Rafael Fernández Fleites

Ejercicio 5.4

A. En la versión menos optimizada, el registro %xmm0 se utiliza como un valor temporal en cada iteración del bucle, mientras que en la versión optimizada, se utiliza como un acumulador que guarda el resultado parcial en cada paso, lo que permite evitar lecturas y escrituras innecesarias.

B. Las dos versiones de "combine3" tienen la misma funcionalidad, incluso si hay solapamiento de memoria. La optimización realizada no afectará el comportamiento esperado del código.

C. Esta transformación puede llevarse a cabo sin alterar cómo funciona el programa. Esto se debe a que, exceptuando la primera iteración, el valor leído desde dest al inicio de cada iteración es el mismo valor que fue escrito en ese registro al finalizar la iteración anterior. Por lo tanto, la instrucción de combinación puede simplemente utilizar el valor que ya se encuentra en el registro %xmm0 al inicio del bucle. La optimización no producirá resultados diferentes en casos donde la ubicación de destino (dest) y la ubicación de algunos de los datos del vector sean iguales.

Ejercicio 5.5

1. A. El código proporcionado realiza dos multiplicaciones y una suma en cada iteración del bucle. Por lo tanto, el número total de adiciones es n, y el número total de multiplicaciones es 2n.

B. El cálculo que limita el rendimiento en este caso es la repetición del cálculo de la expresión xpwr = x \* xpwr, lo cual implica una multiplicación de punto flotante (5 ciclos de reloj) y el cálculo de una iteración no puede comenzar hasta que el cálculo de la iteración anterior haya finalizado. La actualización del resultado solo requiere una suma de punto flotante (3 ciclos de reloj) entre iteraciones sucesivas.

Ejercicio 7.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Symbol | .symtab entry? | Symbol type | Module where defined | Section |
| buf | Sí | extern | m.o | .data |
| bufp0 | Sí | global | swap.o | .data |
| bufp1 | Sí | global | swap.o | COMMON |
| swap | Sí | global | swap.o | .text |
| temp | No | - | - | - |

Los símbolos buf, bufp0, bufp1 y swap son entradas de la tabla de símbolos.

El símbolo temp no tiene entrada en la tabla de símbolos ya que constituye una variable local.

El símbolo buf está declarado como externo mientras que bufp0, bufp1 y swap son globales.

Las variables globales y estáticas inicializadas pertenecen a .data, por lo que buf y bufp0 (variables globales inicializadas) pertenecen a esta sección.

El símbolo bufp1 pertenece a COMMON por ser una variable global no inicializada.

La función swap pertenece a la sección .text porque es una función.

Ejercicio 7.2

A. El enlazador escoge el símbolo fuerte definido en el módulo 1, sobre el símbolo débil definido en el módulo 2.

(a) REF(main.1) → DEF( main.1)

(b) REF(main.2) → DEF( main.1)

B. Error, porque cada módulo define un símbolo fuerte principal y la regla 1 plantea que no se permiten múltiples símbolos fuertes con el mismo nombre.

C. El enlazador escoge el símbolo fuerte definido en el módulo 2, sobre el símbolo débil definido en el módulo 1.

(a) REF(x.1) → DEF(x . 2)

(b) REF(x.2) → DEF( x . 2)

Ejercicio 7.3

A. En este escenario, solo se tiene una dependencia directa, donde "p.o" depende de "libx.a". La solución más simple sería ejecutar: gcc p.o libx.a.

B. En este caso el archivo de objeto p.o depende de la biblioteca estática libx.a, y también depende de la biblioteca estática liby.a. Por lo tanto, para enlazar correctamente el programa, se necesita proporcionar al enlazador el archivo de objeto p.o, la biblioteca estática libx.a y la biblioteca estática liby.a; de la siguiente manera: gcc p.o libx.a liby.a.

C. En este caso, se presenta que libx.a depende de liby.a y, a su vez, liby.a depende de libx.a.. Por lo tanto, necesitamos proporcionar al enlazador el archivo de objeto p.o, la biblioteca estática libx.a, la biblioteca estática liby.a y la biblioteca estática libx.a nuevamente para enlazar correctamente el programa, quedando con la siguiente forma: gcc p.o libx.a liby.a libx.a.