INSTRUCCIONES DE REGISTRO

Ejercio Planteado.

```
Int main(){
        F=11355;            F=%L1
        A[4]=A[3]+F;        A[4]=%L3    A[3]=%L0
        B=107;              B=%L4
        Return F-B-A[4];}
```

```
Sethi  11,  %L0
Or     %L1,  91,  %L1
Ld     [%L0 + (3*4)],  %L2
Add    %L2,  %L1,  %L2
St     %L2,  [%L3 + (4*4)]
Or     %g0,  %107,  %L4
Sub    %L1,  %L4,  %L5
Sub    %L5,  %L3,  %O0
```

OP	Rd	OP3	Rs1	i	unused(zero)	Rs2	conversion hexadecimal
00	10001	100			00000000000000000001011		0X2300000B
10	10001	000010	10001	1	0000001011011		0XA214605B
11	10010	000000	10000	1	0000000001100		0XE404200C
10	10010	000000	10010	0	00000000	10001	0XA4048011
11	10010	000100	10011	0	00000000	10000	0XE424C010
10	10100	000010	00000	1	0000001101011		0XA810206B
10	10101	000100	10001	0	00000000	10100	0XAA244014
10	01000	000100	10101	0	00000000	10011	0X90254013

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”.

Procedimiento:

1. Se declara los registros de cada variable del programa.
2. Después tomamos cada una de las funciones para analizar que instrucciones tienen y así hacer el código SPARK V8.
3. En la primera instrucción tenemos en cuenta que es un número que supera el número de registros así que hallamos el numero en binario y completamos con ceros hasta tener 32bytes.
4. Luego seguimos identificando las instrucciones load, store y add para poder cagar el registro, sumarlo y guardarlo en otra posición.
5. A continuación, seguimos con una variable que se le asigna un registro así que utilizamos la instrucción Or con el registro global 0
6. Para la ultima instrucción es tan solo restas para obtener el return.
7. Luego pasamos estos registros a lenguaje de maquina dependiendo cada formato de cada instrucción para convertirlos a en ceros y unos.
8. Y finalizamos con el código Hexadecimal.

Daniel Humberto Gallego López.

CC: 1093225688