Taller #001 - Febrero de 2016

- 1. Escriba los 4 principios de diseño de hardware aprendidos en clase.
 - 1. La simplicidad favorece la regularidad
 - 2. Entre más pequeño, más rápido
 - 3. Hacer el caso comðn mÃ;s rÃ;pido
 - 4. Buenos diseños demandan buenos compromisos

2. Convertir a instrucciones de bajo nivel.

```
int x = 0;

int y = 8;

int z = 1;

y=x+3;

z=z+3;

x=(x-z)+(3+y);

# Inicializo las variables

add %g0, 0, %l0; # x = 0

add %g0, 8, %l1; # y = 8

add %g0, 1, %l2; # z = 1
```

					ĺ	simm13	hex
add	10	10000	000000	00000	1	000000000000	A0002000
						000000001000	
add	10	10010	000000	00000	1	0000000000001	A4002001

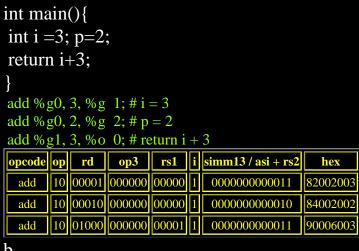
Procedo con las operaciones

add %10, 3, %11; # y = x + 3; add %12, 3, %12; # z = z + 3; sub %10, %12, %10; # x = x - z; add %11, 3, %13; # tmp = 3 + y; add %10, %13, %10; # x = x + tmp;

opcode	op	rd	op3	rs1	i	simm13/	asi + rs2	hex
add	10	10001	000000	10000	1	0000000	000011	A2042003
add	10	10010	000000	00010	1	0000000	000011	A400A003
sub	10	10000	000100	10000	0	00000000	10010	A0240012
add	10	10011	000000	10001	1	0000000	000011	A6046003
add	10	10000	000000	10000	0	00000000	10011	A0040013

```
3. Usar el ld, y st.
a[4] = a[2] + x; y = y[40] + 13;
# %10 contiene a[]
# %11 contiene y[]
add \% g0, 0, \% 13; \# x = 0
ld [%10 + (2*4)], %1 4; # cargo el contenido de a[2] en %14
add %13, %14, %14; \# %14 = a[2] + x
st %14, [%10 + (4*4)]; # guardo el contenido de %14 en a[4]
opcode op rd op3 rs1 i simm13 / asi + rs2
A6002000
 E8042008
 A804C014
                          000000010000
                                       E8242010
ld [%11 + (40*4)], %1 5; # cargo el contenido de y[40] en %15
add %15, 13, %11; \# y = %15 + 13
opcode op rd op3 rs1 i simm13 / asi + rs2
ld | 11 | 10101 | 000000 | 10001 | 1 | 0000010100000 | EA0460A0
 add 10 10001 000000 10101 1 000000001101
                                      A205600D
```

4. Convertir a lenguaje de máquina.



```
b.
int main(){
int p=3; x=1; z=4;
int w=0;
w=(p+40)+(x-z);
```

a.

return 0; # Inicializo las variables add % g0, 3, % g 1; # p = 3add % g0, 1, % g2; # x = 1 add %g0, 4, %g 3; # z = 4 add % g0, 0, % g 4; # w = 0opcode op rd op3 rs1 i simm13 / asi + rs2 hex 84002001 add 10 00011 000000 00000 1 000000000100 86002004 10 00100 000000 00000 1 0000000000000 88002000 add # Procedo con las operaciones add % g1, 40, % 10; # % 10 = p + 40sub % g2, % g3, %11; # %11 = x - z add %10, %11, %g4; # w = %10 + %11 add %g0, 0, %o0; # return 0 opcode op rd op3 rs1 i simm13 / asi + rs2 add 10 10000 000000 00001 1 0000000101000 A0006028 10 10001 000100 00010 1 00000000 00011 A220A003 sub add 10 00100 000000 10000 1 0000000 10001 88042011 10 01000 000000 00000 1 add 0000000000000 90002000

5. Inicializar las siguientes variables negativas usando OR.

```
n=-12,
a=-11,b=-14.
or %g0, -12, %l0; # n = -12
or %g0, -11, %l1; # a = -11
or %g0, -14, %l2; # b = -14
```

opcode	op	rd	op3	rs1	i	simm13 / asi + rs2	hex
or	10	10000	000010	00000	1	11111111110100	A0103FF4
or	10	10001	000010	00000	1	1111111110101	A2103FF5
or	10	10010	000010	00000	1	11111111110010	A4103FF2