Nro. ord.	Apellido y nombre	L.U.	#hojas

SISTEMAS DIGITALES - **Parcial**

Primer Cuatrimestre 2024

Ej. 1	Ej. 2	Ej. 3	Ej. 4	Nota

Correctorx:

Aclaraciones

- Anote apellido, nombre, LU y numere *todas* las hojas entregadas, entregando los distintos ejercicios en hojas separadas.
- Cada ejercicio será calificado con una de las siguientes tres notas: Bien, Regular o Mal. La división de los ejercicios en incisos es meramente orientativa. Los ejercicios se calificarán globalmente.
- El parcial **no es a libro abierto** pero pueden utilizar la cartilla de referencia entregada por la materia.
- **Importante:** Justifique sus respuestas.
- Un resultado sin suficiente justificación equivale a un ejercicio no resuelto.
- El parcial se aprueba con al menos dos ejercicios Bien y uno Regular. Para obtener un Regular es necesario demostrar conocimientos sobre el tema del ejercicio. Para la promoción deben contar con al menos tres ejercicios bien y uno regular.

Ejercicio 1 Se cuenta con cuatro datos sin signo de un byte cada uno almacenados en el registro `s0` y queremos sumar el valor de los cuatro datos.. Escriba un programa de ensamblador `RISC V` que realice esta operación y almacene el resultado en el registro `a0`.

Ejemplo:

Bits	31 24	23 16	15 8	7 0
s0	0x90	0x1A	0x00	0x02

Con este dato el registro debería valer 0x000000BC.

Ejercicio 2 Implemente la función hanoi en el lenguaje ensamblador `RISC V` de forma recursiva, respete la convención de llamada presentada en la materia, explique el uso que le dará a cada registro y cómo se asegura que sus valores se preservan antes y después de cada llamada a función.

$$hanoi(n) = \begin{cases} 1, & \text{si } n = 1 \\ 2 * hanoi(n-1) + 1, & \text{si } n > 1 \end{cases}$$

Guía de resolución (opcional):

- Escriba una versión de pseudocódigo.
- Transforme cada caso a su equivalente de operaciones atómicas (descomponga las operaciones lógicas, aritméticas y llamadas a función).
- Identifique los registros a emplear para cada dato.
- Si debe preservar algún registro para respetar la convención, indique qué mecanismo utilizará.
- Defina un flujo de ejecución tentativo.

Ejercicio 3 Un sistema de gestión de notas mantiene registro de las notas de una clase en un arreglo de datos de 1 byte sin signo. Queremos agregar lógica para determinar la cantidad de estudiantes que obtuvieron un valor mayor a 0xA0, que es el valor con el cual promocionan.

Se cuenta con un arreglo `notas` de datos de 8 bits **sin signo** empaquetados de forma contigua. El largo del arreglo (en bytes) se define en la constante `largo`.

Escriba un programa que cuente la cantidad de notas que se encuentran por sobre el valor `0xA0`. Si la cantidad de valores que superan este límite es mayor a la mitad del largo debemos poner un 1 en el registro `a0`, en caso contrario debemos poner un 0.

Ejemplo:

Dirección	0x00000000	0x00000001	0x00000002	0x00000003
notas	0xA1	0xB0	0xF1	0x07

En este caso debemos poner un 1 en a0 porque más de la mitad de los valores valen más que 0xA0.

Esqueleto de programa:

```
1 .data:
2 mediciones: .byte 0xA1 0xB0 0xF1 0x07
3 largo: .byte 4
4
5 .text:
6 # Escribir el programa aca.
```

Ejercicio 4 ¿Cuál es la ventaja de tener un registro de valor constante 0? ¿Cómo maneja las escrituras a este registro?

B-

① Ejercicio 1:

```
ADDI  t0, a0, 0
ADDI  t1, a0, 0
ADDI  t2, a0, 0
ADDI  t3, a0, 0
```

} Cada uno de t0, t1, t2, t3 es igual al valor de a0

t0 es igual para el byte 0 luego se limpia
t1 " " " " 1 " " "
t2 " " " " 2 " " "
t3 " " " " 3 " " "

// Limpio t0 para quedarme con el byte 0

```
SLLI  t0, t0, 24
SLRI  t0, t0, 24
```

// Limpio t1 para quedarme con byte 1

```
SLLI  t1, t1, 16
SLRI  t1, t1, 24
```

// Limpio t2 para quedarme con byte 2

```
SLLI  t2, t2, 8
SLRI  t2, t2, 24
```

// Limpio t3 para quedarme con byte 3

```
SRLI  t3, t3, 24
```

// Los sumo en a0

```
ADD a0, t0, t1
ADD a0, a0, t2
ADD a0, a0, t3
```

RET

RET es una pseudo instrucción para dale x0
En este contexto no es necesario hacer eso

NOTA

Explicación del código.

① Me copio el valor de SO en R0, R1, R2, R3

② Limpieza de R0

Dado que el BYTE 0 involucra con los bits de 0 al 7 voy a querer que decrece con eso en R0.

③ Muevo todos los BITS a izquierda y dado que uso SLLI me completa con 000 desde A [24]. Una vez que movi a [24] "BYTE" tengo el BIT "0" en la posición 24, el 1 en la 25 ...

Y el 7 en la 31.

④ Muevo el BYTE a su lugar correspondiente

Para eso hago SRLI que me completa ~~los bits desde A BIT 0~~ con 0 y desplazo los bits tanto como desee. Para el primer uso tengo que llevar a lograr que el BIT 24 (MOST SIG) este en el BIT 0, por lo tanto desplazo a derecha 24 BITS

③ Caso limpieza BYTE 3: Dado que ya esta lo mas a la izq. posible, si lo desplazo perdo información, para mi caso esto no lo desplazo A [24], sino que a derecha "24 BITS" para lograr que el BIT 24 quede en el BIT (0)

④ Hago la suma BIT A BIT entre cada uno de los registros y la almaceno en R0.

B ② HANOI PSEUDO CÓDIGO.

```
HANOI (M):
IF (M == 1)
  RET 1
ELSE
  TEMP0 = HANOI (M-1)
  TEMP1 = 2 * TEMP0
  TEMP2 = TEMP1 + 1
  RETURN TEMP2
```

REGISTROS: a0 = M
ra
t0 = 1 // PARA GRAFICAR IF

VARIÁVEIS A GUARDAR: ra
EM MEMÓRIA

CÓDIGO RISC V

COMENTÁRIO: ASSUMO CONTEXTO M ≥ 1

```
HANOI: ADDI t0, ZERO, 1        (1)
       BEQ a0, t0, BASE        (2)
       ADDI sp, sp, -4         (3)
       SW ra, 0(sp)            (4)
       ADDI a0, a0, -1         (5)
       JAL HANOI               (6)
       ~~ADDI a0, a0, -1~~ SLLI a0, a0, 1   (7)
       ADDI a0, a0, 1          (8)
       LW ra, 0(sp)            (9)
       ADDI sp, sp, 4          (10)
       RET                     (11)

BASE: RET                      (12)
```

(B⁻) ③

EXPLICACIÓN:

① ASIGNO 1 AL REGISTRO R0 PARA VERIFICAR EL IF.

② VERIFICO SI R0 (CASO) ES IGUAL A 1, SI ES IGUAL SALTO AL CASO BASE, SINO A LA INSTRUCCIÓN SIGUIENTE [SUMA 4 AL PC].

③ BUSCO GUARDAR UN ESPACIO EN MEMORIA, POR ESO RESTO 4 AL PC.

④ GUARDO EL VALOR DE RETORNO EN MEMORIA.

⑤ RESTO 1 A R0 PARA ASÍ CALCULAR EL HANOI (n-1).

⑥ HAGO UN SALTO A HANOI GUARDANDO EN RA EL VALOR DEL PC +4 (GUARDA LA SIG. INSTRUCCIÓN)

⑦ MULTIPLICO R0 POR 2 (DESPLAZO 1 BIT A IZQ) = 2×HANOI (n-1)

⑧ SUMO 1 A R0 = 2×HANOI (n-1) +1

⑨ CARGO EL VALOR DE LA MEMORIA (UN VALOR QUE ALMACENAMOS ANTERIORMENTE) EN RA

⑩ RESTAURO / LIMPIO LA MEMORIA INCREMENTANDO EN 4

⑪ HAGO RET PARA SOLTAR EL VALOR DEL RA OBTENIDO CON EL SW

⑫ CASO BASE, SE QUE R0 = 1 YA QUE CAI EN EL CASO BASE ENTONCES ME ALCANZA CON DEVOLVER EL VALOR QUE EXISTA EN RA CON UN RET Y NO CAMBIAR EL VALOR DE R1 QUE ESTÁ EN EL PSEUDOCÓDIGO QUEDA TOTALMENTE ✓

B

③

.TEXT

LA a0, NOTAS

LA a1, LARGO

~~ADDI t0, a1, 0~~

LBU a1, 0(a1) // VALOR DE LARGO SIN EXTENCIÓN DE SIGNO

ADDI t0, a1, 0 // CONTADOR DEL LARGO
ADDI t3, zero, 0 // CONTADOR MAYORES < 0xA0

WHILE: Beq t0, zero, FIN

LBU t1, 0(a0)

ADDI t4, zero, 0xA0

~~SUB~~ SLTU t2, t4, t1 // 1 SI t4 < t1 (PARÁMETRO < NOTA) SINO 0

ADD t3, t3, t2 // SUMO EL VALOR AL CONTADOR

ADDI t0, t0, -1
ADDI a0, a0, 1

J WHILE

FIN: ADDI t0, a1, 0 // ME GUARDO A DONDE ESTABA EL VALOR DEL LARGO

SRLI t0, t0, 1

SLT a0, t0, t3 // 1 SI LARGO/2 < CONTADOR

RET MISMO QUE EJERCICIO 1)

NOTA

B

(5) LA VENTAJA DE TENER UN REGISTRO CONSTANTE CERO ES EN PARTE LA FACILIDAD PARA CREAR PSEUDO INSTRUCCIONES, LAS CUALES LE FACILITAN AL PROGRAMADOR LA CREACIÓN DE CÓDIGO MAS CLARO Y CONCISO. LAS ~~[illegible]~~ PSEUDO INSTRUCCIONES SON UNA CONFIGURACIÓN DE LAS INSTRUCCIONES DE RISC V CON ALGUNA CONFIGURACIÓN ESPECIAL EN LOS REGISTROS. POR EJEMPLO SI TOMAMOS RET.

ESTA INSTRUCCIÓN BASICAMENTE ES UNA SALR x0, x1, 0 Y AL MISMO TIEMPO ES MUCHO MAS FACIL DE ESCRIBIR Y RECORDAR.

ESTA CREACIÓN DE PSEUDO INSTRUCCIONES SE PERMITEN YA QUE EL REGISTRO X0 NO PERMITE SU ESCRITURA Y POR ESO SI TRATO DE ESCRIBIR EN EL NO VA A AFECTAR EN NADA AL REGISTRO Y VA A SEGUIR TENIDO EL VALOR CONSTANTE "0".

OTRO PUNTO DE VISTA ES QUE ~~RET~~ RET ESTA USANDO EL REGISTRO X0 PARA ASIGNAR ~~[illegible]~~ UN REGISTRO DE DESTINO AL CUAL NO VA A PODER ~~[illegible]~~ MODIFICAR, YA QUE A RET SOLO LE INTERESA SACAR EL VALOR DEL PC ALMACENADO EN RA.

(6) LAS ESCRITURAS, POR LO DICHO ANTERIORMENTE, ESTAN PROHIBIDAS. ES DECIR SI YO TRATO DE PISAR EL VALOR DE X0 CON OTRO VALOR SIMPLE NO VA A HACER NADA, SU VALOR SEGUIRA CONSTANTE EN 0.

MIENTRAS QUE LA LECTURA AL REGISTRO SI ESTAN HABILITADAS, OSEA SI TRATO DE LEER ~~[illegible]~~ EL VALOR DE X0 VOY A OBTENER EL "0".

ENTIENDO QUE ENTENDES LA IDEA, NO ES QUE ESTEN PROHIBIDAS Y SE GENERE UN ERROR, SINO QUE CUANDO SE TRATE DE ESCRIBIR NO VA A TENER EFECTO ALGUNO.

NOTA