Informe TP1

Cotarelo Rodrigo, *Padrón Nro. 98577*
cotarelorodrigo@gmail.com

Etchegaray Rodrigo, *Padrón Nro. 96856*
rorroeche@gmail.com

Longo Nicolás, *Padrón Nro. 98271*
longo.gnr@hotmail.com

1er. Cuatrimestre de 2019
66.20 Organización de Computadoras – Práctica Martes
Facultad de Ingeniería, Universidad de Buenos Aires

7/5/19

1. Enunciado

1. Objetivos
   Familiarizarse con el conjunto de instrucciones MIPS y el concepto de ABI, extendiendo un programa que resuelva el problema descripto en la sección 4.

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

3. Requisitos
   El informe deberá ser entregado personalmente, por escrito, en la fecha estipulada, con una carátula que contenga los datos completos de todos los integrantes. Además, es necesario que el trabajo práctico incluya (entre otras cosas, ver sección 6), la presentación de los resultados obtenidos, explicando, cuando corresponda, con fundamentos reales, las causas o razones de cada caso.

4. Descripción
   En este trabajo, vamos a implementar dos programas en assembly MIPS, que nos permitan convertir información almacenada en streams de texto, de acuerdo a lo descripto en el trabajo anterior [1]: unix2dos, para transformar archivos de texto de UNIX a DOS; y dos2unix para realizar la operación inversa. Es decir, siguiendo la línea de trabajo de los prácticos anteriores, buscamos generar dos binarios diferentes en vez de un único programa que resuelva ambos problemas. Además, por tratarse de un TP orientado a practicar el conjunto de instrucciones, ambos programas deberán escritos completamente en assembly, en forma manual.

5. Interfaz
   Por defecto, ambos programas operan exclusivamente por stdin y stdout. Al igual que en el trabajo anterior, al finalizar, estos programas retornarán un valor nulo en caso de no detectar ningún problema; y, en caso contrario, finalizarán con código no nulo (por ejemplo 1).

6. Ejemplos
   Cuando el documento de entrada es vacío, la salida debería serlo también:

```
$ unix2dos </dev/null | wc -c
0
$ dos2unix </dev/null | wc -c
0
```

También se aplican todos los casos de prueba definidos para el TP anterior. Por ejemplo:

```
$ od -t c /tmp/dos.txt
0000000 U n o \r \n D o s \r \n T r e s \r \n
0000020
$ dos2unix -i /tmp/dos.txt -o - | od -t c
0000000 U n o \n D o s \n T r e s \n
0000015
```

Por último, podemos usar el programa que sigue para generar secuencias de datos aleatorias, pasarlas a hexa, reconvertirlas al dominio original, y verificar que la operación no haya alterado la información:

```
$ n=0; while :; do
> n="`expr $n + 1`";
> head -c 100 </dev/urandom >/tmp/test.$n.u;
> unix2dos </tmp/test.$n.u >/tmp/test.$n.d || break;
> dos2unix </tmp/test.$n.d >/tmp/test.$n.2.u || break;
> diff -q /tmp/test.$n.u /tmp/test.$n.2.u || break;
> rm -f /tmp/test.$n.*;
> echo Ok: $n;
> done; echo Error: $n
Ok: 1
Ok: 2
Ok: 3
...
```

7. Implementación
   El programa a implementar deberá satisfacer algunos requerimientos mínimos, que detallamos a continuación:

   a) ABI
   Será necesario que el código presentado utilice la ABI explicada en la clase: los argumentos correspondientes a los registros $a0-a3$ serán almacenados por el callee, siempre, en los 16 bytes dedicados de la sección "function call argument area" [2].

   b) Casos de prueba
   Es necesario que la implementación propuesta pase todos los casos incluidos tanto en este enunciado como en el conjunto pruebas suministrado en el informe del trabajo, los cuales deberán estar debidamente documentados y justificados.

   c) Documentación
   El informe deberá incluir una descripción detallada de las técnicas

y procesos de medición empleados, y de todos los pasos involucrados en el desarrollo del mismo, ya que forman parte de los objetivos principales del trabajo.

8. Informe
   El informe deberá incluir:
   - Este enunciado;
   - Documentación relevante al disenio, implementación y medición de las características del programa;
   - Las corridas de prueba, (sección 5.2) con los comentarios pertinentes;
   - El código fuente completo, en dos formatos: digitalizado1 e impreso en papel.
9. Fechas
   Fecha de vencimiento: 7/5.
10. Referencias Enunciado del primer trabajo práctico, primer cuatrimestre de 2019. http://groups.yahoo.com/groups/orga-comp/.

2. Implementación

Se implementaron dos funciones para poder convertir un archivo de DOS a UNIX y viceversa. La implementación se basó en un ciclo while que recorre el archivo byte por byte mientras consulta si está parado sobre un '\r' o '\n'. En caso de ser alguno de estos caracteres, se efectua la modificación correspondiente y sino, simplemente se escribe el mismo caracter en la salida. Para lograr esto, decidimos dividir los programas en pequeñas funciones.Esto nos permitió por un lado reducir la complejidad e ir teniendo seguridad de que las cosas que desarrollábamos funcionaban de la manera correcta. Y por otro lado, reutilizar funciones que aplicaban tanto a unix2dos como a dos2unix. Las funciones desarroladas fueron:

- getch: devuelve el caracter leído por stdin y en caso de ser EOF devuelve -1.
- putch: escribe el caracter que recibe por stdout.
- isLF: devuelve 1 si el caracter es LF y 0 si no lo es.
- isCR: devuelve 1 si el caracter es CR y 0 si no lo es.

3. Utilización

4. Generación de binarios

5. Pruebas

6. Código fuente

6.1. unix2dos

6.2. dos2unix

7. Conclusiones