***Iniciacion de variables:**

OR, %G0,1,%L1
OR, %G0,7,%L2
OR, %G0,4,%L3

***Ejecuacion de programa:**

Add %L1,%L2,%L1
Sub %L1,%L3,%L3
Ld %L0+(7*4),%L4
Ld %L0+(8*4),%L5
Add %L4,%L5,%L6
Add %L6,%L3,%L7
St %L7,[%L0+(5*4)]

Direccion	op	RD	op3	Rs1	i	unused(zero)	rs2	conversión hexadecimal
0x0000	10	10001	000010	00000	1	0000000000001		A 2 1 0 2 0 0 1
0x0004	10	10010	000010	00000	1	0000000000111		A 4 1 0 2 0 0 7
0x0008	10	10011	000010	00000	1	0000000000100		A 6 1 0 2 0 0 4
0x000C	10	10001	000000	10001	0	00000000	10010	A 2 0 4 4 0 1 2
0x0010	10	10011	000100	10001	0	00000000	10011	A 6 2 4 4 0 1 3
0x0014	11	10100	000000	10000	1	0000000011100		E 8 0 4 2 0 1 C
0x0018	11	10101	000000	10000	1	0000000100000		E A 0 4 2 0 2 0
0x001c	10	10110	000000	10100	0	00000000	10101	A C 0 5 0 0 1 5
0x0020	10	10111	000000	10110	0	00000000	10011	A E 0 5 8 0 1 3
0x0024	11	10111	000100	10000	1	0000000010100		E E 2 4 2 0 1 4

Conclusiones:

. Se realiza la práctica de lo explicado en clases, ayudando a adquirir destreza en las instrucciones de alto nivel y bajo nivel.

.se conoce más afondo la función de los instrucciones Load y Store que son de gran ayuda para el manejo de registros en la memoria, al igual que operador OR que una función principal para inicializar los registros globales.

. El manejo del formato 3 fue de gran importación para determinar las funciones de cada registro, el manejo de direcciones, los formatos op load-store "11" y aritmtico-logico "10".

INSTRUCCIONES DE REGISTRO

Daniela Delgado Galeano. CC.: 1088327948

***Ejercicio planteado:**

Int main(){

Int d=1

Int n=7

Int m= 4

b[5]=(((d+n)-m)+(b[7]+b[8]))

***Registros:**

d= %L1 b= %L0

n= %L2 m= %L3

Procedimiento:

Despues de tener plantado los registo se inicia el lenguaje ensamblador:

1. se declara los registros de cada variavle del programa.

2. se inicializan los registros utilizando registros glovales con el operdor OR.

3. inicia la ejecucion del programa tomando el operador load para cargar los registro de memoria que alla en la direccion.

4. despues se continua con las operaciones simples como suma o resta, con ayuda de los operadores add y sub.

5. para concluir las operaciones que se realizaron anteriormente con ayuda del operador store se almcena el valor de las operaciones en la memoria.

6. al terminar el lenguaje ensamblador, continuamos trabajando con el lenguaje maquina, mediante el formato 3.

7. el formato 3 contiene direccione de memoria, tipo de instrucciones "op", registros destino "RD", tipo de operadores como el add (000000), sub (000100), load (000000), store(000100), y OR(000010) "op3", el imediante si en el registro hay un numero "i" y el unusired y resgistro de salida 2, estos dos ultimos campos se unen siempre que halla un inmediato y se escribe el numero que conlleva el inmediato, cabe recordad que este formato se completa trabajando con numeros binarios.

8. cuando se obtiene el formato 3 basado en el lenguaje emsablador que se realizo, se hace la conversión hexadecimal que consiste en tomar la fila del recuadro y unir los digisto de a grupos de 4 .