## TP0

# Organización del cumputador

Ramiro Sánchez — Lukas De Angelis Riva — Matías Merlo October 22, 2020

### 1 Introducción

El objetivo de este proyecto es implementar un codificador y un decodificador de Base64 para una arquitectura MIPS32. Utilizaremos el emulador QEMU

### 2 Detalles de diseño

Tanto el decodificador como el codificador están diseñados para ser ejecutados en una computador que implemente Big Endian.

La codificación se realiza sobre un conjunto de bloques de 3 bytes (3 caracteres) mediante sucesivos desplazamientos de 2 bits a derecha para obtener 4 palabras de 6 bits. Luego concatenando la conversión de cada bloque de 3 bytes a 4 bytes en un arreglo más grande ya convertidos a base 64.

La decodificación también se realiza mediante desplazamientos sobre un bloque de 4 bytes para convertirlo en bloque de 3bytes en representación 8bits.

#### 3 Recomendaciones de uso

Nomenclatura utilizada: input format - output format. Donde input-format toma los valores:

- estándar: Hace referencia a la entrada estándar.
- archivo: Hace referencia a la entrada mediante un archivo.

Y donde output-format toma los valores:

- estándar: Hace referencia a la salida estándar.
- $\bullet\,$ archivo: Hace referencia a la salida por medio de un archivo.

Para la decodificación y codificación estándar-estándar y estándar-archivo recomendamos utilizar los programas 'echo' y 'cat'. Con la pequeña salvedad de que debe tener en cuenta que el programa 'echo' inserta el carácter \n al final

del texto a decodificar/codificar, por lo que, en caso de la codificación se estará codificando un texto alterado al esperado. Luego, en caso de la decodificación, este no debería suceder, el carácter \n no pertenece a la tabla de codificación, por lo que: El programa se encarga de sacar este último \n de la entrada en caso de que exista. Pero, también como recomendación es que lo tenga en cuenta al utilizar el programa, por lo que, puede utilizar el programa 'echo' con el tag -n para evitar cualquier tipo de malentendido.

#### 4 Pruebas

Se corrieron las siguientes pruebas:

• prueba 1: Se codificó la palabra "Man"

```
> echo -n "Man" | ./TPO
> TWFu
```

Coincidiendo con el resultado esperado.

• prueba 2: Se codificó el archivo "quijote.txt"

Coincidiendo con lo esperado.

• prueba 3: Se codificó la palabra "M"

```
> echo -n "M" | ./TPO
> TQ==
```

Coincide con lo esperado, completó con los "=" necesarios.

• prueba 4: Se codificó la palabra "OS"

```
> echo -n "OS" | ./TPO
> T1M=
```

Nuevamente se completó con el "=" necesario.

• prueba 5: Se decodificó la palabra "TWFu"

```
> echo -n "TWFu" | ./TPO -d
> Man
```

Cuyo resultado es correcto.

• prueba 6: Se decodificó la palabra "TWFuQ2F0"

```
> echo -n "TWFuQ2F0" | ./TP0 --d
> ManCat
```

```
• > echo -n "TQ==" | ./TPO -d > M
```

Por lo que decodifica correctamente a los símbolos "="

- prueba 7: Se codificó el archivo "preambulo.txt" en "preambulo64.txt"
  - > cat preambulo.txt
  - > Nos los representantes del pueblo de la Nacion Argentina,
  - → reunidos en Congreso General Constituyente por voluntad

  - $\ensuremath{\scriptscriptstyle{\hookrightarrow}}$  cumplimiento de pactos preexistentes, con el objeto de
  - $\scriptscriptstyle
    ightarrow$  constituir la union nacional, afianzar la justicia,
  - $\scriptscriptstyle
    ightarrow$  consolidar la paz interior, proveer a la defensa comun,
  - $_{\mathrel{\mathrel{\hookrightarrow}}}$  promover el bienestar general, y asegurar los

  - → posteridad, y para todos los hombres del mundo que

  - $_{\mathrel{\mathrel{\rightarrow}}}$  ordenamos, decretamos y establecemos esta Constitucion,
  - → para la Nacion Argentina.
  - > ./TPO -i preambulo.txt -o preambulo64.txt
  - > cat preambulo64.txt

```
> Tm9zIGxvcyByZXByZXNlbnRhbnRlcyBkZWwgcHVlYmxvIGRlIGxhIE5hY
   2lvbiBBcmdlbnRpbmEsIHJldW5pZG9zIGVuIENvbmdyZXNvIEdlbmVy
   YWwgQ29uc3RpdHV5ZW50ZSBwb3Igdm9sdW50YWQgeSBlbGVjY2lvbiB
   kZSBsYXMgcHJvdmluY2lhcyBxdWUgbGEgY29tcG9uZW4sIGVuIGN1bX
   BsaW1pZW50byBkZSBwYWN0b3MgcHJ1ZXhpc3RlbnRlcywgY29uIGVsI
   G9iamV0byBkZSBjb25zdG10dWlyIGxhIHVuaW9uIG5hY2lvbmFsLCBh
    ZmlhbnphciBsYSBqdXNOaWNpYSwgY29uc29saWRhciBsYSBwYXogaW5
    OZXJpb3IsIHByb3Z1ZXIgYSBsYSBkZWZ1bnNhIGNvbXVuLCBwcm9tb3
   ZlciBlbCBiaWVuZXNOYXIgZ2VuZXJhbCwgeSBhc2VndXJhciBsb3MgY
   mVuZWZpY2lvcyBkZSBsYSBsaWJlcnRhZCwgcGFyYSBub3NvdHJvcywg
   cGFyYSBudWVzdHJhIHBvc3RlcmlkYWQsIHkgcGFyYSB0b2RvcyBsb3M
    gaG9tYnJlcyBkZWwgbXVuZG8gcXVlIHF1aWVyYW4gaGFiaXRhcmVuIG
   VsIHN1ZWxvIGFyZ2VudGlubzogaW52b2NhbmRvIGxhIHByb3R1Y2Npb
    24gZGUgRGlvcywgZnVlbnRlIGRlIHRvZGEgcmF6b24geSBqdXNOaWNp
   YTogb3JkZW5hbW9zLCBkZWNyZXRhbW9zIHkgZXNOYWJsZWN1bW9zIGV
    zdGEgQ29uc3RpdHVjaW9uLCBwYXJhIGxhIE5hY21vbiBBcmdlbnRpbm |
   EuCg==
```

Cuyo resultado coincide con el devuelto por el programa base64 de Linux.

Otras pruebas cuyos resultados no estarán en el informe por legibilidad:
 Codificamos el archivo TP0.c con el comando ./TP0 -i TP0.c -o TP064.txt
 y luego lo volvimos a decodificar con el comando ./TP0 -d TP064.txt -o
 TP0\_vuelta.txt corroboran que este último tiene lo mismo que TP0.c
 Repetimos todas las pruebas anteriores cambiando las formas de entrada
 y de salida para contemplar todos los casos posibles.

Como extra, también fue entregado un archivo: TP0\_test.c donde hay algunas pruebas sobre el uso básico del TP.

## 5 Archivos entregados

Los archivos .c .h y .s entregados se encuentran en el repositorio de github, cuyo link está en la secciónde Links. Los archivos con código entregados son:

| • TP0.c                   | $\bullet$ makefile |
|---------------------------|--------------------|
| • TP0.s                   | • mystrlen.S       |
| $\bullet$ decode_encode.c |                    |
| • decode_encode.h         | • TP0_test.c       |

Para compilar el programa:

make tp0

Para ejecutar las pruebas:

make test

También fueron entregados 2 archivos .txt para testear.

- $\bullet$  preambulo.txt
- quijote.txt

## 6 Links

Link al repositorio de Github:

 $\verb|https://github.com/RamiroSanchez-dev/Organizacion-Del-Computador|\\$