucle for y una instrucción de multiplicación con el producto y el valor inicial n veces.

3. Se no vuelve a dar el código prácticamente hecho, únicamente debemos comprobar cual de los valores introducidos es el mayor. Para ello usaremos en la función max un condición que opera con valores dados en coma flotante. Estas instrucciones son, para nuestro código en particular, c.lt.s y bc1f. Cabe decir que no existe el el comprobar si un elemento es mayor en el coprocesador, pero se puede obtener si comprobamos si es menor y, en el caso de no serlo, sabremos que es mayor o igual, siendo esta la intención del ejercicio. Veamoslo con mas detalles:

   a) c.lt.s compara si el primer registro es menor que el segundo y, dado en este formato, pondrá el flag predeterminado (el flag 0) en true o false según si se cumple dicha condición. Se puede dejar indicado si se quiere otro flag solo con ponerlo justo a continuación. Si el flag 0 es true, entonces $X < Y$ e Y es el máximo, si por el contrario, es false, sabremos que $X \geq Y$ con lo cual podremos afirmar que X sera el máximo, o igual a Y con lo cual dará igual si escogemos como X el máximo.

   b) bc1f comprobara el valor del flag indicado, en este caso al no venir ninguno indicado comprobara el predeterminado (el flag 0) y si es false, saltara a la etiqueta indicada. Recordad que si salta significa que $X \geq Y$ y si no lo hace y sigue entonces $X < Y$.

   Finalizamos la función pasando el valor adecuado al registro $f0 que sera el parámetro de salida, obteniendo en este el máximo.