Practica de Laboratorio 1 de Jesús Rachadell 31421494

1. ¿Como se implementa la recursividad en MIPS32? ¿Qué papel cumple la pila ($sp)?

Por la naturaleza de la recursividad es necesario guardar los valores actuales y recuperarlos una vez vuelto de la función, como en MIPS32 tenemos un numero limitado de registros usamos una pila, en este caso se usa ($sp), que funciona al "apartar" un espacio para guardar los parámetros de la función actual, así una vez realizada la llamada recursiva se pueden recuperar los datos de la función actual. Lo primero es restar 4*n (n items) que vayamos a guardar a $sp usando sw, luego una vez recuperados los datos usando lw se suma 4*n a $sp así "barrer" la memoria y volviendo a su posicion anterior a la entrada en la función.

2. ¿Que riesgos de desbordamiento existen? ¿Cómo mitigarlos?

Para que exista un problema de desbordamiento hay que considerar el caso base, ya que es la medida que se tiene para evitar que una función recursiva no se extienda al infinito, para ello se trabaja la recursividad alrededor del caso base, en MIPS32 también es importante ir "barriendo" la memoria a medida que se deja de utilizar, ya que el programa a diferencia de un lenguaje de alto nivel, no lo hace por sí solo

3. ¿Qué diferencias encontraste entre una implementación iterativa y una recursiva en cuanto al uso de memoria y registros?

La principal diferencia es que la llamada iterativa no requiere guardar y liberar memoria, por ende no existe ese riesgo de desbordamiento que se puede encontrar en las recursivas. Además de eso, el uso de los registros se mantiene relativamente igual, pero eso está más atado a la naturaleza del ejercicio y no tanto al método que se use para realizarlo, en el caso de está practica el número de registros utilizados se mantiene igual y la gran diferencia recae en que para la implementación recursiva se necesita usar $sp.

4. ¿Qué diferencias encontraste entre los ejemplos académicos del libro y un ejercicio completo y operativo en MIPS 32?

Creo que la primera diferencia notable es el uso y no uso de las pseudo-instrucciones, en el libro intentan en la medida de lo posible no usar pseudo-instrucciones y solo usar sus derivadas, a parte de eso no logro encontrar mucha diferencia entre los ejemplos y los ejercicios realizados en MIPS32.

5. Ejecución paso a paso en MARS

Para el ejercicio iterativo lo primero que se hace es recibir el dato del terminal puesto por el usuario y se carga en $a0, luego se inicializa i en 0 y se guarda su valor en $t0, por ultimo se inicializa $a1 en 1 positivo, ya en el ciclo primero comprueba que n sea diferente a i, luego multiplica por -1 a $a1, esto lo hace para luego hacer la comprobación de si es par o no, por ultimo se aumenta a i en 1, y esto se repite hasta que se cumpla la condición inicial. luego vuelve a main con el valor en $v0 y hace la comprobación, sí es 1 es par si es -1 es impar, el programa finaliza imprimiendo el resultado.

Para el ejercicio recursivo lo primero es recibir el dato del terminal y cargarlo en $a0, se procede entonces a la función, ahí lo primero que se hace es guardar $ra actual dentro de $sp, pasa a comprobar si se cumple el caso base $n = 0$, si se cumple, retorna 0 sí no, continua primero modificando $a0 a ser $n - 1$ y llama de nuevo a la función ahora con un nuevo valor de n, una vez conseguido el caso base y vuelto de la funcion se procede a ser una operación $1 - par(n - 1)$ y retornar el valor resultante. Una vez terminado la función vuelve a main y se imprime el resultado.

6. Justificar la elección del enfoque(iterativo o recursivo) según la eficiencia y claridad en MIPS32

En el ejercicio iterativo utilice el 1 o -1 para hacer más sencilla la comparación, además que no utiliza más registros de los necesarios, y creo que el concepto de flip-flop es más sencillo de entender mirando el código que otras soluciones que pensé.

En el ejercicio recursivo opté por una solución estándar ya que la complejidad del ejercicio y el manejo de los casos no era difícil, opte por un diseño claro y directo al punto, utilizando el 0 y 1 para determinar el resultado. Opte solo por guardar el $ra ya que es lo único que cambia y se necesita restaurado una vez vuelto de la función.

7. Análisis y discusión de los resultados

Ambas implementaciones funcionan y cumplen con lo que deben hacer, pero al medir la eficiencia entre ambos es claro cual es mejor para la función que debe cumplir, la implementación iterativa tiene la ventaja de no necesitar guardar grandes cantidades de memoria lo que lo hace más viable cuando el n aumenta.