Universidad de Buenos Aires - FIUBA
66.20 Organización de Computadoras
Trabajo práctico 1: Programación MIPS
1^{er} cuatrimestre de 2020

`$Date: 2020/05/11 01:29:44 $`

1. Objetivos

Familiarizarse con el conjunto de instrucciones MIPS y el concepto de ABI, extendiendo un programa que resuelva el problema descripto a continuación.

2. Alcance

Este trabajo práctico es de elaboración grupal, evaluación individual, y de carácter obligatorio para todos alumnos del curso.

3. Requisitos

El trabajo deberá ser entregado personalmente, en la fecha estipulada, con una carátula que contenga los datos completos de todos los integrantes, un informe impreso de acuerdo con lo que mencionaremos en la sección 6, y con una copia digital de los archivos fuente necesarios para compilar el trabajo.

4. Descripción

El programa a desarrollar deberá procesar un *stream* de vectores de números enteros. A medida que el programa avance en la lectura de éstos, deberá ordenar cada vector en forma creciente, e imprimir inmediatamente el resultado por el *stream* de salida.

Los vectores ingresarán como texto por entrada estándar (`stdin`), donde cáda línea describe completamente el contenido del mismo, según el siguiente formato:

v_1 v_2 ... v_N

El fin de línea es el caracter \n (*newline*). Debido a que cada línea contiene exactamente un único vector, el fin del mismo coincidirá siempre con el final de la línea que lo contiene. A su vez, cada entero del vector estará separado de otros elementos por uno o más caracteres de espacio en blanco.

Por ejemplo, dado el siguiente flujo de entrada:

```
$ cat input.txt
3 2 1
6 5 1 2 9 3 8 7 4   9
6  0  0 1 3
-1
```

Al ejecutar el programa la salida sería:

```
$ tp1 -i input.txt -o -
1 2 3
1 2 3 4 5 6 7 8 9
0 0 1 3 6
-1
```

Ante un error, el progama deberá detenerse informando la situación inmediatamente (por `stderr`).

4.1. Ejemplos

Primero, usamos la opción `-h` para ver el mensaje de ayuda:

```
$ tp1 -h
Usage:
  tp1 -h
  tp1 -V
  tp1 -i in_file -o out_file
Options:
  -V, --version     Print version and quit.
  -h, --help        Print this information and quit.
  -i, --input       Specify input stream/file, "-" for stdin.
  -o, --output      Specify output stream/file, "-" for stdout.
Examples:
   tp1 < in.txt > out.txt
   cat in.txt | tp1 -i - > out.txt
```

A continuación, ejecutamos algunas pruebas:

```
$ cat example.txt
1
-1       +1


$ cat example.txt | ./tp1
1
-1 1
```

5. Implementación

El programa a desarrollar constará de una mezcla entre código MIPS32 y C, siendo la parte escrita en *assembly* la encargada de ordenar un vector de enteros pasado por parámetro. El formato de dicha función será:

```
void merge_sort(int *vec, size_t len);
```

Asimismo deberá usarse el algoritmo merge sort [1] y el modo 1 del sistema operativo para manejo de acceso no alineado a memoria [3].

En cuanto al manejo de memoria dinámica realizado por `merge_sort()`, deberá realizarse íntegramente en MIPS usando la implementación de referencia `mymalloc/myfree` disponible en [5].

6. Informe

El informe deberá incluir:

- Documentación relevante al diseño e implementación del programa;
- Comando(s) para compilar el programa;
- Las corridas de prueba, con los comentarios pertinentes;
- El código fuente, en lenguaje C y MIPS;
- El código MIPS32 generado por el compilador[1];
- Este enunciado.

7. Entrega de TPs

La entrega de este trabajo deberá realizarse usando el campus virtual de la materia [4]. Asimismo, en todos los casos, estas presentaciones deberán ser realizadas durante los días martes. El *feedback* estará disponible de un martes hacia el otro, como ocurre durante la modalidad presencial de cursada.

Por otro lado, la última fecha de entrega y presentación para esta trabajo será el martes 26/5.

Referencias

[1] Merge sort, `https://en.wikipedia.org/wiki/Merge_sort`.

[2] Bubble sort, `https://en.wikipedia.org/wiki/Bubble_sort`.

[3] Controlling the kernel unalignment handling via debugfs, https://www.linux-mips.org/wiki/Alignment.

[1] Por motivos prácticos, en la copia impresa sólo es necesario incluir la primera página del código assembly MIPS32 generado por el compilador.

[4] Aula Virtual - Organización de Computadoras 86.37/66.20 - Curso 1 - Turno Martes. https://campus.fi.uba.ar/course/view.php?id=649

[5] https://drive.google.com/open?id=1RZNflRb6sG8nqsUAVxS2Ch1pNnEJCxxi