YANIA CATALINA GONZÁLEZ VALENCIA
1.088.285.640

TALLER PARA PRIMER PARCIAL ARQUITECTURA DE COMPUTADORES

1. **Describa la Taxonomía de Flynn**

   En 1996 Michel Propuso el teorema de Flynn, el cual se basa en el número de instrucciones y de secuencia de dato que la computadora utiliza para procesar información. Existen cuatro tipos de computadoras.

		DATOS	
		SIMPLE	**MULTIPLES**
INSTRUCCIONES	**SIMPLE**	SISD(Una instrucción, un dato)	SIMD(Una instrucción, múltiples datos)
	MULTIPLES	MISD(Múltiples instrucciones, un dato)	MIMD(Múltiples instrucciones, múltiples datos)

2. **Diga cuales son los 4 principios de diseño**
   1. La simplicidad favorece la regularidad
   2. Entre más pequeño, más rápido
   3. Hacer el caso común más rápido
   4. Buenos diseños demandan grandes compromisos

3. **Explique los 3 formatos de instrucciones principales existentes en la arquitectura SPARCV8**

   FORMATO 1

op1	disp 30
31 29	0

   Se utiliza para funciones de llamado.
   Op_: Específica tipo de instrucción
   Disp 30: Se utiliza para almacenar el número de saltos o desplazamientos de la memoria.

   FORMATO 2

op	a	cond	op2	desplazamiento 22 bits
2	1	4	3	22

op	rd	100	22 bits constantes
2	5	3	22

   Op_: Es el tipo de instrucción
   Cond: Elige cual es la instrucción que se utiliza
   Op2: Operando con el cual se realiza la comparación
   22 bits:

   FORMATO 3

op	rd	op3	rs1	0		rs2
2	6	5	5	1	8	5

op	rd	op3	rs1	1	signed 13 bit const
2	5	5	5	1	13

Aritmética y lógica
Rd: Registro de destino
Op3: Es la instrucción específica
Rs1: Registro fuente 1
Rs2: Registro fuente 2

4. **Explique cómo inicializar un valor grande, que ocupe más de 13 bits, en la arquitectura SPARCV8**
   Se comienza por pasar el número a binario y se completa con ceros (0) hasta alcanzar los 32 bits. Si es un valor positivo sólo se pasa a binario, si es negativo se le realiza un complemento a 2.
   Se toman los 22 bits más significativos y se invierte a positivo y se le suma 1

5. **Cómo puedo reescribir la instrucción OR y SUBcc cuando inicializo y comparo 2 registros**
   OR se puede reescribir con la instrucción MOV
   SUBcc se puede reescribir con la instrucción CMP

6. **Implementar el lenguaje de bajo nivel y lenguaje máquina a los siguientes programas:**
   a. **int main(){**

```
int a = 8;
int b = 16800;
int c = 33;
if((a+b) <= b*32){
        c = a+(b*2);
        }

else{

        return b;

        }

return a+c;

}
```

LENGUAJE BAJO NIVEL

0x0000 MOV 8, %L0
0x0004 SETHI -17, %L1
0x0008 OR %L1, 608, %L1
0x000C MOV 32, %L2
0x0010 ADD %L0, %L1, %L3

0x0014 SLL %L1, 32, %L4
0x0018 CMP %L3, %L4
0x001C BG a FALSE
0x0020 SLL %L1, 2, %L5
0x0024 ADD %L0, %L5, %L2
0x0028 BA a EXIT
FALSE
0x002C MOV %L1, %00
EXIT
0x0030 ADD %L0, %L2, %01

LENGUAJE MAQUINA

-16800

00000000000000000100000110100000
11111111111111111011111001011111
_____1_
11111111111111111011111001100000 =608

11111111111111111101111
00000000000000000010000
_____1_
00000000000000000010001 =17

	op	rd	op3	rs1	i	simm13	
0x0000	10	10000	000010	00000	1	00000000010000	
	op	rd	op2	disp 22			
0x0004	00	10001	100	11111111111111111101111			
	op	rd	op3	rs1	i	simm13	
0x0008	10	10001	000010	10001	1	0001001100000	
	op	rd	op3	rs1	i	simm13	
0x000C	10	10010	000010	00000	1	0000000100000	
	op	rd	op3	rs1	i	unused	rs2
0x0010	10	10011	000000	10000	0	00000000	10001
	op	rd	op3	rs1	i	unused	rs2
0x0014	10	10100	100101	10001	1	00000000	11111
	op	rd	op3	rs1	i	unused	rs2
0x0018	10	00000	010100	10011	0	00000000	10100
	op	a	cond	op2	disp 22		
0x001C	00	1	1010	010	00000000000000000100		
	op	rd	op3	rs1	i	unused	rs2
0x0020	10	10101	100101	10001	1	00000000	00001
	op	rd	op3	rs1	i	unused	rs2
0x0024	10	10010	000000	10000	0	00000000	10100
	op	a	cond	op2	disp 22		
0x0028	00	1	1000	010	00000000000000000100		
	op	rd	op3	rs1	i	unused	rs2
0x002C	10	01000	000010	10001	0	00000000	00000
	op	rd	op3	rs1	i	unused	rs2
0x0030	10	01001	000000	10000	0	00000000	10001

b.
```
int main(){
        int a = 8;
        int b = 10;
        if(a!=b){
                return c/8;
                }
        else{
                return b;

                }

}
```

LEGUAJE BAJO NIVEL

0x0000 MOV 8%L0
0x0004 MOV -10%L1
0X0008 CMP %L0,%L1
0X000C BE FALSE
0X0010 MOV 0 %L2
0X0014 SRL %L2, 8, %L2
FALSE
0X0018 MOV%L2 %O0
OXO2C MOV %L1 %O1

LENGUAJEMAQUINA

10001
01110
____1_
01111

	op	rd	op3	rs1	i	simm13	
0x0000	10	100000	000010	00000	1	0000000001000	
	op	rd	op2	rs1	i	simm13	
0x0004	10	100000	000010	00000	1	0000000000110	
	op	rd	op3	rs1	i	unused	rs2
0x0008	10	00000	010100	10000	0	00000000	10001
	op	a	cond	op2	disp 22		
0x000C	00	1	0001	010	00000000000000000101		
	op	rd	op3	rs1	i	simm13	
0x0010	10	10010	000010	0000	1	0000000000000	
	op	rd	op3	rs1	i	unused	shift
0x0014	10	10010	100110	10010	1	00000000	01000
	op	rd	op3	rs1	i	unused	rs2
0x0018	10	01000	00010	10001	0	00000000	00000
	op	rd	op3	rs1	i	unused	rs2
0x001C	10	01001	00010	10000	0	00000000	00000

c.

```
int main(){
        int a = - 21180;
        return a;
}
```

LENGUAJE DE BAJO NIVEL

0X0000 SETHI -21, %L0
0X0004 OR %L0,324,%LO
0X0008 MOV %L0,%O0

LENGUAJE DE MAQUINA

	op	rd	op2	disp 22			
0x0000	00	10000	100	11111111111111101010			
	op	rd	op3	rs1	i	simm13	
0x0004	10	10000	000010	10000	1	0000101000100	
	op	rd	op3	rs1	i	unused	rs2
0x0008	10	01000	000010	10000	0	00000000	00000

d.

```
int main(){
        int a = 6; int b=4;
        for(int i=0; i<=3; i++){
                c=(a+b)/8;
        }
return c;
}
```

LENGUAJE DE BAJO NIVEL

0X0000 MOV 6%LO0
0X0004 MOV 4%L1
0X0008 MOV 0 %L2
0X000C CMP %L2,3
0X0010 BG A FALSE
0X0014 ADD %L0,%L1,%L3
0X0018 SLL%L3,8%O0

0X002C BA FOR
0X0020 ADD %L2,1,%L2
SALTO FALSE
0X0024 NOP
LENGUAJE MAQUINA

	op	rd	op3	rs1	i	simm13	
0x0000	10	10000	000010	0000	1	0000000000110	
	op	rd	op3	rs1	i	simm13	
0x0004	10	10001	000010	0000	1	0000000000100	
	op	rd	op3	rs1	i	simm13	
0x0008	10	10010	000010	00000	1	0000000000000	
	op	rd	op3	rs1	i	unused	rs2
0x000C	10	00000	100101	10010	0	00000000	00011
	op	a	cond	op2	disp 22		
0x0010	00	0	1010	010	00000000000000000011		
	op	rd	op3	rs1	i	unused	rs2
0x0014	10	10011	000000	100000	0	00000000	10001
	op	rd	op3	rs1	i	simm13	
0x0018	10	1000	100101	10011	1	0000000001000	
	op	a	cond	op2	disp 22		
0x001C	00	0	1000	010	00000000000000000011		
	op	rd	op3	rs1	i	simm13	
0x0020	10	10010	000000	10010	1	0000000000001	
	op	OP	OP2	disp 22			
0x0024	00	00000	100	00000000000000000001			

8. Convierta el siguiente código a lenguaje de máquina SPARCV8.

```
int ejemplo(int x, int y, int z){

int a;

a = x y

+ z*8;

return a + 2;

}

int main(){

int x = 4, y = 2, z = 128;

int c= 0;

int c = ejemplo(x,y,z);

return c + 45;

}
```

LENGUAJE DE BAJO NIVEL

EJEMPLO
0X0000 SUB %i0, %i1,%L2
0X0004 SLL %i2,8,%i2
0X0008 ADD %L2,%i2,%L2
0X001C JMPL %O7,8,%g0
0X0010ADD%L1,2,%O1

MAIN
0X0014 MOV 4 %i0
0X0018 MOV 2 %i1
0X002C MOV -128 i2
0X0020 CALL EJEMPLO
0X0024 MOV 0 %L0
0X0028 ADD %L0,45,%O2

LENGUAJE A MAQUINA

	op	rd	op3	rs1	i	unused	rs2
0x0000	10	10010	000100	01000	0	00000000	01001
	op	rd	op3	rs1	i	simm13	
0x0004	10	01010	100101	01010	1	0000000001000	
	op	rd	op3	rs1	i	unused	rs2
0x0008	10	10010	000000	10010	0	00000000	01010
	op	rd	op3	rs1	i	simm13	
0x000C	10	00000	111000	01111	1	0000000001000	
	op	rd	op3	rs1	i	simm13	
0x0010	10	01000	000000	1001	1	0000000000010	
	op	rd	op3	rs1	i	simm13	
0x0014	10	01000	000010	00000	1	0000000000100	
	op	rd	op3	rs1	i	simm13	
0x0018	10	01001	000010	00000	1	0000000000010	
	op	rd	op3	rs1	i	simm13	
0x001C	10	01010	000010	00000	1	1111101111111	
	op	disp 30					
0x0020	01	00000000000000000000000001000					
	op	rd	op3	rs1	i	simm13	
0x0024	10	10000	000010	00000	1	0000000000000	
	op	rd	op3	rs1	i	simm13	
0x0028	10	01001	000000	10000	1	0000000101101	

9. Implemente una función mul en lenguaje de alto nivel, lenguaje de bajo nivel SPARCV8 y lenguaje de máquina SPARCV8 que realice la multiplicación de dos enteros sin signo usando solo sumas.

10. Implemente la función pot en lenguaje de alto nivel, lenguaje de bajo nivel SPARCV8 y lenguaje de máquina SPARCV8 que realice la potencia de dos números enteros sin signo realizando llamados a la función desarrollada en el punto 9.

LENGUAJE DE BAJO NIVEL

```
0X0000 MUL
0X0004 MOV%i0
0X0008 MOV 1 %L2
0X000C CMP %L2, i1
0X0010 BGE RETORNO
0X0014 ADD %L1,i0,%L3
0X0018 ADD %L2,1,%L2
0X001C RETORNO
0X0010 JMPL% O7,8,%g0
0X0014 MOV%L3,O3

MAIN
0X0018 MOV 1,%i0
0X002C MOV  2,%i1
0X0020 CMP % i2,0
0X0024 BNE ELSE
0X0028 MOV 1 %L0
ELSE
0X003C MOV 1 %L0
0X0030 CMP L0,i1
0X0034 BG EXIT
0X0038 CALL MUL
0X004C MOV 0, %L4
EXIT
0X0040 ADD %L0,1,%L0
0X0044 MOV%L4,O2
```