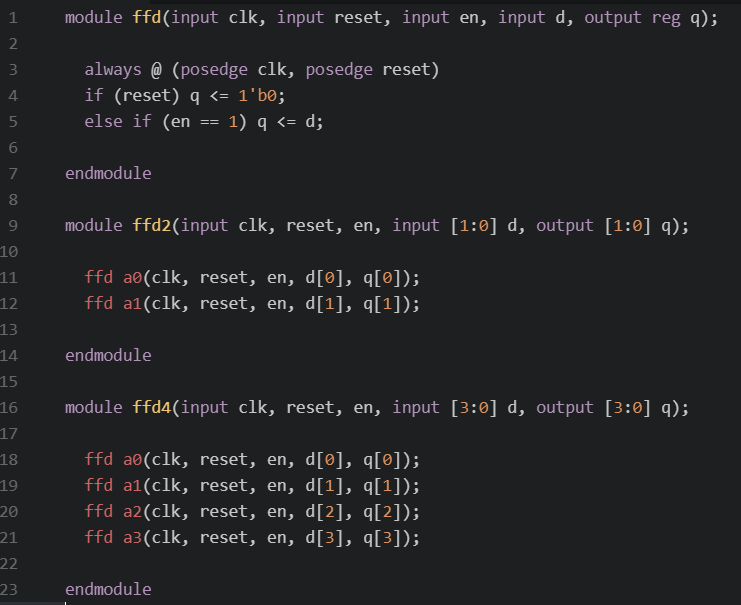
Fernando Arribas

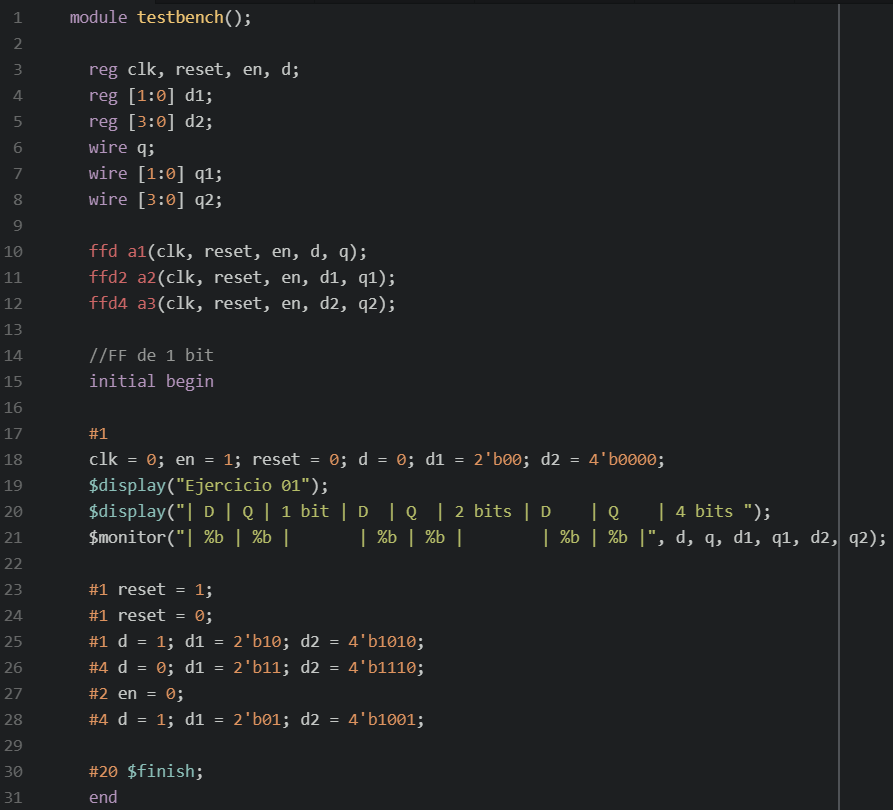
Carné: 19030

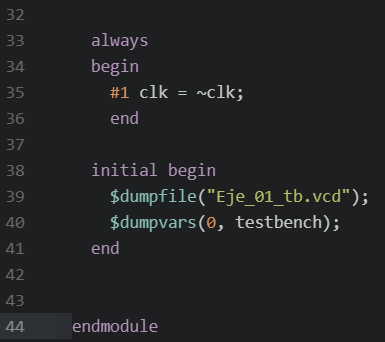
Lab 09 Digital 1

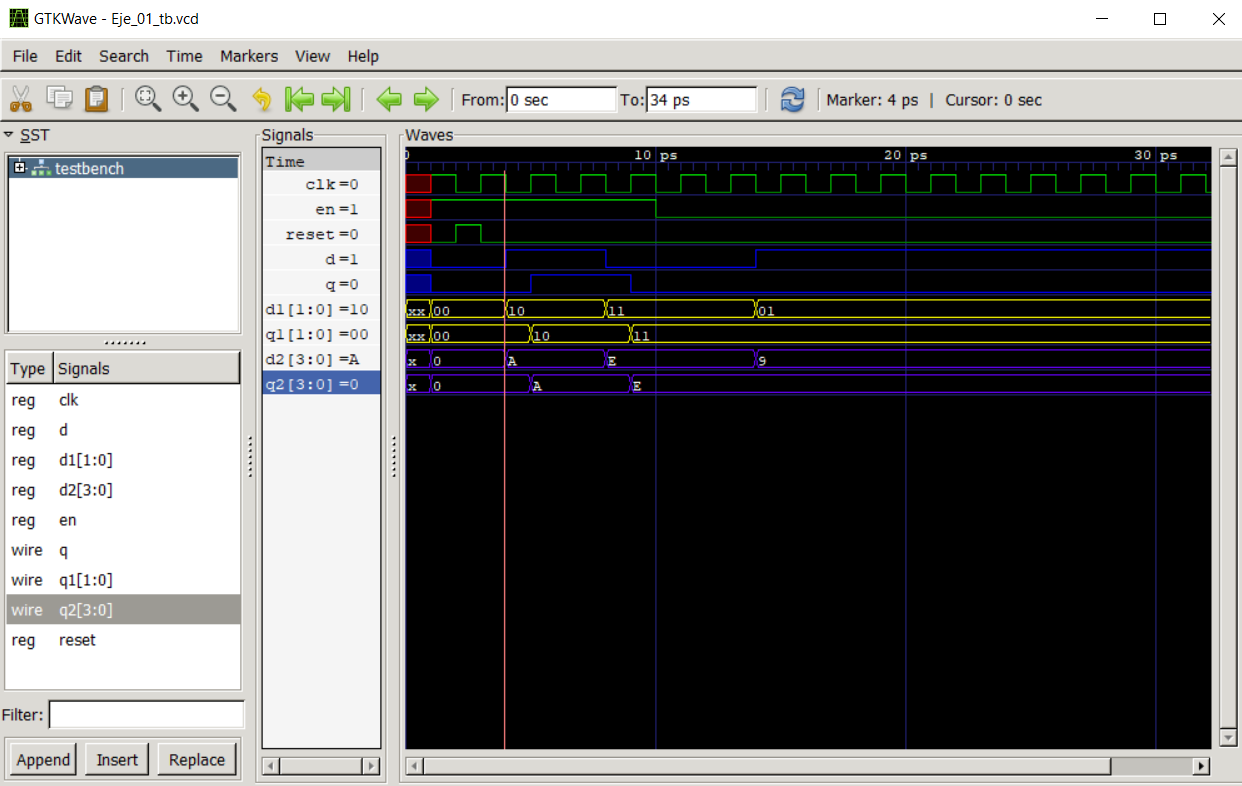
**Ejercicio 1**



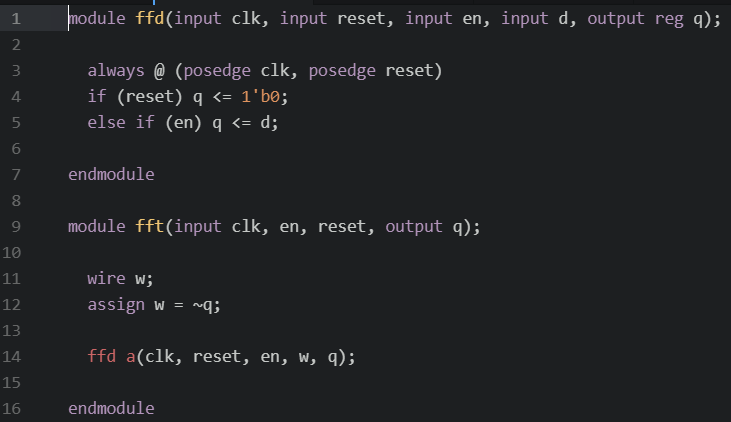
En el primer ejercicio se realizo un modulo principal, el cual es un flip flop tipo d, con el cual se prosiguieron a realizar otros dos flip flops tipo de d. Los cuales eran de 2 y 4 bits, lo cual se logró llamando el módulo principal las veces necesarias para lograr nuestra cantidad de bits. En el caso del de 2 bits se llamo un total de 2 veces el FFD, mientras que para el de 4 bits se llamo un total de 4 veces, aunque también se pudo llamar el FFD de 2 bits un total de 2 veces para acortar un poco el código.



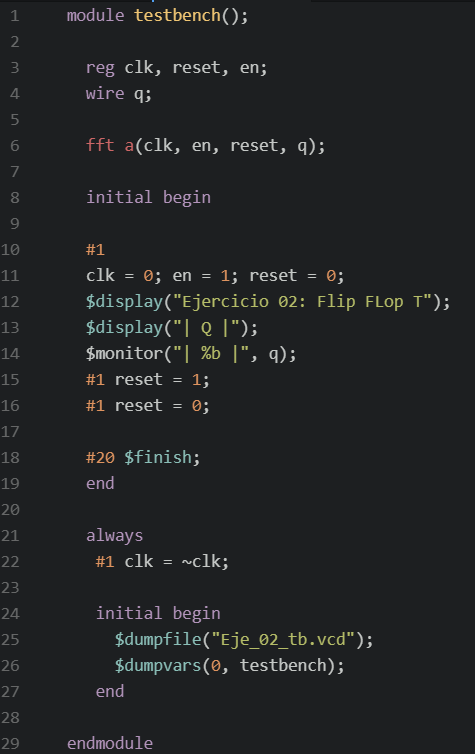


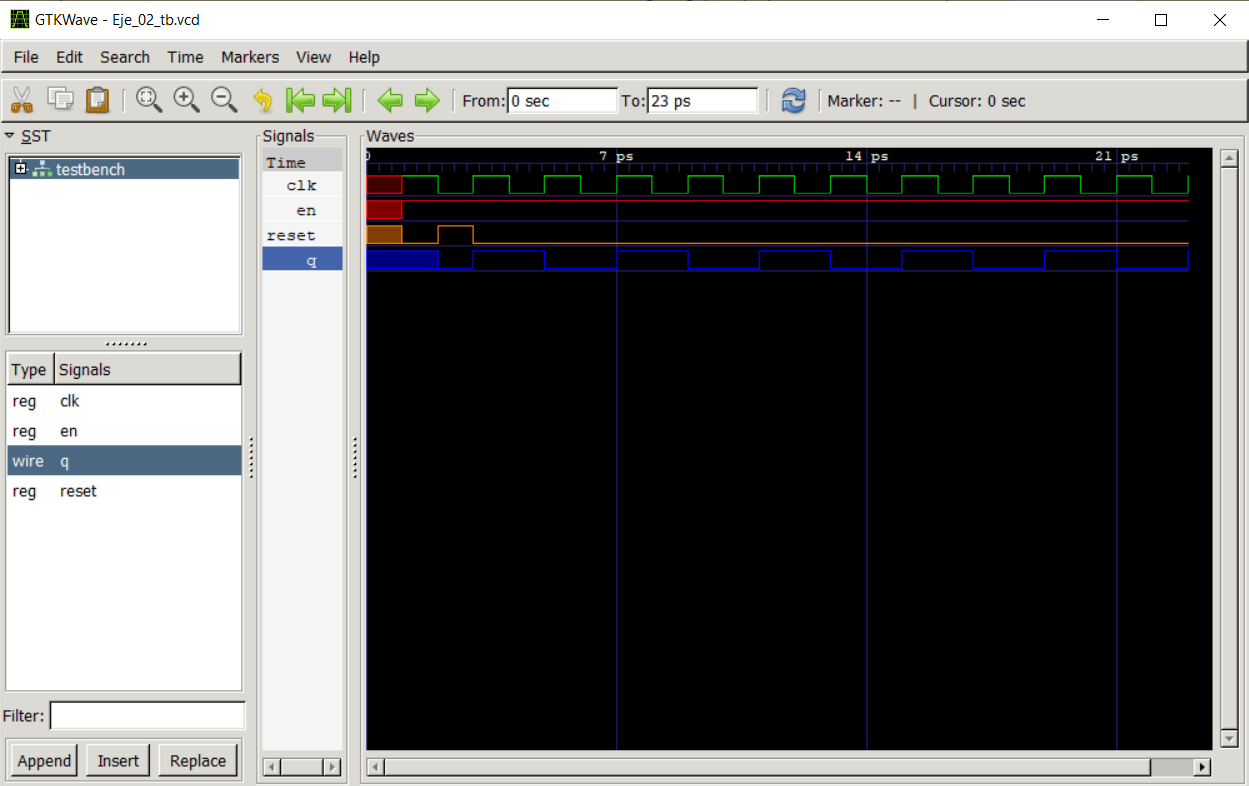


**Ejercicio 2**

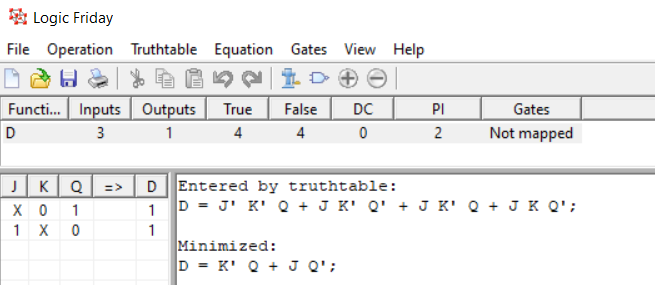


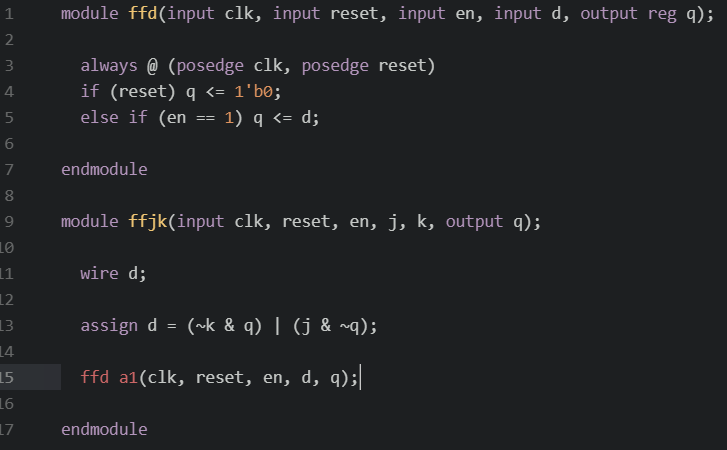
Primero se construyo un flip flop tipo d, luego se prosiguió con la construcción del toggle FF. Para lograrlo se utilizó un cable, el cual conecta la salida Q del FFD a una compuerta NOT, la cual entrara en la entrada D del FFD. Logrando así al llamar el módulo del FFD un ToggleFF.



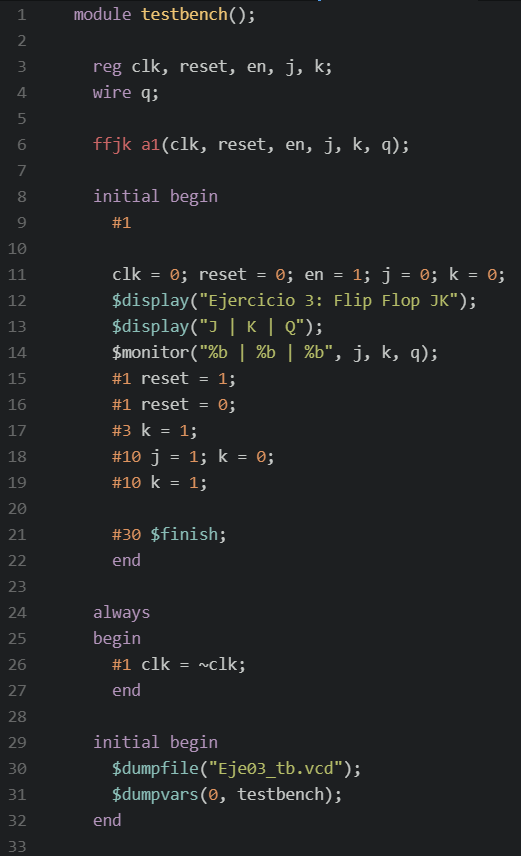


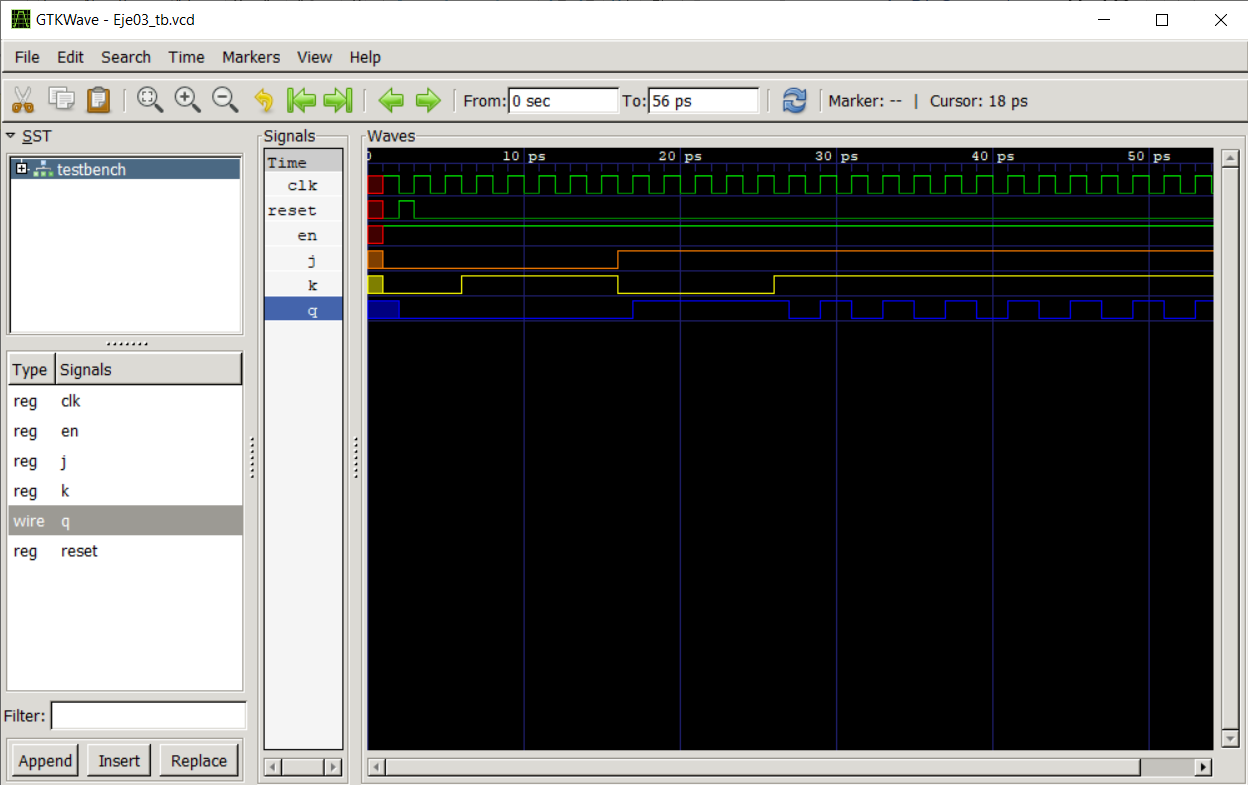
**Ejercicio 3**



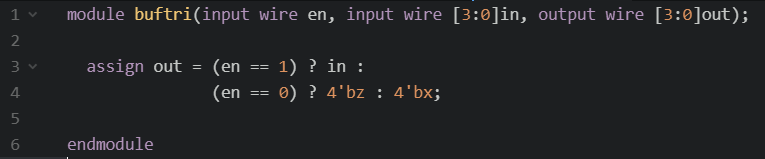


Primero se construyó una tabla de verdad, la cual describiría el comportamiento de un FFJK por medio de una nube combinacional. Luego se prosiguió a construir el módulo con el FFD, el cual fue llamado a otro modulo, en donde se primero se aplico la nube combinacional que entraría en la entrada D del FFD y se llamo al módulo el FFD.



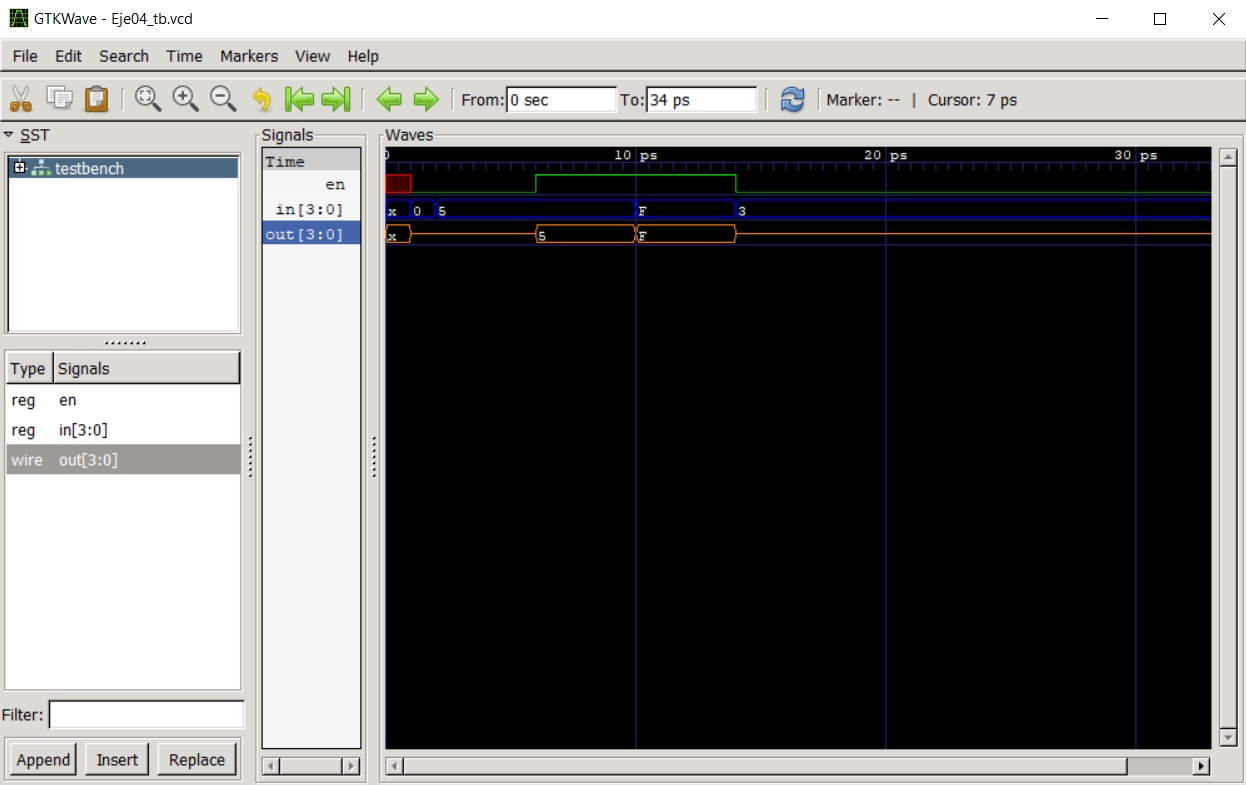


**Ejercicio 4**

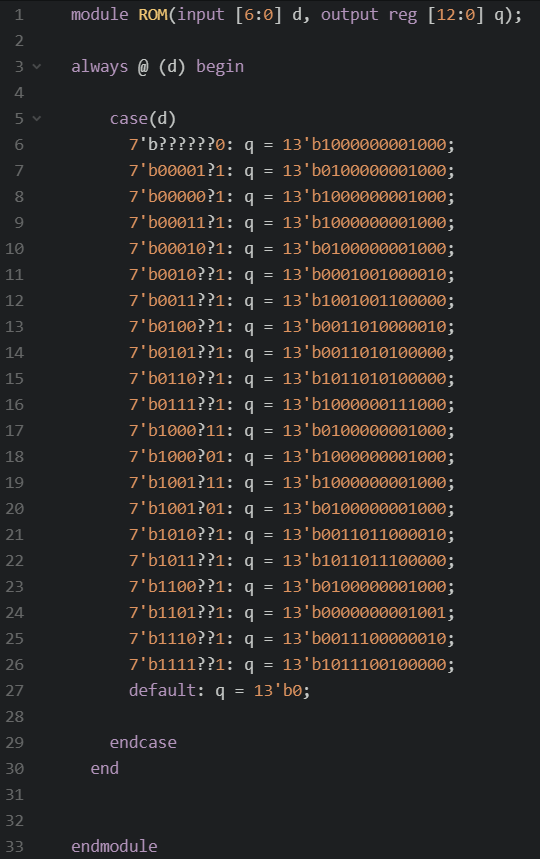


Primero se llamaron las variables, de entrada, tenemos el enable y una de 4 bits llamada in, mientras que de salida tenemos una variable de 4 bits llamada out. Luego empezamos a construir el buffer triestado, asignamos a out un condicional en donde si en es igual a 1 deja pasar a in, de lo contrario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



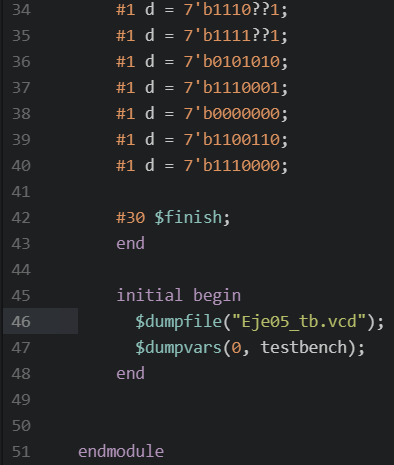


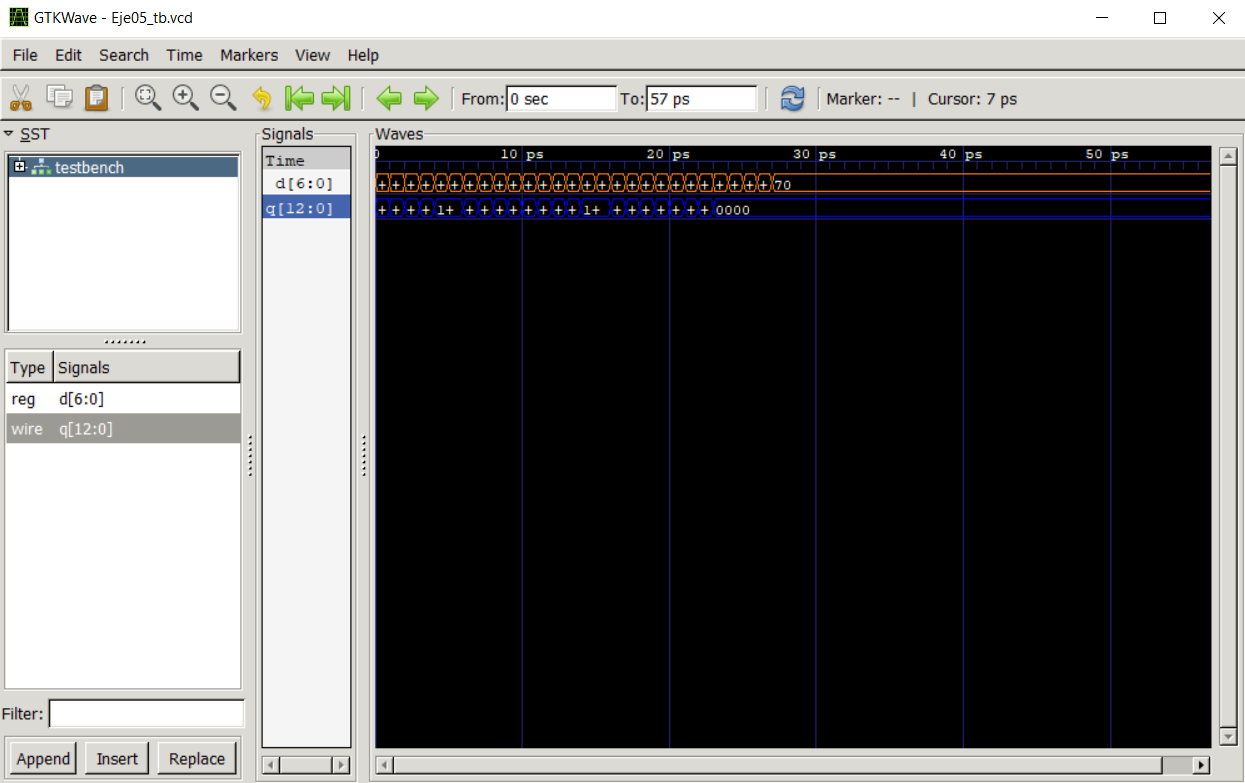
**Ejercicio 5**



La memoria ROM se construye de manera sencilla con un case, en el cual hay una entrada de 7 bits llamada d y una salida de 13 bits llamada q. En el case se colocaron las 21 combinaciones posibles que existen en la tabla, siendo los don´t cares representados en verilog por un signo ‘?’.







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