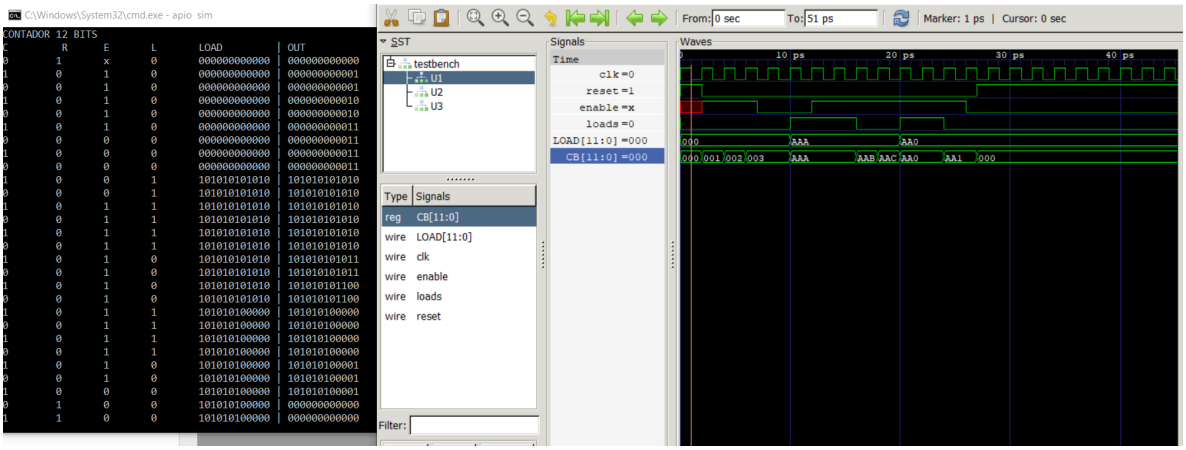
Gabriel Alexander Fong Penagos

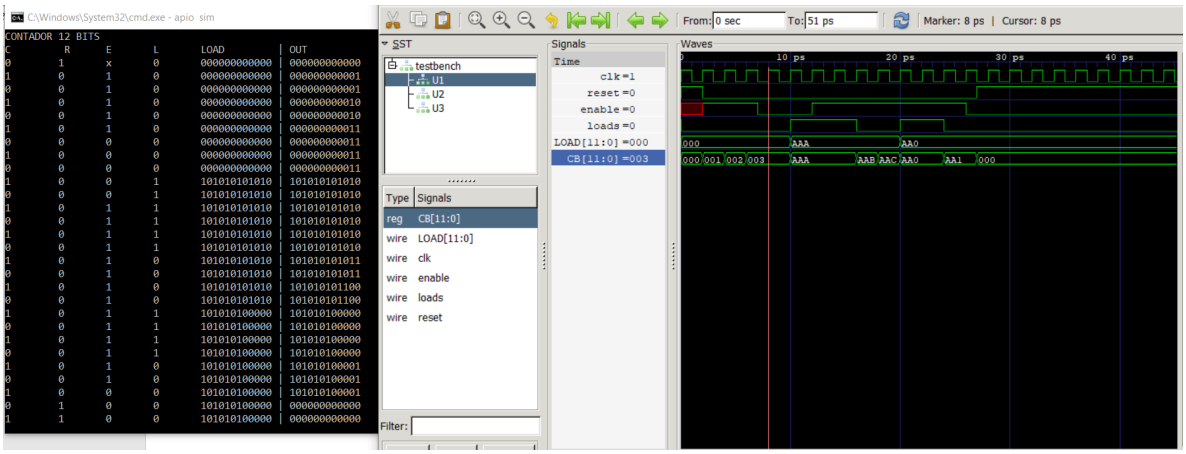
LABORATORIO #8

Ejercicio #1

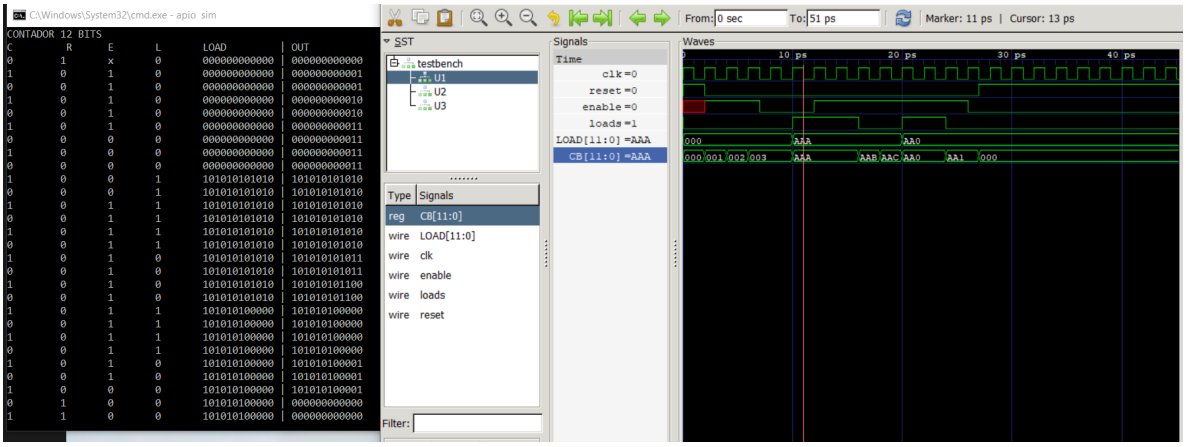
El reset = 1 coloca la salida del contador es 000.



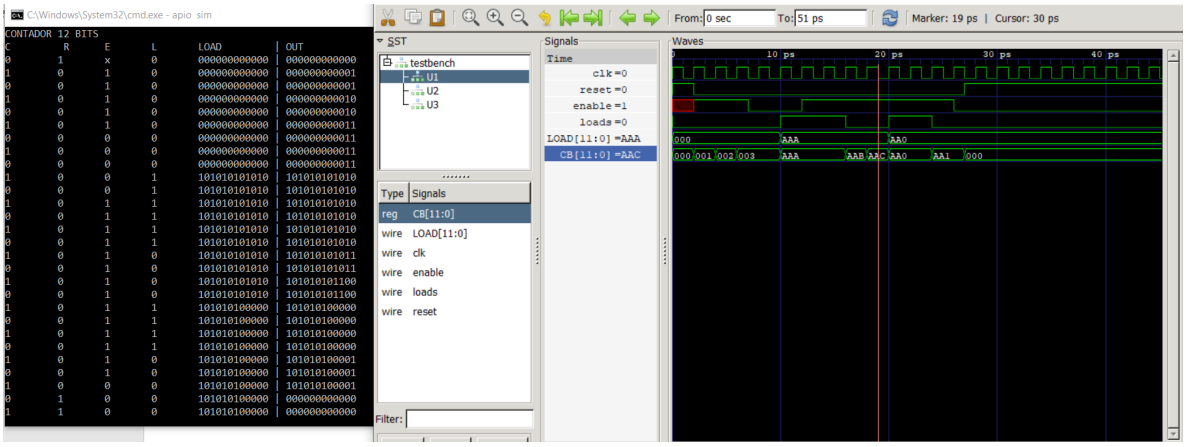
El enable = 1, la salida del contador comienza a sumar 1 en cada flanco positivo.



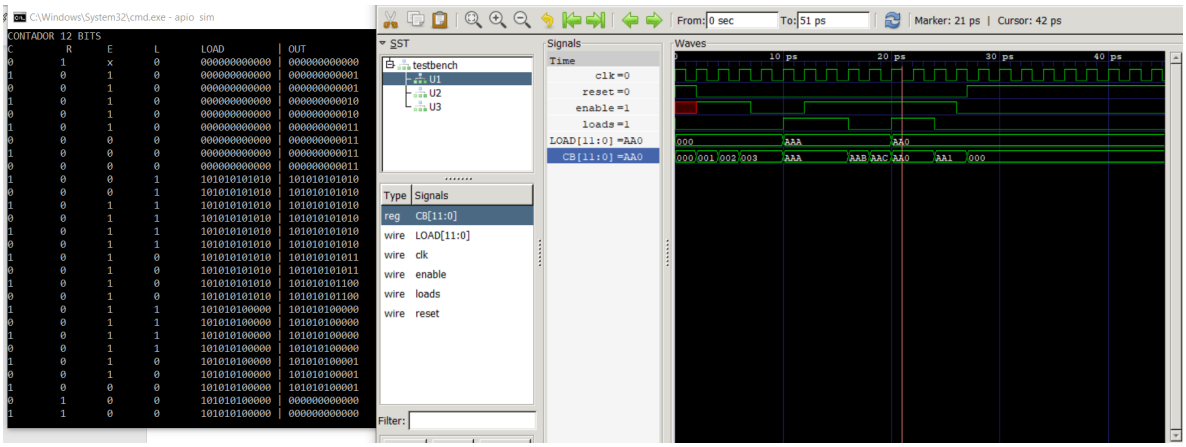
El enable = 0, la bandera del load = 1 salida del contador es igual al valor pre-colocado en el LOAD, aunque enable = 1, la señal de salida sigue siendo = a la pre-colocada.



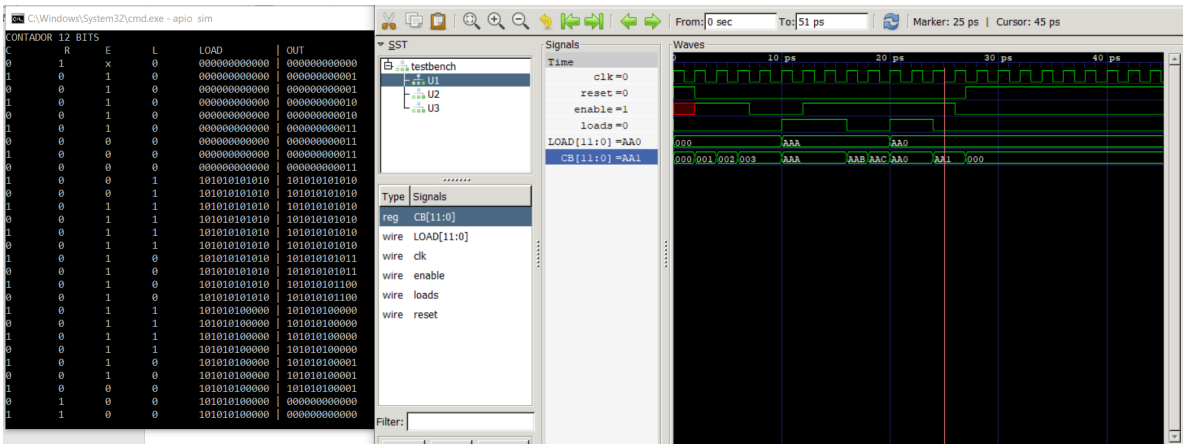
Después de deshabilitar la bandera del load y el enable = 1, vuelve a contar + 1 desde el valor precolocado.



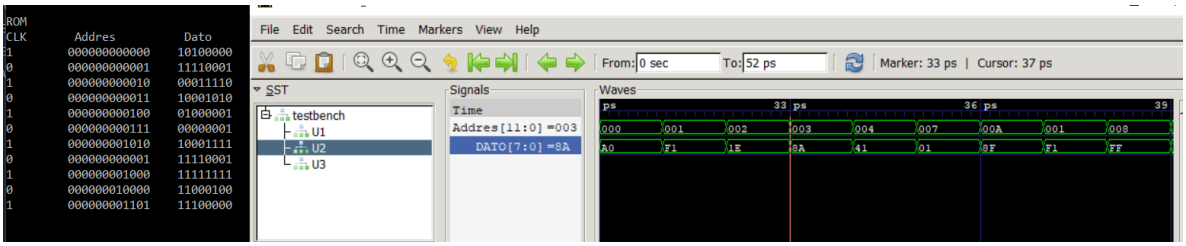
Con el enable = 1, Habilitamos la bandera del load y coloca el valor pre-colocado y dejar de contar

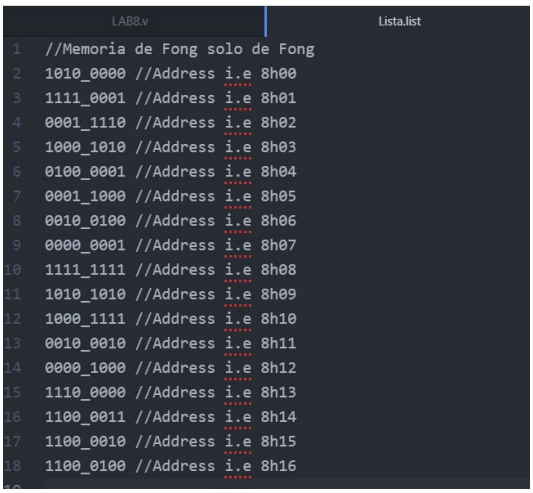


Apagamos la bandera del load y enable = 1 y sigue contando después del valor preseleccionado, y por ultimo reseteamos



Ejercicio #2





Se muestran las 10 direcciones de la lista, las cuales se ingresan en el address y nos da el valor en la salida que se encuentra en esa dirección.

El array de datos ordena y selecciona los datos en grupos y los coloca en forma de matriz. El cual genera una variable de dos dimensiones

$readmemb lee los arreglos en sistema binario.

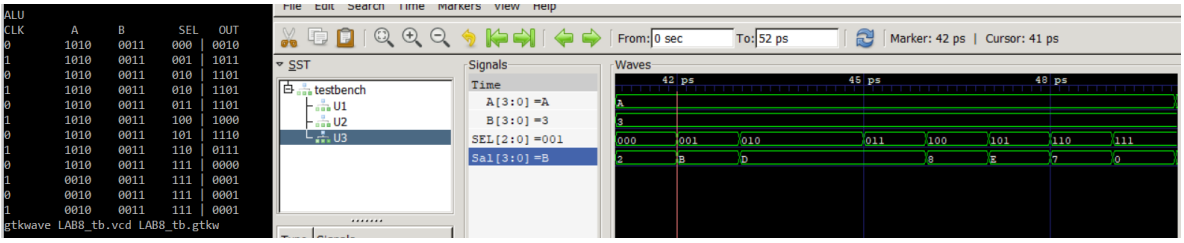
$readmemh lee los arreglo en sistema hexadecimal

Código de Lectura la lista

```
24  //Modulo de Lectura de datos de la ROM de 4k X 8
25  module ROM (input wire [11:0]Addres, // DIRECCION DE 12 BITS
26    output wire [7:0]DATO); //TAMAÑO DEL DATO DE 8 BITS
27
28    reg [7:0] MEM[0:4095]; //MEM ES DE TAMAÑO DE 2^12 = 4096 Y CREA EL ARREGLO PARA NUESTROS DATOS
29    initial begin         //INICIAMOS LA LECTURA DE NUESTRA LISTA
30      $readmemb("Lista.list", MEM);//LEEMOS LOS VALORES DE LA LISTA EN BINARIO
31    end
32    assign DATO = MEM[Addres]; //LE ASIGANMOS EL VALOR DE MEM EN LA DIRECCION INGRESADA
33  endmodule // ROM
34
```

En la línea 28 creamos el arreglo, de ancho de 8 bits [7:0] y crea 4096 localidades (2^{12}), luego en el $readmemb lee los valores de la lista creada con valores binarios. Luego asignamos el valor de la lista que nos brinda el addres.

Ejercicio 3



El funcionamiento de la ALU tiene 7 funciones u operandos diferentes los cuales los seleccionamos con una entrada de selección de 3 bits, en donde utilizamos un case que nos ayuda a seleccionar la operación que se desea realizar, el cual se encuentra dentro de un always en donde siempre que ocurra un cambio en nuestras entradas nos entrega el nuevo valor a realizar., el cual realiza las operaciones de 2 valores de entrada A y B de 4 bits, y la salida de la función es de 4bits.

Las operaciones a realizar son las siguientes

$F_{2:0}$	Function
000	A AND B
001	A OR B
010	A + B
011	not used
100	A AND $\overline{B}$
101	A OR $\overline{B}$
110	A – B
111	SLT

link del repositorio: https://github.com/GAFong/LABORATORIOS/tree/master/LABORATORIO-8