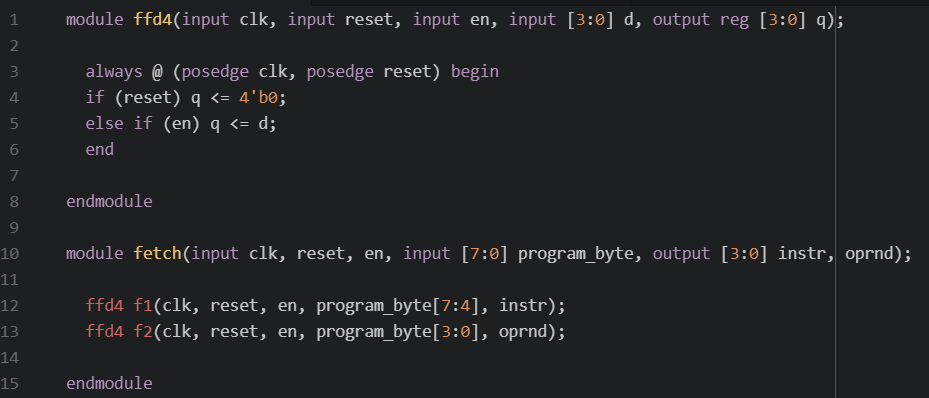
Fernando Arribas

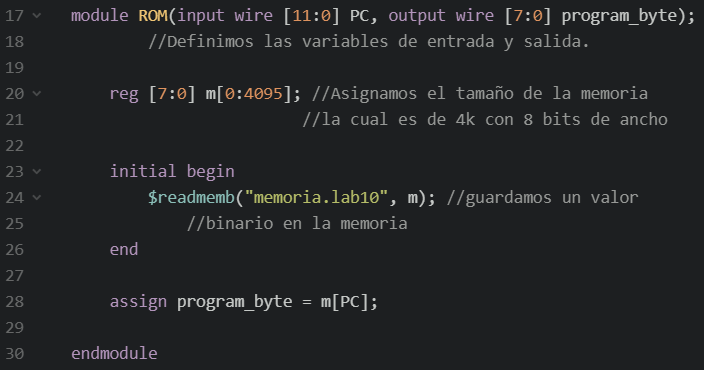
Carné: 19030

Lab 10

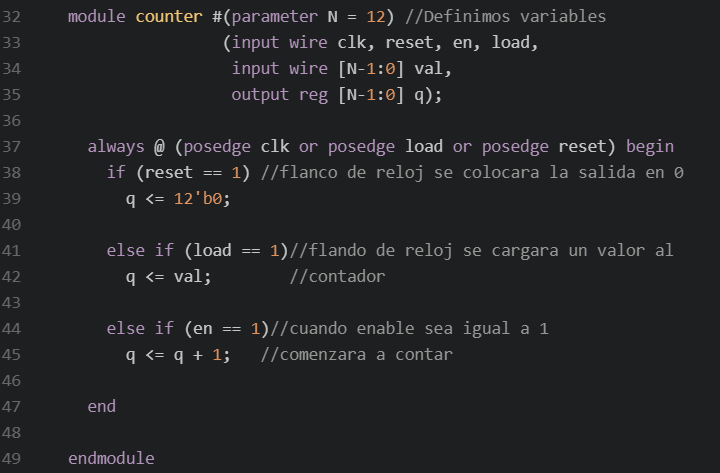
**Ejercicio 1**



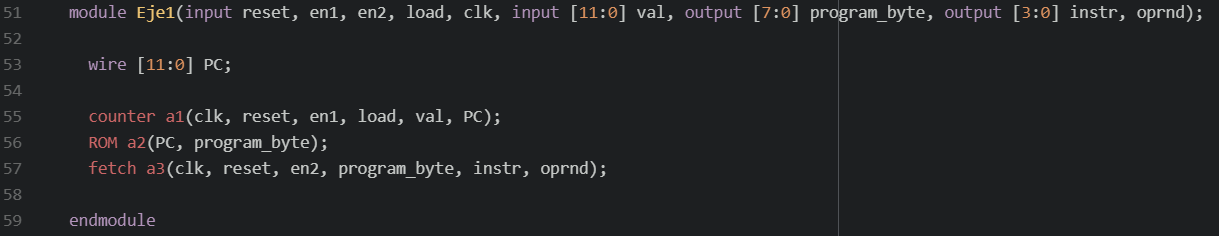
El Fetch se realizo con la utilización de 2 módulos, aunque se puede realizar a la perfección solo con uno debido a que el fetch es un flip flop tipo D de 8 bits. Pero para facilitar la muestra de las salidas Instruction y Opernad del fetch se decidió a realizar en un módulo un FFD de 4 bits para luego en un segundo módulo juntar 2 FFD de 4 bits.



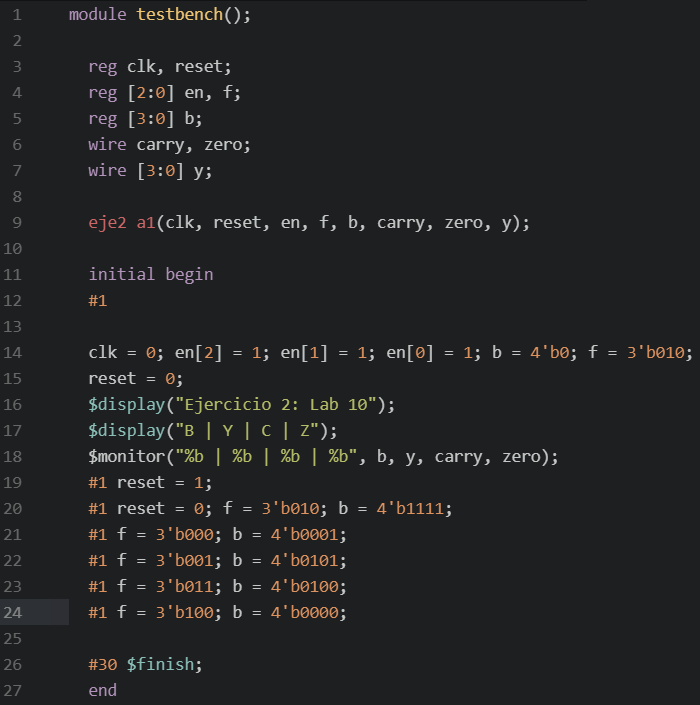
Primero declaramos las variables de entrada, siendo una entrada L de 12 bits, la cual nos servirá para buscar en las 4096 localidades de nuestra memoria y una salida Y que nos desplegará lo que esta guardado en cierta localidad de nuestra memoria. Lo primero a realizar fue un array de memoria, el cual será de 4k localidades y 8 bits de ancho, la cual será una variable M. Luego por medio de la instrucción $readmemb leeremos el archivo externo que guardaremos en nuestra memoria, la cual será guardada en el arreglo m. Por último, asignamos que el valor de la salida será la localidad que seleccionemos con L.

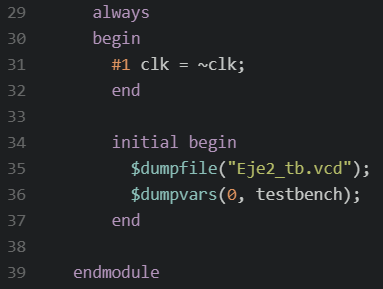


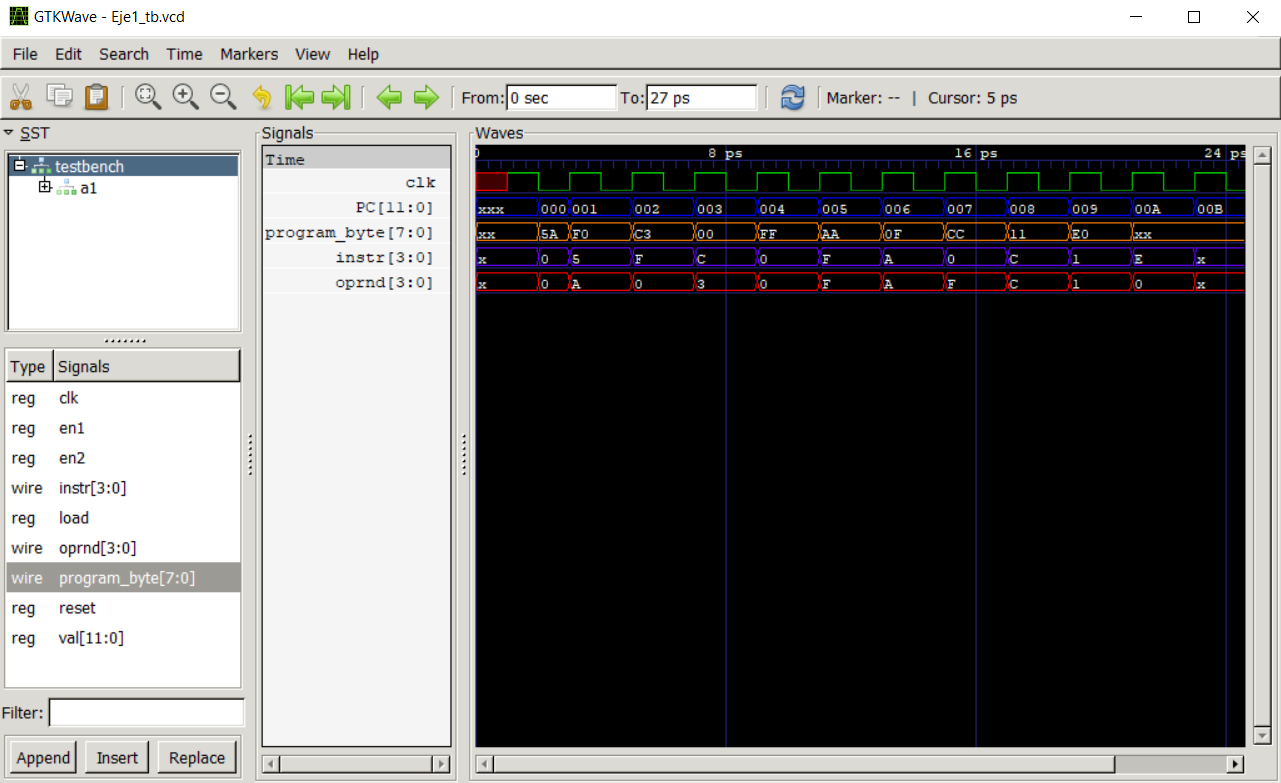
Primero definimos las variables a utilizar las cuales son un clock, un reset, un enable (en), un load,  
un variable de 12 bits llamada val, y la salida de 12 bits; adicionalmente definimos un parámetro, el  
cual nos indicará de cuantos bits será nuestro contador, en este caso 12 bits. Luego todo eso se ingresa a un always, en donde el reloj, el reset y load trabajaran con flancos de reloj positivos. Dentro existen un total de 3 condicionales, el primero indica si hay un cambio de reloj para reset la salida se colocara en 0. Luego si existe un flanco de reloj en load la salida pasara a ser los que tengamos en la variable val. Por último, si enable es igual a 1 la salida del contador será el valor pasado del mismo más uno. Hay que mencionar que todas estas operaciones están realizadas con non-blocking assignment, para que el contador pueda saltar entre las distintas opciones sin ninguna restricción.



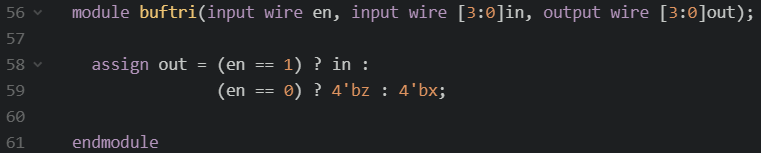
Luego se juntan los módulos fetch, contador y ROM en un mismo módulo para poder interconectarlos entre si y poder obtener los resultados deseados.



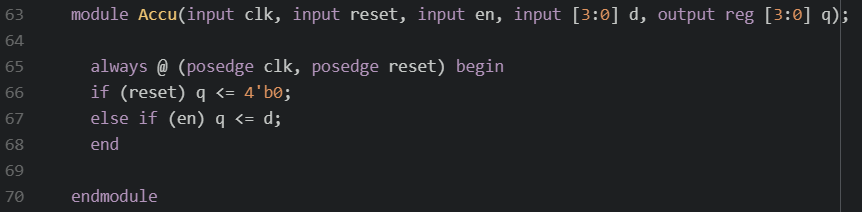




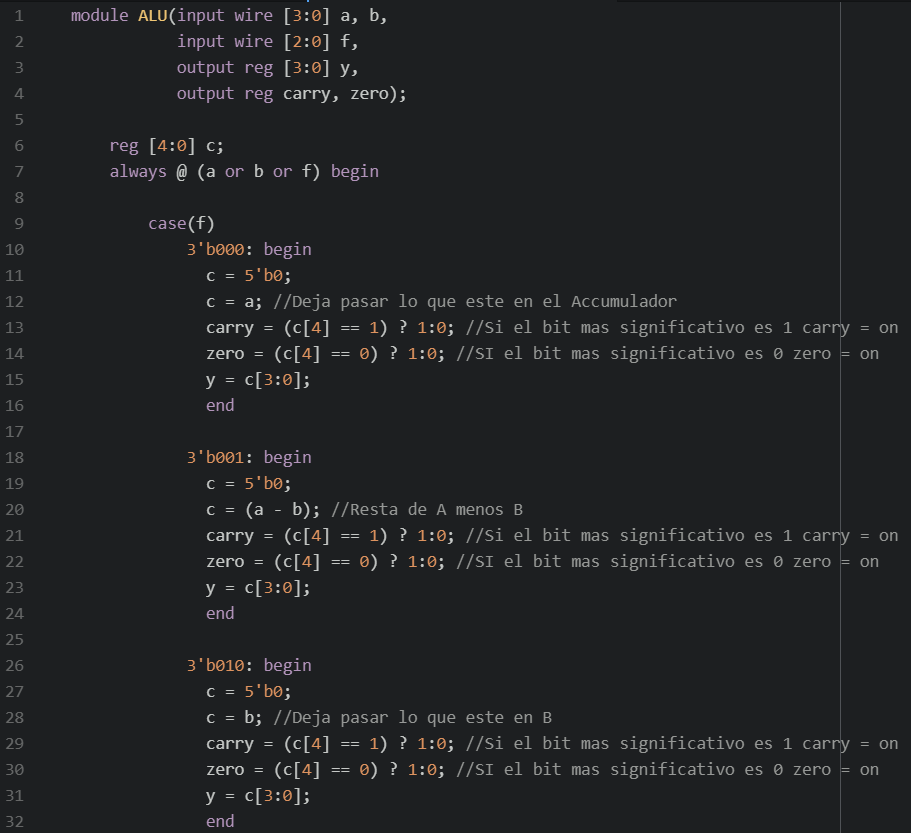
**Ejercicio 2**

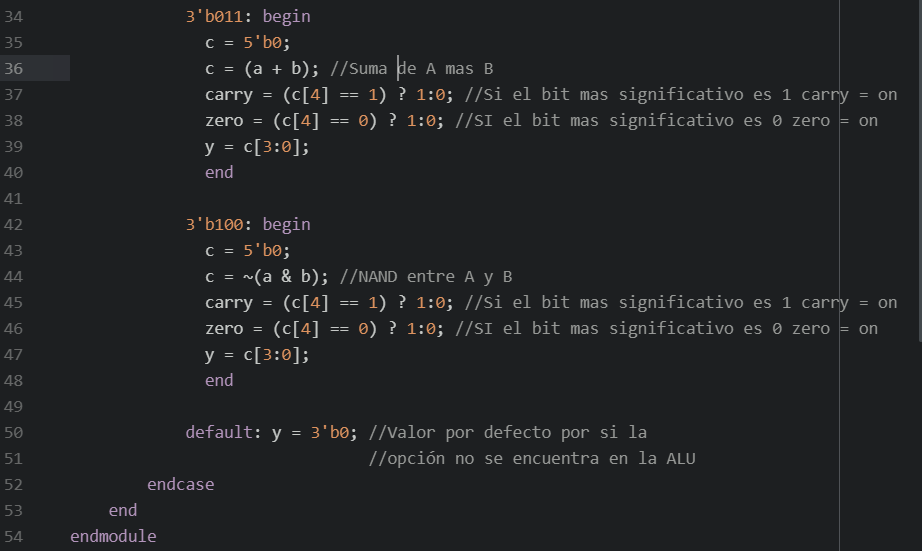


Primero se llamaron las variables, de entrada, tenemos el enable y una de 4 bits llamada in,  
mientras que de salida tenemos una variable de 4 bits llamada out. Luego empezamos a construir el buffer triestado, asignamos a out un condicional en donde si en es igual a 1 deja  
pasar a in, de lo contrario estará en alta impedancia.

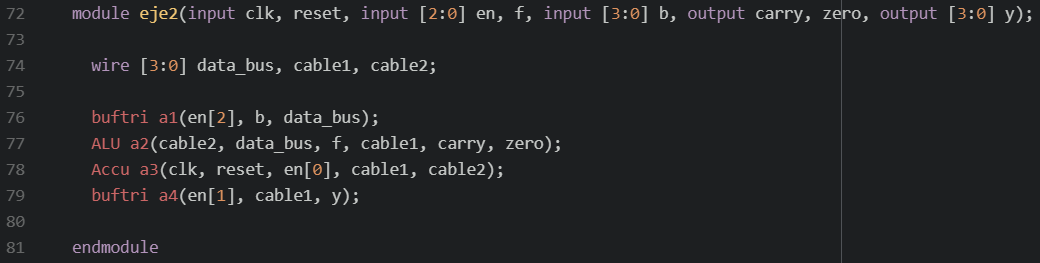


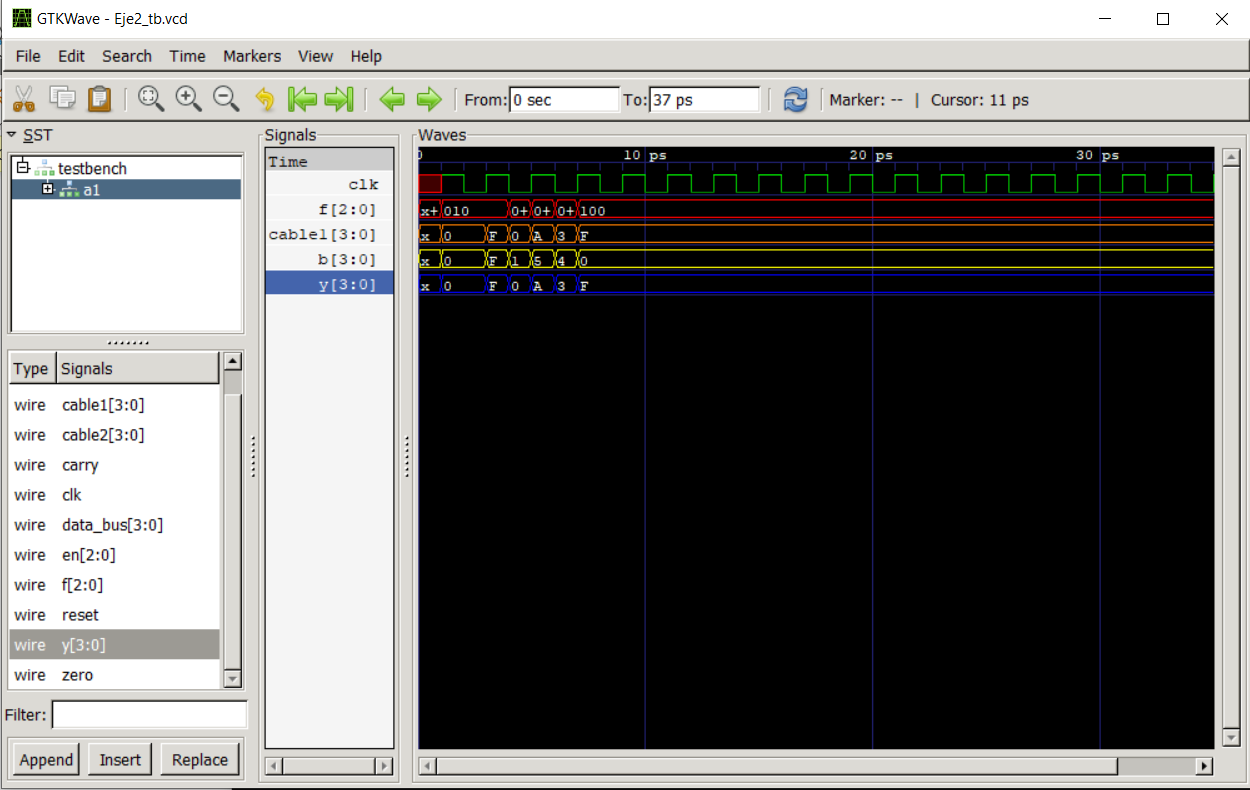
Para el accumulador se construyo un flip flop tipo de 4 bits.





Para la ALU se modifico la original para colocar una con solo cinco instrucciones, dejar pasar A, dejar pasar B, suma, comparación y una compuerta NAND o NOR. Adicionalmente se le agregaron las salidas Carry y Zero, en donde carry se activa cuando hay overflow en alguna operación y zero se activa cuando todo es igual a 0.





Link Repositorio:  
https://github.com/fernando19030/LaboratoriosElectronica\_Digital\_1-  
19030.git