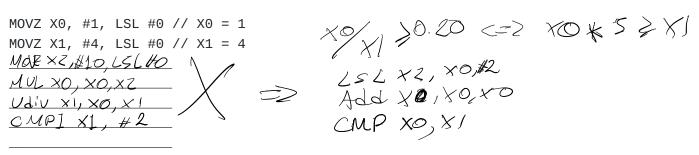
# Parcial 2 2023-06-16

Ejercicio 1
 Determinar cuáles de las siguientes instrucciones pueden ser ensambladas en LEGv8
 Justificar las respuestas negativas

Instruccion	SI/NO	Justificacion
BR 0	SÍ	
STUR X3, [X1, XZR]	10	
MOVK X15, #0xCOCA, LSL #24	NO	50/0 admite 60 16 32, 98
ORRI X0, X1, #-32	NO	vo admite
LSL X30, X31, #32	SÎ	No admits negatives Shamtes de 6 bits=> 00 a

### • Ejercicio 2

Usando el siguiente ejemplo, escriba un programa en LEGv8 que divida x0 por x1 y salte dependiendo si el resultado es mayor igual a 0.20. Se pueden usar todas las instrucciones de LEGv8 excepto las de punto flotante. Debe resolverse en 5 instrucciones o menos.



B.GE end // Saltar si X0/X1 >= 0.20

## • Ejercicio 3

Dada la siguiente sección de un programa en assembler LEGv8, el registro X1 contiene la dirección base de un arreglo A, mientras que X0 contiene el tamaño de dicho arreglo. Asuma que los registros y la memoria contienen los valores mostrados en la tabla al inicio de la ejecución de dicha sección.

X10=0 ADDI X10, XZR, #0 X0 = 3 SUBI X0, X0, #1 LOOP: CMP X10, X0 B.GE LOOP\_END // GT ← mayor o igual LSL X11, X10, #3 × 1 = × 10 \* 8 PROC: ×11=0×10000100+×1/ ADD X11, X1, X11 LDUR X12, [X11, #0] LDUR X13, [X11, #8] CMP X13, X12 B.GT NO\_XCHG // GT: mayor STUR X13, [X11, #0] STUR X12, [X11, #8]

NO\_XCHG: ADDI X10, X10, #1

B LOOP

LOOP\_END: ...

#### Memoria

Dirección	Contenido (antes)	Contenido (después)
0x10000100	-15	-30
0x1000010%	-30	- 1.5
0x10000110	70	( 🔿
0x10000118	10	70
0x10000120	200	200
0x10000128	80	80

Registros	
X0	0x0000004
X1	0x10000100

2 de 4 25/6/23, 16:36

## Responder

- a) ¿Cómo queda el contenido de todas las posiciones de memoria mostradas en la tabla al finalizar la sección del programa? (Responder completando la columna en blanco de la tabla de memoria)
- b) La instrucción contenida en la línea con la label "PROC:" se ejecuta 3 veces
- Ejercicio 4
   Considere el segmento de memoria que se muestra en la primera columna de la forma [direccion : contenido] que contiene codificado un programa en ISA LEGv8. Parte

   B89/03 Eqel programa desensamblado se presenta en la segunda columna.

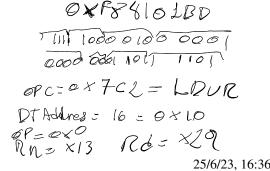
1011 100	p 100 000 000 00 11		
1110 100 7 , 161ts	//abal\ at	Desensamblado	10×14600001
bipo D DT.Adv	0x00000100:0x910013E0	ADDI x0, XZR, #4	0001 0100 0000 0000 0000
	Ox00000104:0xB89103E9	LTURSWX9, [XZR, HOX 110]	0000 0000 000)
Rt= >	0x00000108 : 0xB1000529	SUBI X9, X9, #1	OPC=000101 Branch BRAddress=L
0) 0011 0116		STURW X9, [XZR, 0x110]	=> Salta a 0 x00000118 en vez de a suip.
160 0006 00 1610×69 B.		B SKIP	en vet de a suip.
amb=#1 n= x0 d= x0	0x00000114 : 0x8B1F03E0	ADD X0, XZR, XZR	
	(skip) 0x00000118 : 0xD3600400	LSL XO, XO, #1	

## Se pide:

- a) Completar el desensamblado de las instrucciones faltantes y analice la ejecución del mismo
- b) ¿Cuántas veces se ejecuta la instrucción contenida en la dirección 0x00000114 ?

  \_\_\_\_\_\_ veces.
  c) ¿Cuál es el valos de X0 luego de la ejecución del segmento? X0:
- Ejercicio 5

  Considere que el procesador está ejecutando la instruccion de LEGv8 0xF84101BD y el contenido de los registros es X10 = 0x20 , X11=0x23 , X12=0x54 , X13=0x00 , PC=0x204
- a) Desensamblar la instrucción: LDU (L ×29, [×13,#0x10]
- **b)** ¿Qué operación realiza la ALU? : \_5 u ma
- c) Completar la siguiente tabla con el estado de las señales de control



3 de 4

Reg2Loc	ALUSrc	MentoReg	RegWrite	MemRead	MenWrite	Branch
_Q/×	1_		1_	_1_		

d) Completar la siguiente tabla con el valor de las señales inidicadas, si es de entrada o salida del bloque y la cantidad de bits de la señal

Señal	E/S	Numero de bits	valor	
Register, read data 1	5	64	0×00	
Register, read register 2	E	5	×	_
Data memory, address	5	64	0 × (0	0×00 +0×10
Register, write register	6	5	X	
Entrada del PC	5	64	0×208	0 × 20 4 + 9
Add, ALU result	5	64	0110	

