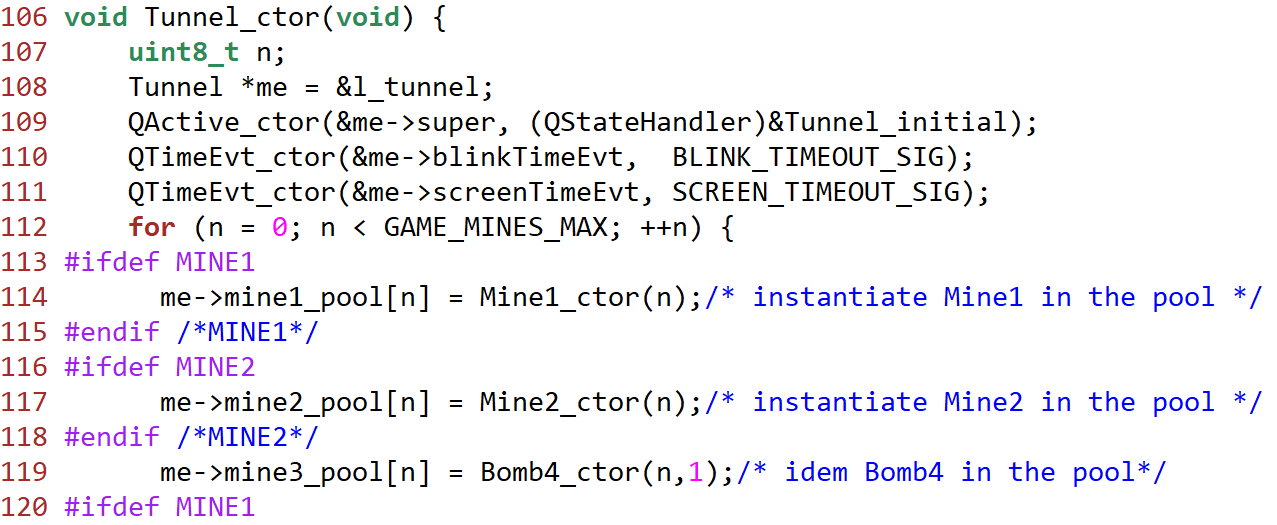
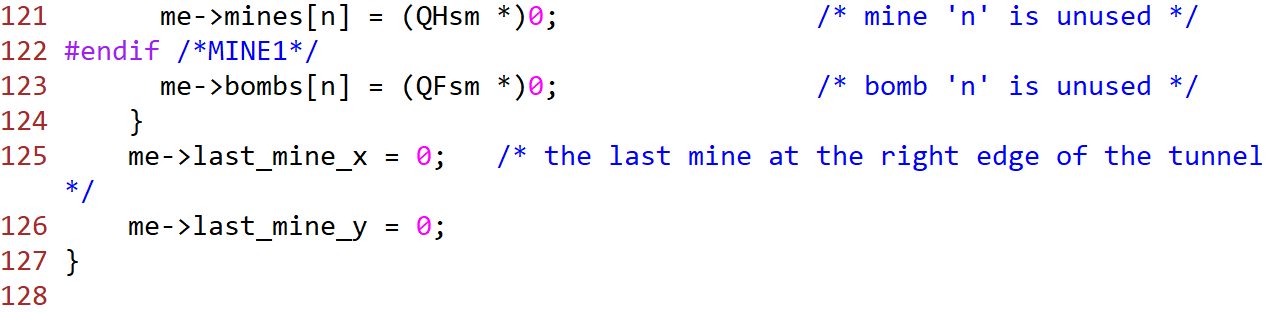
Modificaciones al código fuente de Fly and Shoot (sin Missile y sin Minas)

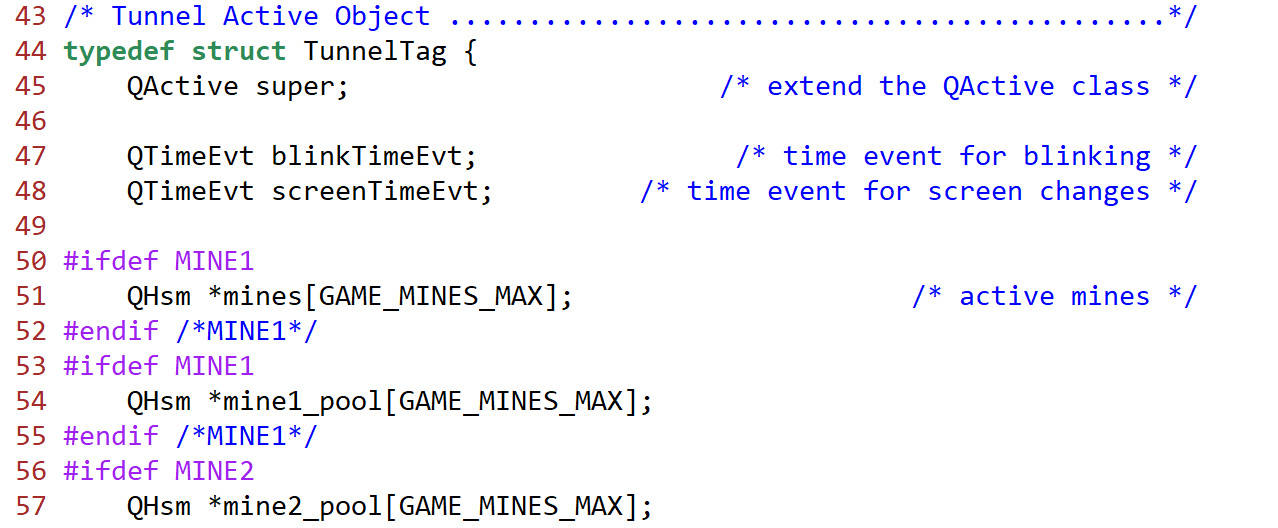
Para agregar instancias de QFsm al juego Fly and Shoot se hicieron las siguientes modificaciones.

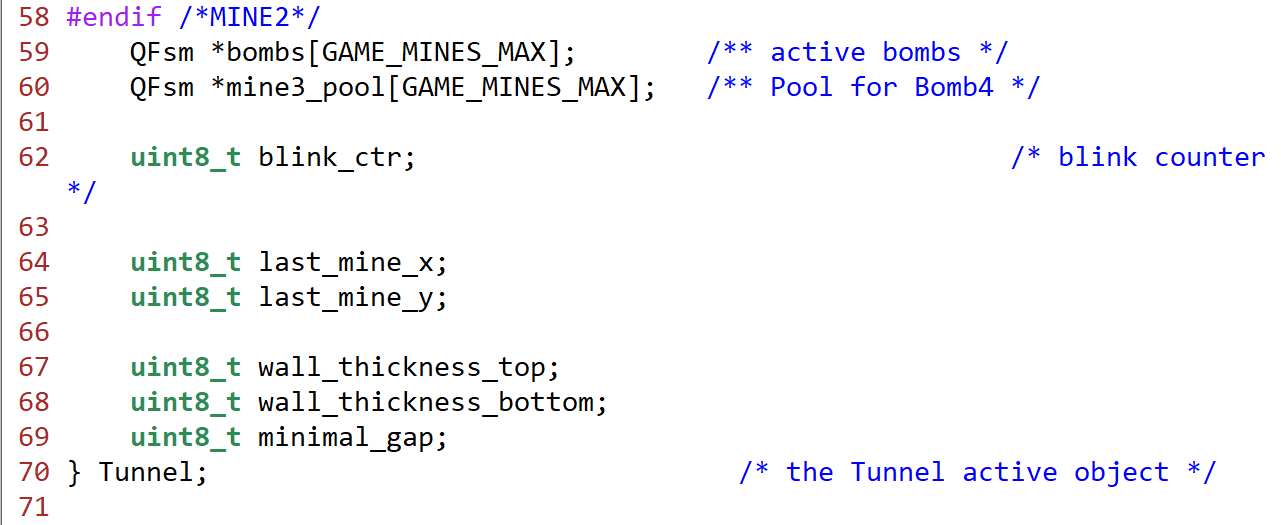
En el archivo main.c, se tiene el llamado a la función Tunnel\_ctor(). En la función Tunnel\_ctor() se agregó el llamado a la función Bomb4\_ctor(). Se incluye la función Tunnel\_ctor() (del archivo tunnel.c)





Los arreglos mine3\_pool y bombs se incluyeron en la declaración de la estructura objeto activo Tunnel (archivo tunnel.c)





## Inicialización de la máquina de estado finito QFsm

Se inicializa la máquina de estado finito llamando a la función

QFsm\_init(QFsm \*, QEvt \*)

Las máquinas de estado finito del objeto activo AO\_tunnel,

typedef struct TunnelTag {

QActive super;

// . . .

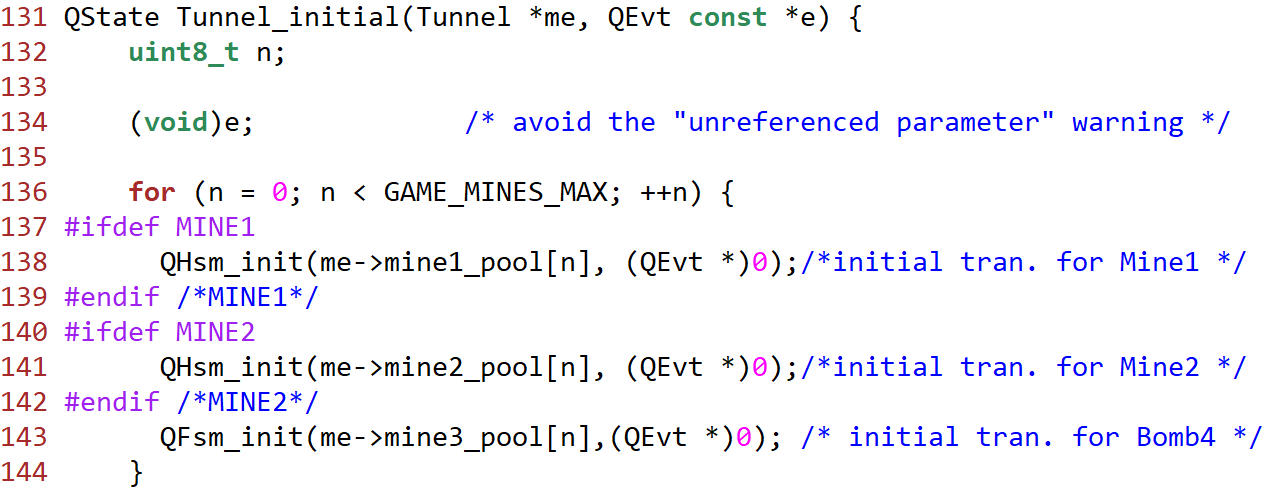
QFsm \*mine3\_pool[GAME\_MINES\_MAX];

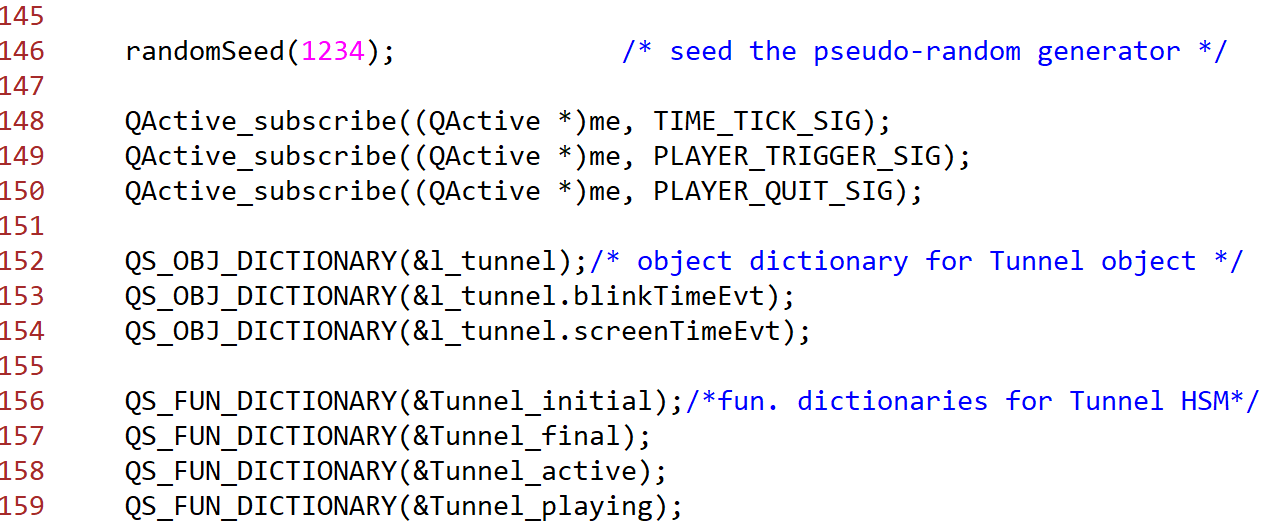
// . . .

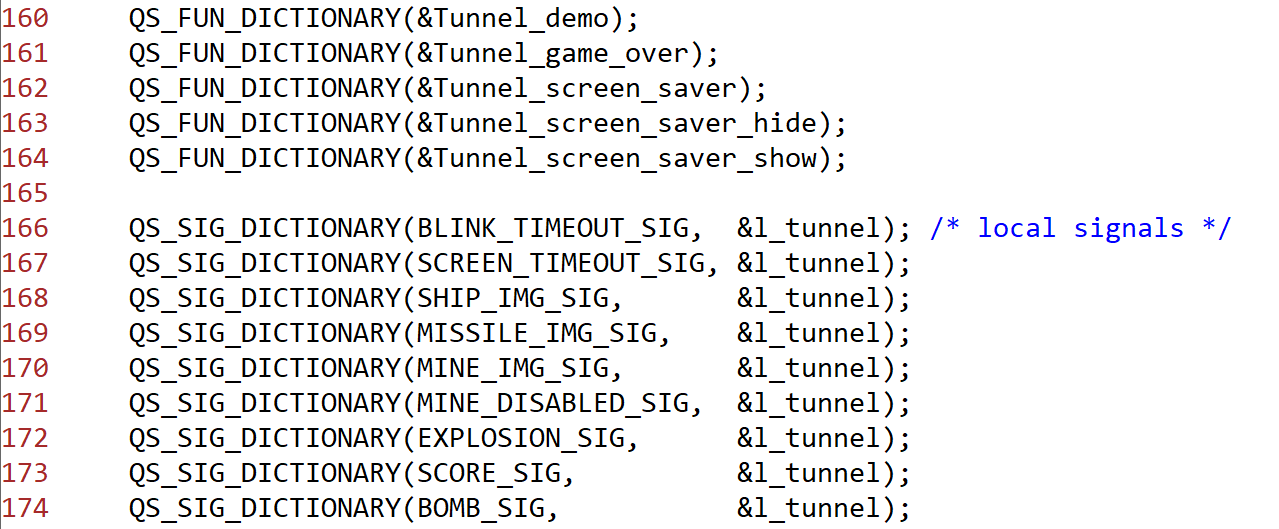
} Tunnel;

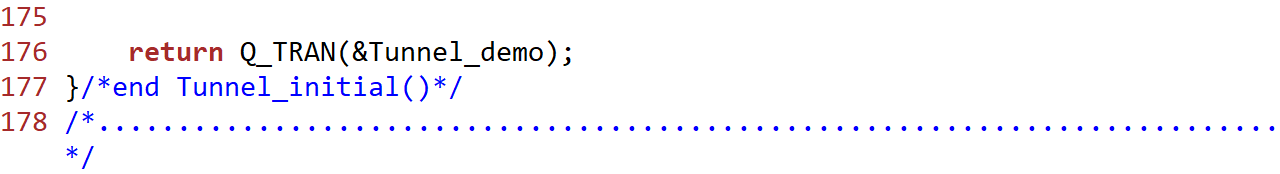
que previamente fueron construidas en la función Tunnel\_ctor(), en la línea 119.

La siguiente modificación, también es el archivo tunnel.c, en la función Tunnel\_initial(), ahí se inicializaron los elementos del arreglo me->mine3\_pool y se agregó la señal BOMB\_SIG al diccionario de señales para el objeto activo l\_tunnel. (líneas 143 y 174, archivo tunnel.c)

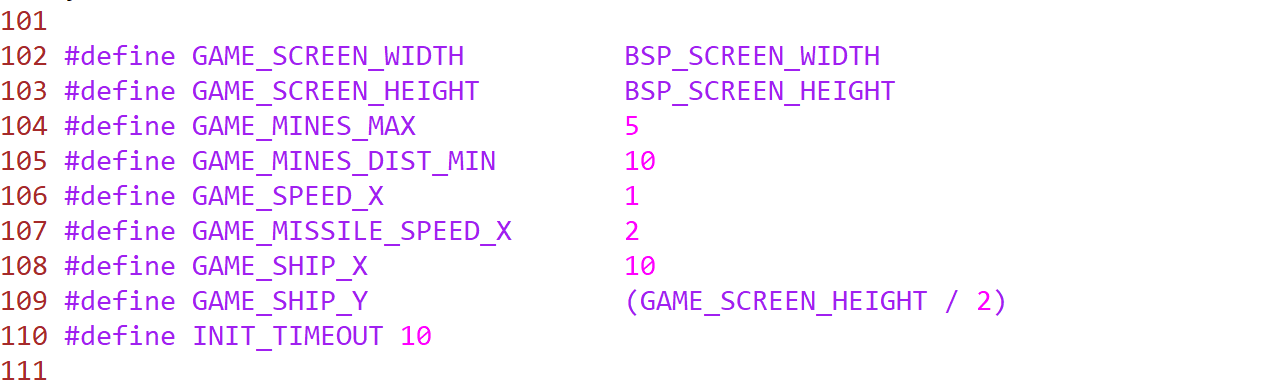




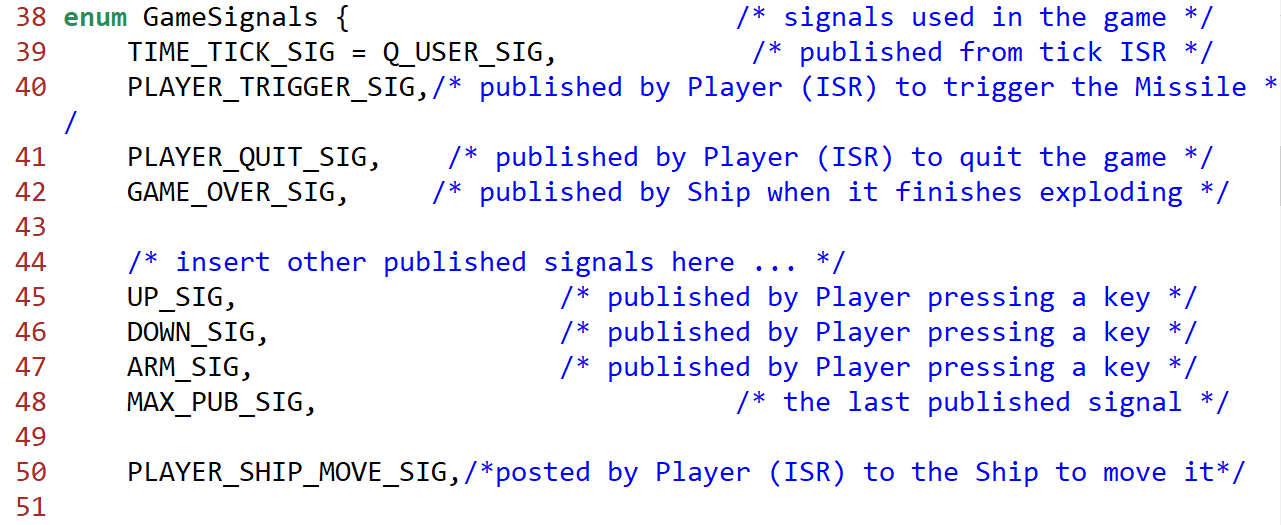


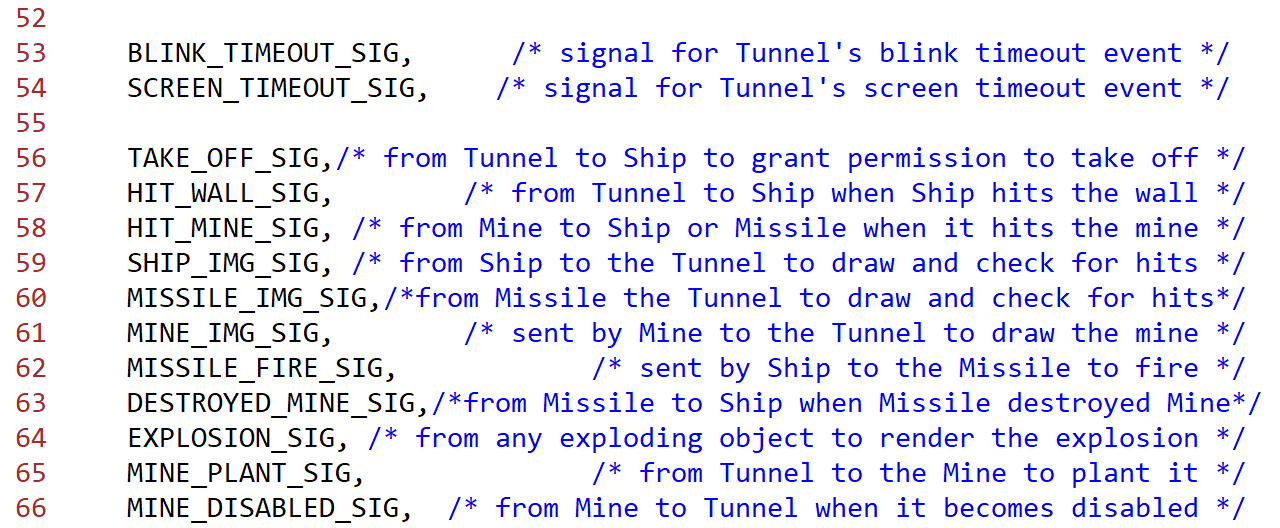


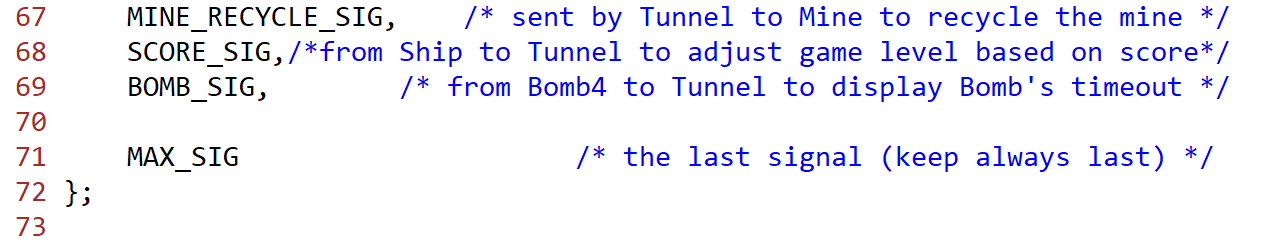
En este archivo se agregaron las líneas 143 y 174. La macro GAME\_MINES\_MAX está definida en el archivo game.h y está definida como 5:



La señal BOMB\_SIG se agregó en el archivo game.h como se muestra a continuación (archivo game.h)

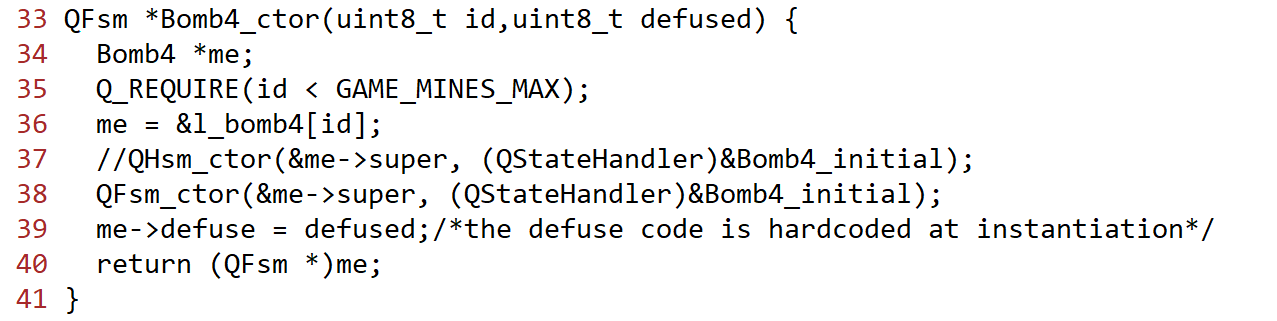






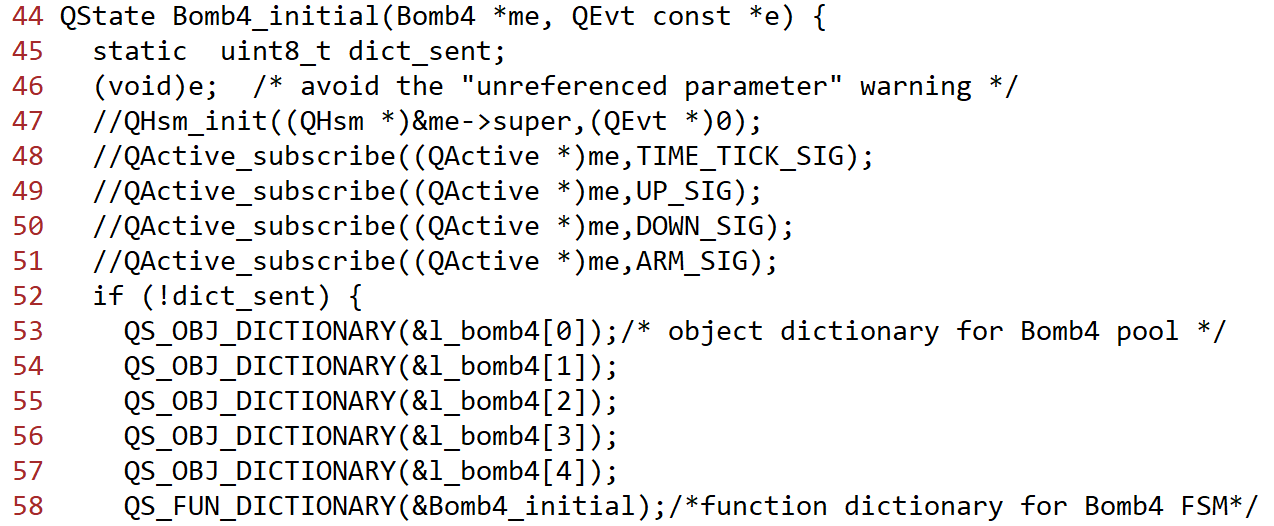
Además de la señal BOMB\_SIG (línea 69), en esta enumeración se agregaron también las señales UP\_SIG, DOWN\_SIG y ARM\_SIG en las líneas 15,46 y 47 respectivamente.

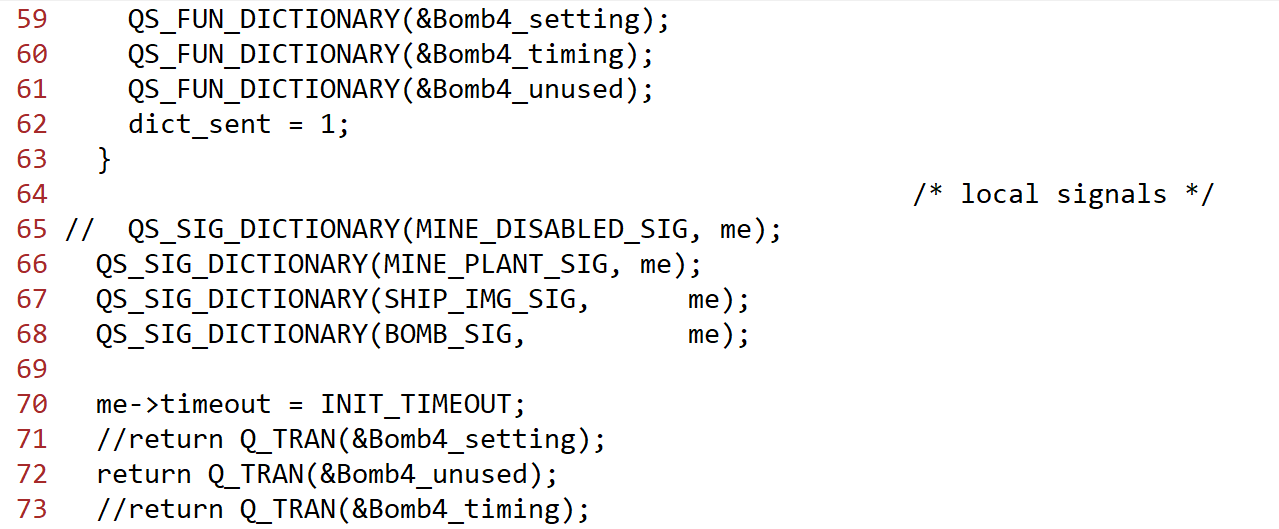
Ahora bien, la siguiente función es el constructor Bomb4\_ctor() (archivo bomb4.c), el cual es llamado en la función Tunnel\_ctor() del archivo tunnel.c.

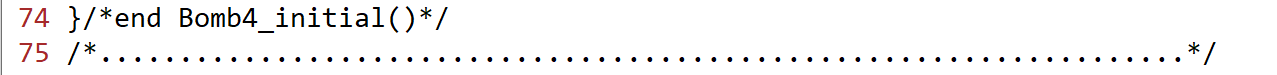


Función Bomb4\_ctor(), (Archivo bomb4.c)

En esta función, en la línea 38 se establece a la función Bomb4\_initial como manejadora de estado (el estado Bomb4\_initial). La función Bomb4\_initial() (archivo bomb4.c) se incluye a continuación:

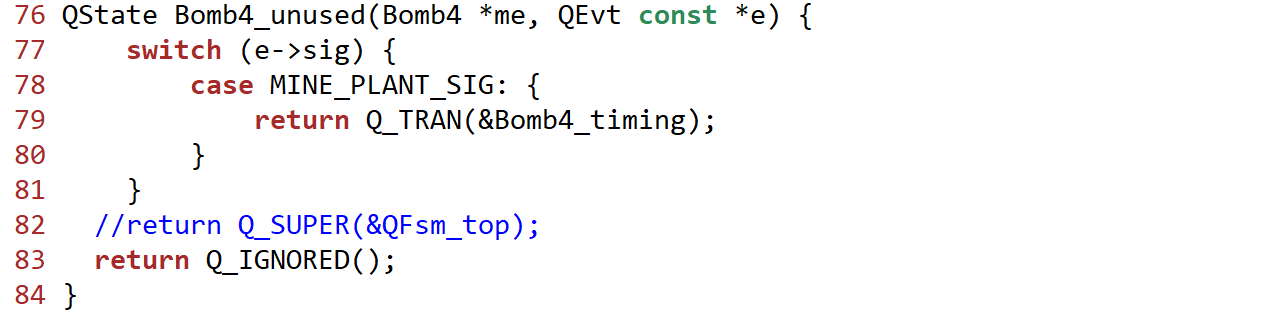






Función Bomb4\_initial(), archivo bomb4.c

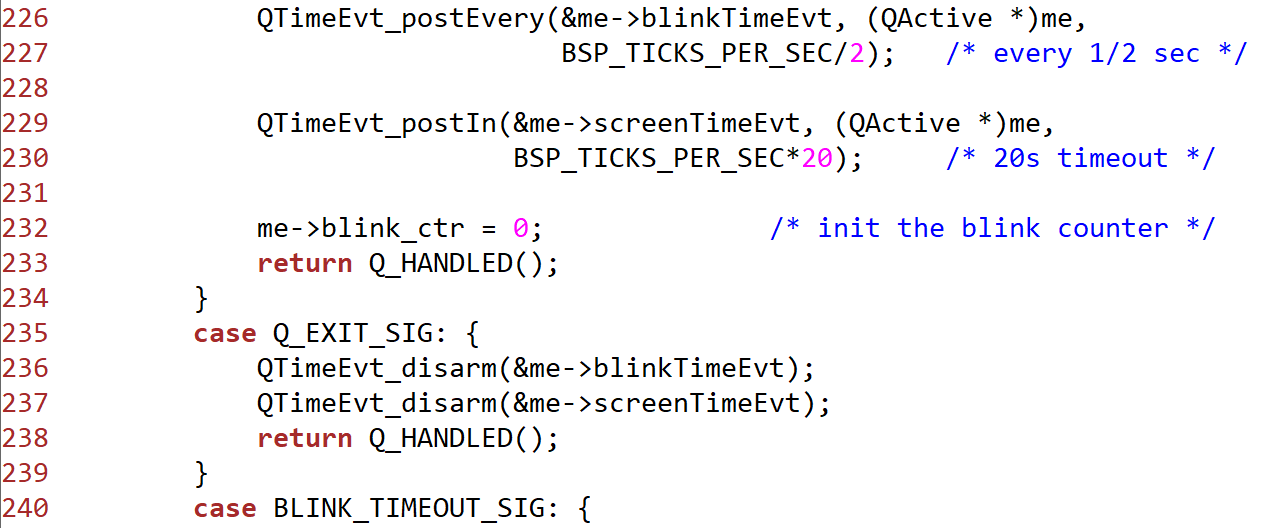
En esta función se agregaron las líneas 61, 66 y 72. En ese mismo archivo bomb4.c, se agregó la función Bomb4\_unused(), que incluyo a continuación:



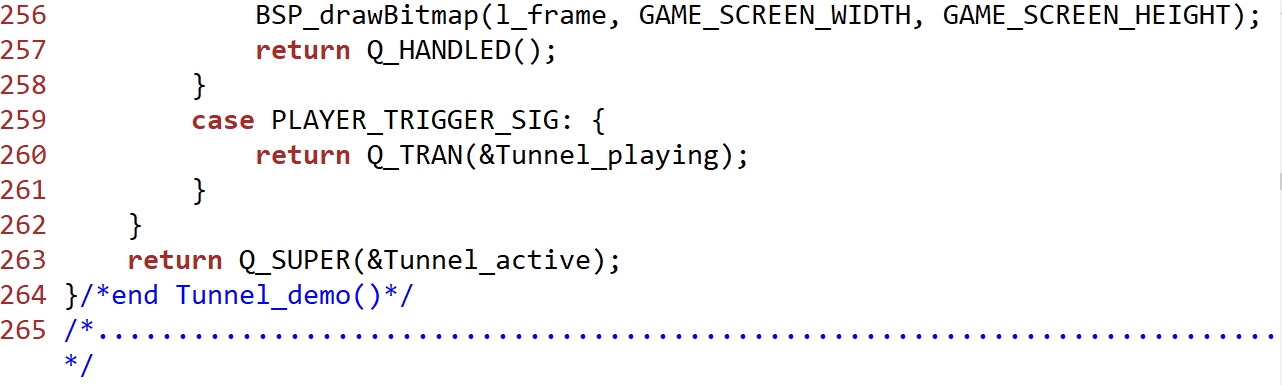
Función Bomb4\_unused(), archivo bomb4.c

Para que esta función (Bomb4\_unused()) se ejecute, primero se ejecuta la función Tunnel\_demo(), cuando se está ejecutando la función Tunnel\_demo() –cuando aparece el mensaje PRESS BUTTON-en letras grandes-, si el usuario presiona la tecla E, en la función Tunnel\_demo(), se gestionará la señal PLAYER\_TRIGGER\_SIG (véase la línea 259 de la función Tunnel\_demo() que se incluye a continuación)



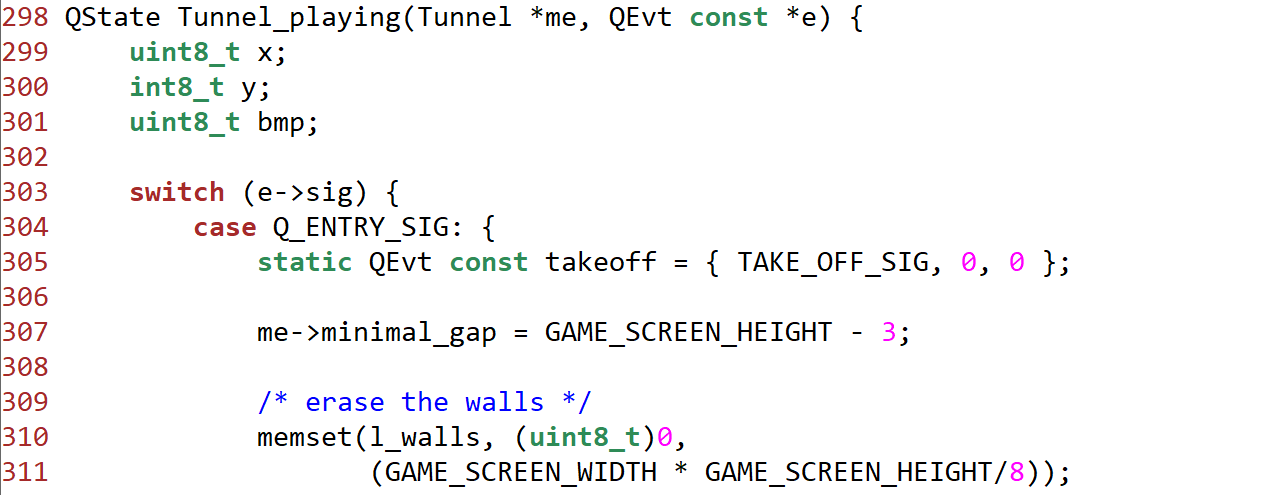




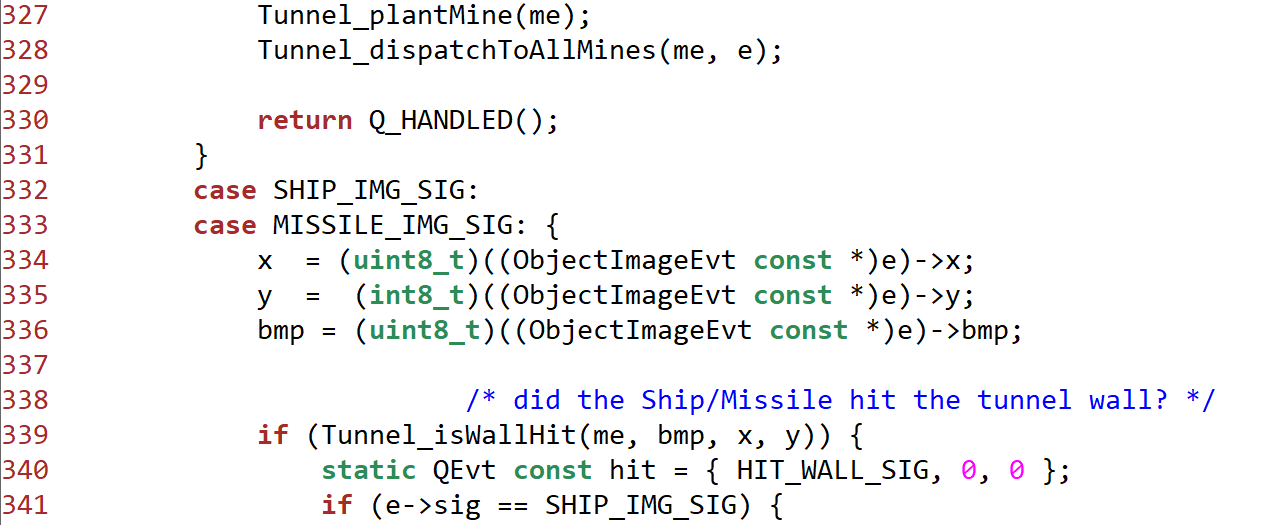


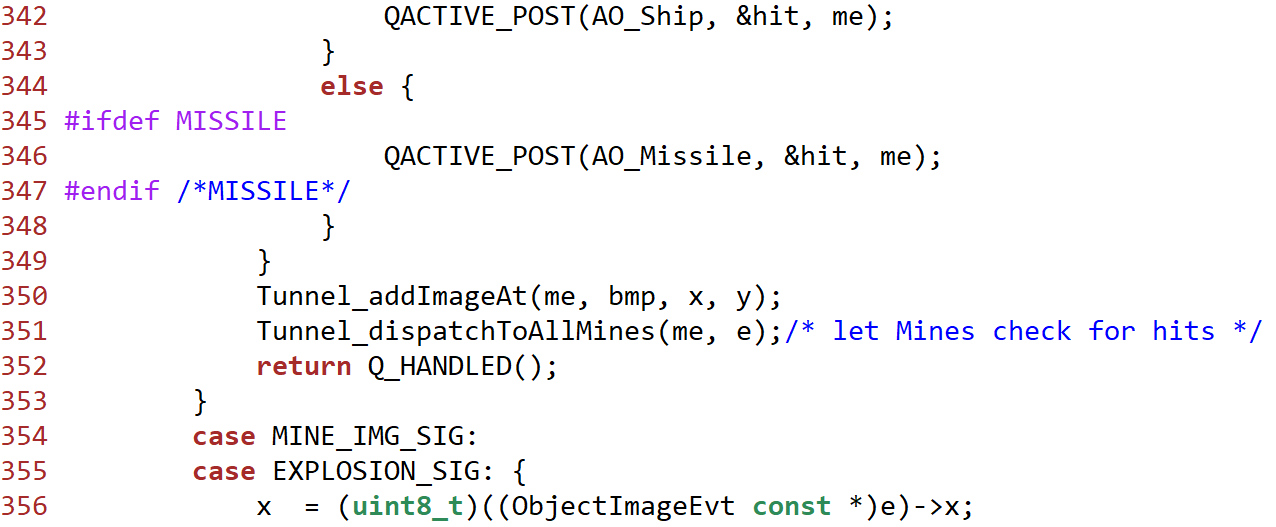
Función Tunnel\_demo(), archivo tunnel.c

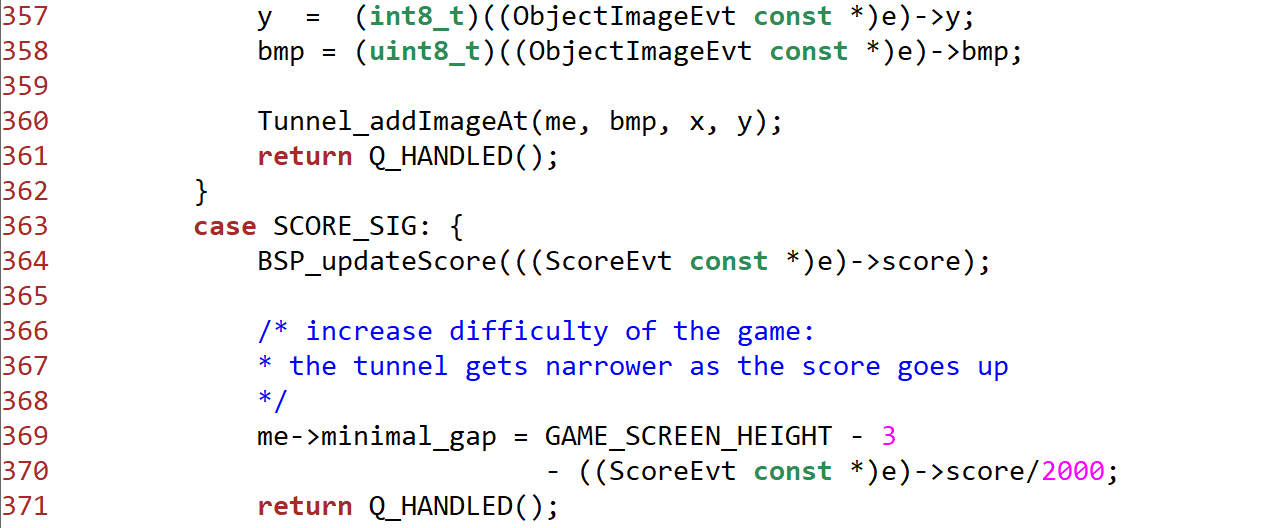
Cuando el usuario presiona la tecla e, se realiza una transición de estado (línea 260) al estado Tunnel\_playing. En la función Tunnel\_playing() se hicieron algunas modificaciones. La función Tunnel\_playing se incluye a continuación:

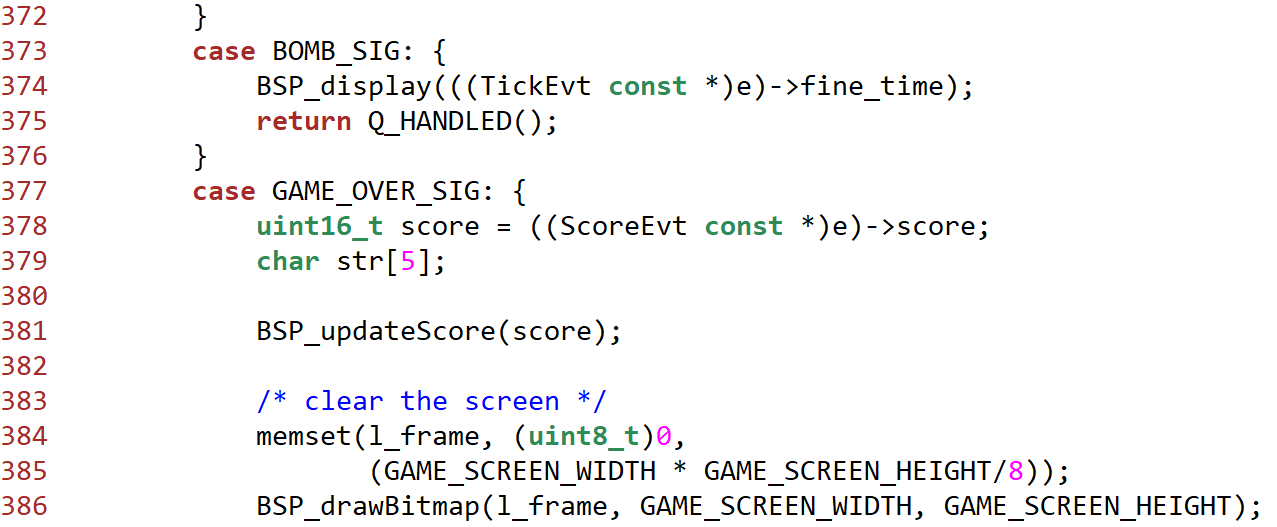


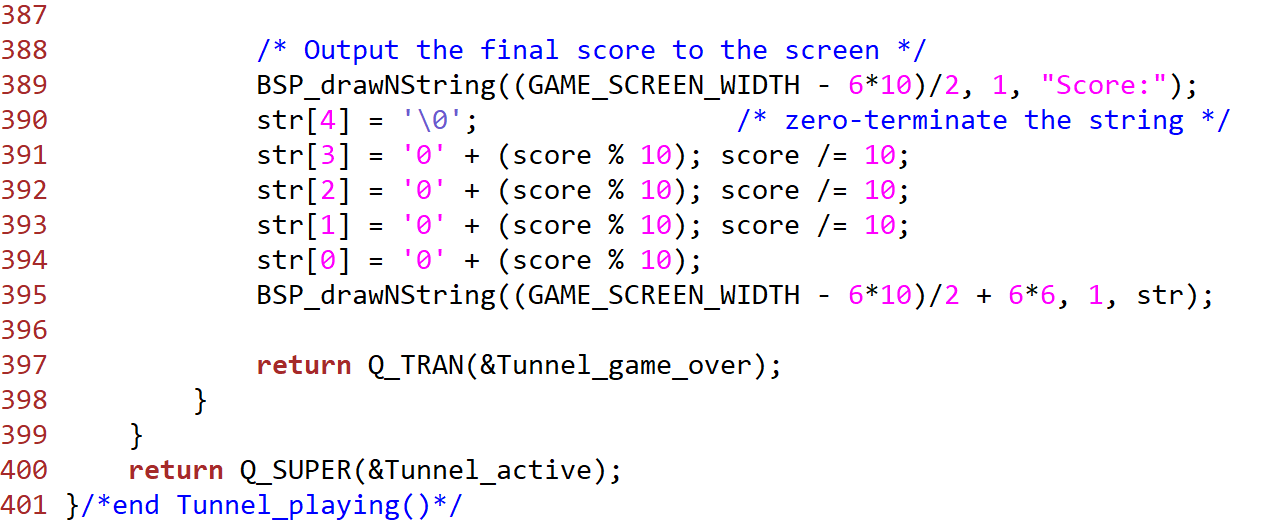






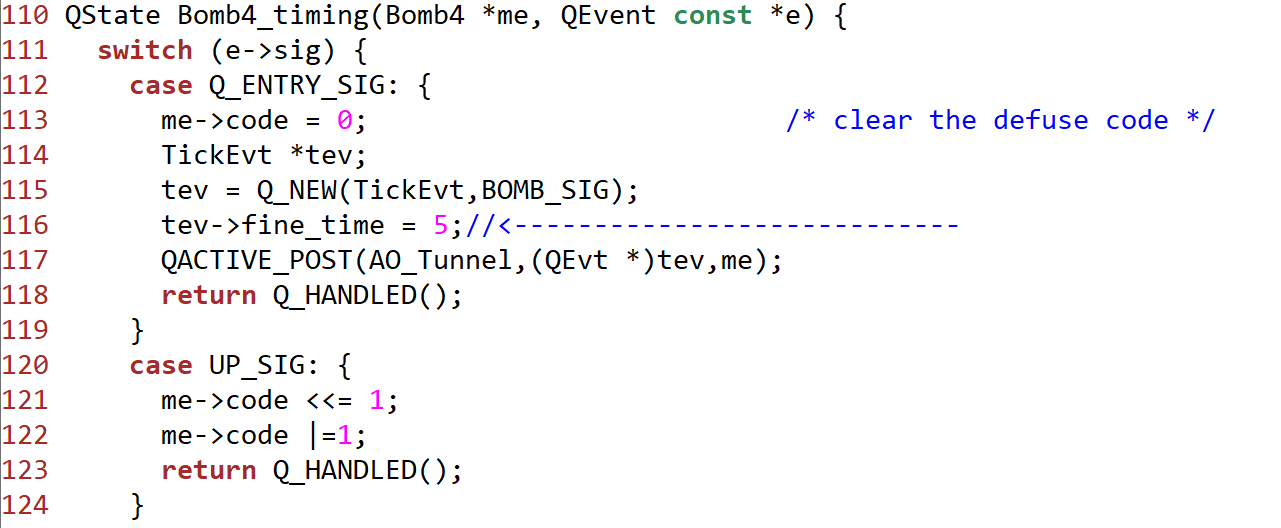


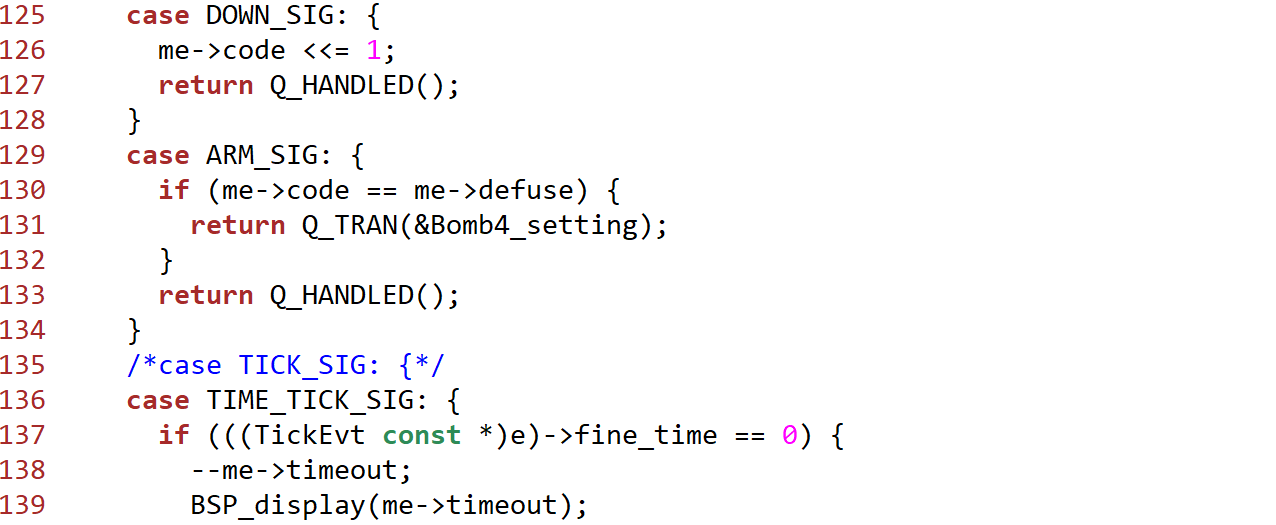


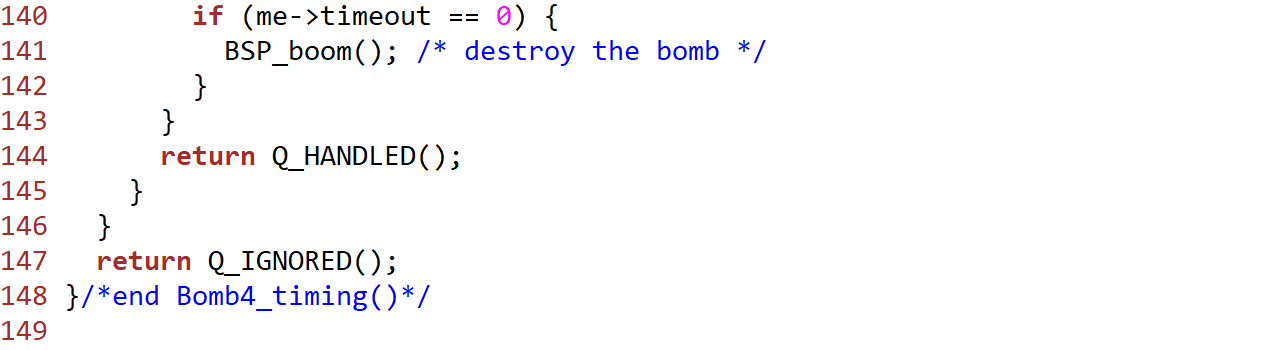


Se modificó la función Tunnel\_playing en el archivo tunnel.c (se agregaron las líneas 373, 374, 375 y 376). Con estas tres líneas, cuando el túnel esté en el estado Tunnel\_playing y reciba la señal BOMB\_SIG, por el llamado a la función BSP\_display(), imprimirá en la pantalla (cerca de la esquina inferior derecha de la pantalla) el valor del atributo fine\_time del evento TickEvt que se habría recibido como argumento en el llamado a la función Tunnel\_playing(). El lugar en el código en donde se le envía el evento TickEvt al túnel es en la función Bomb4\_timing(), archivo bomb4.c.

La función Bomb4\_timing() se incluye a continuación:





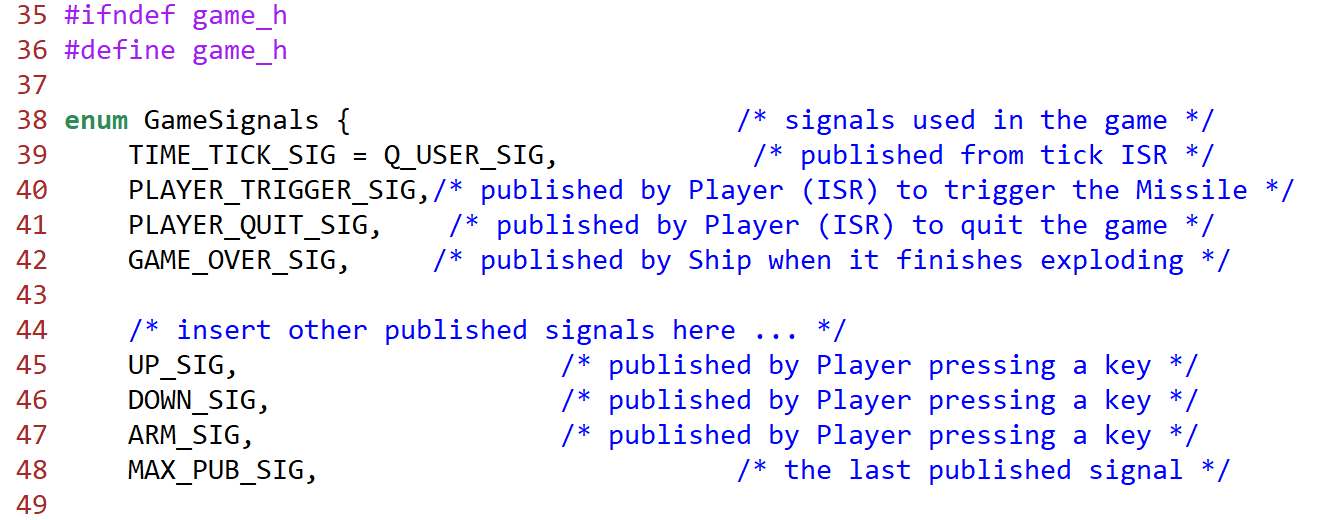


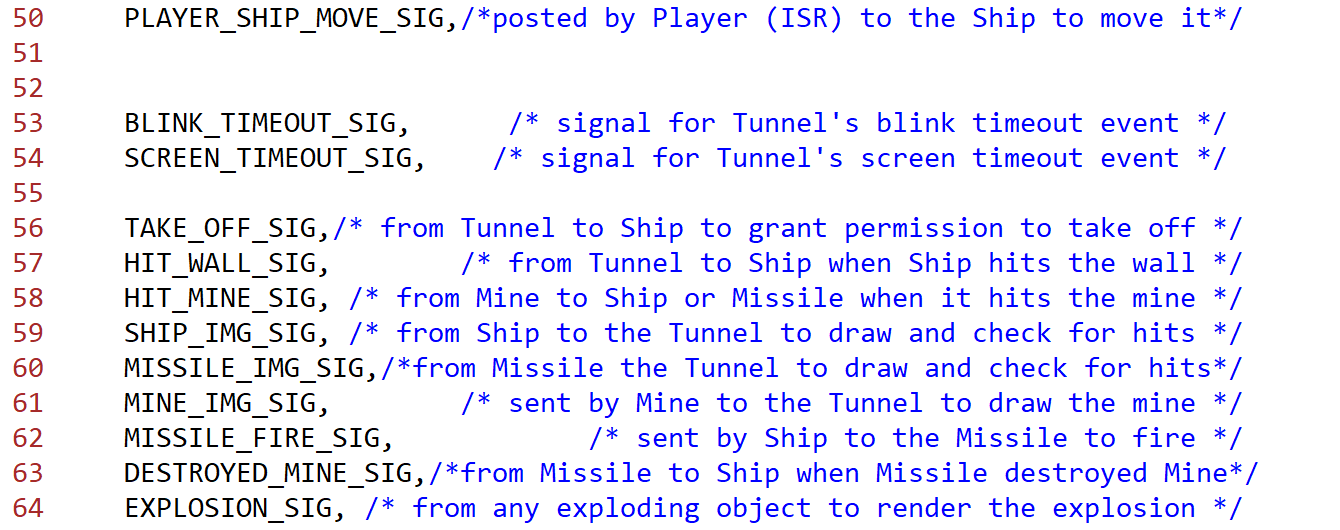
void Tunnel\_plantMine(Tunnel \*me);

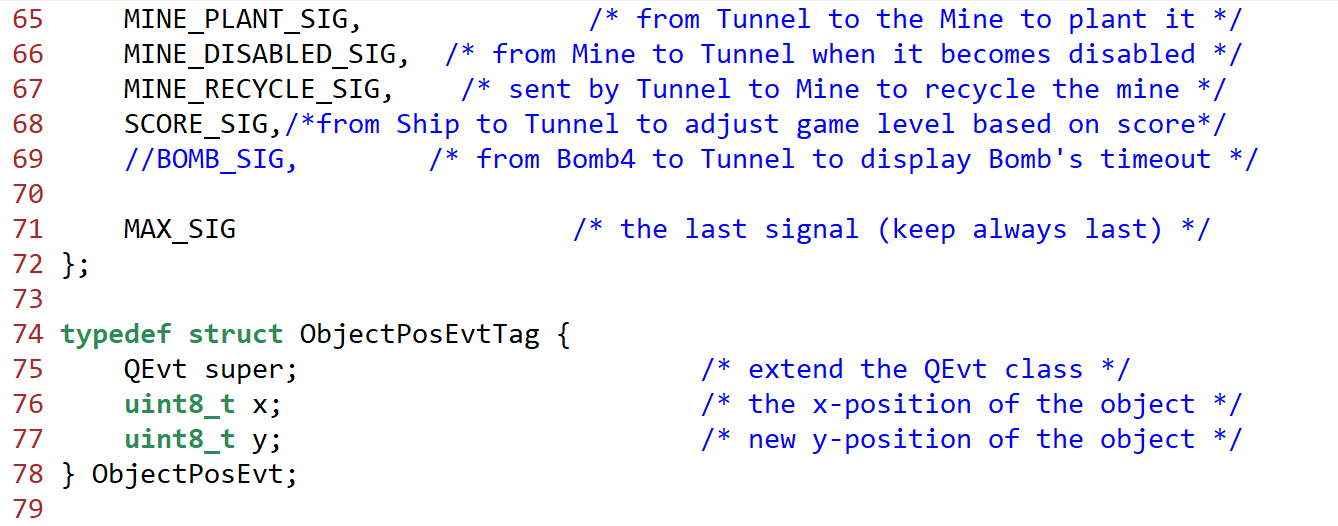
void Tunnel\_dispatchToAllMines(Tunnel \*me, QEvt const \*e)

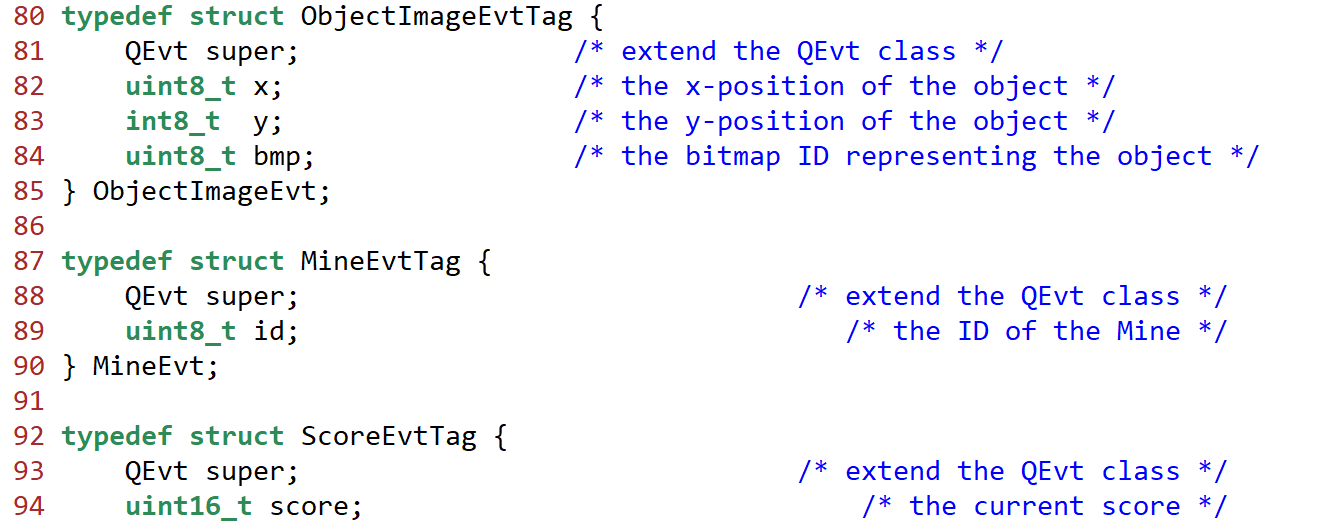
# Añadiendo el evento BombEvt para enviar la señal UP\_SIG a la máquina de estado finito Bomb4

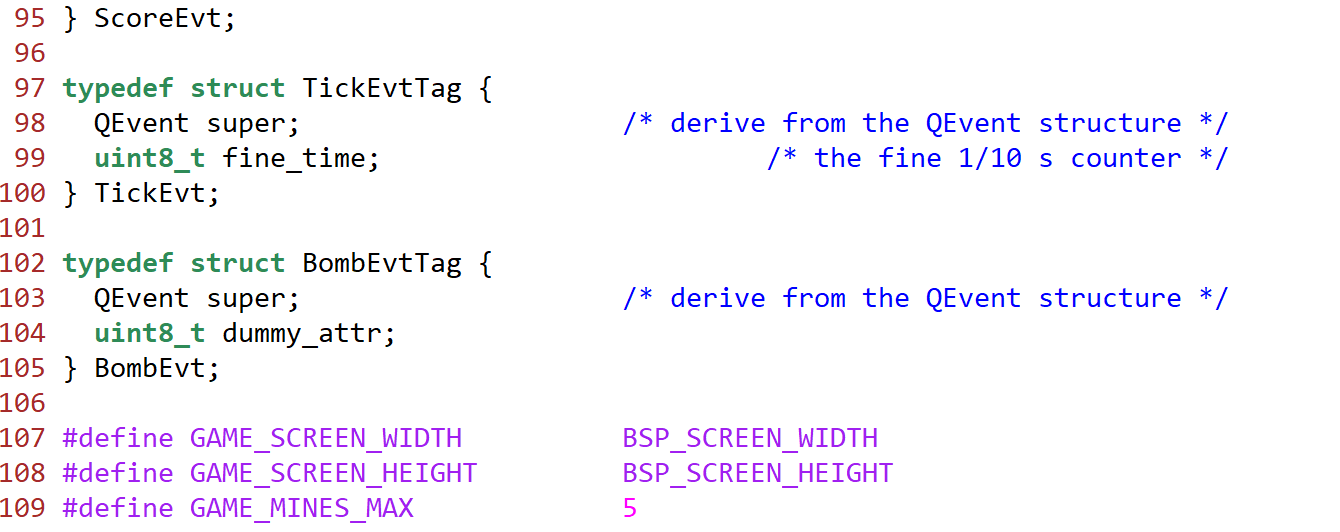
En el archivo game/game.h se incluye la señal UP\_SIG – en la enumeración de señales GameSignals --, y se agrega la estructura BombEvt.

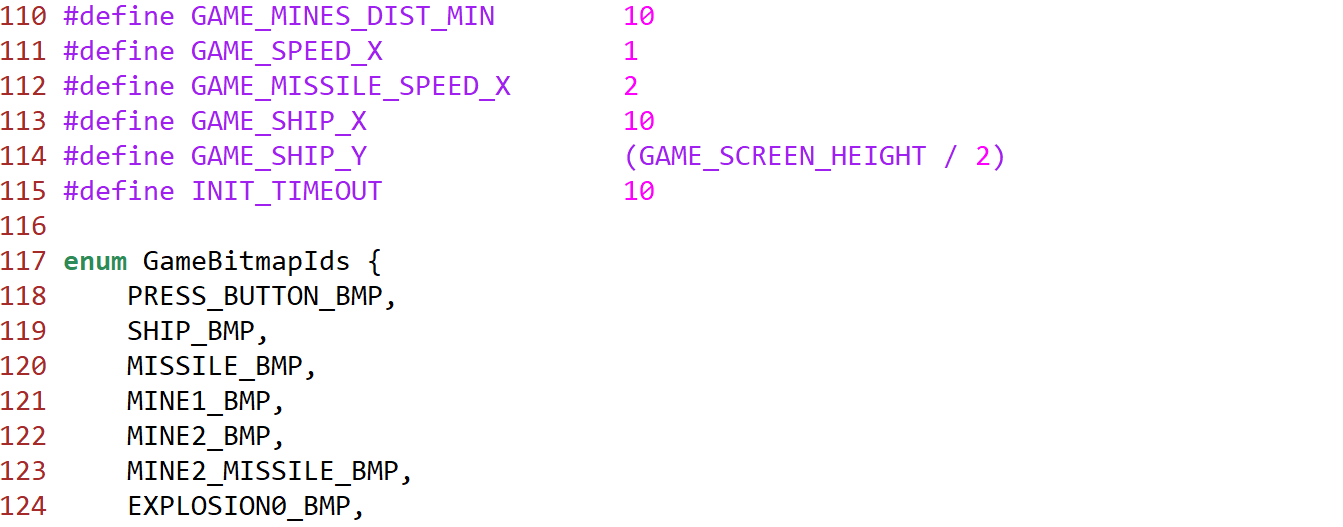


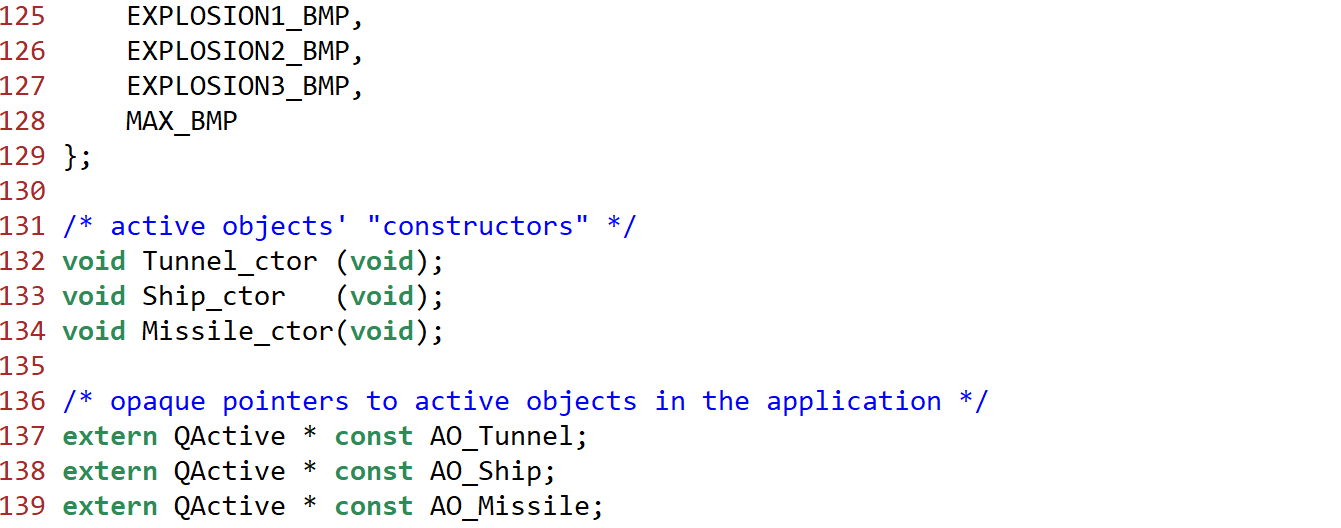


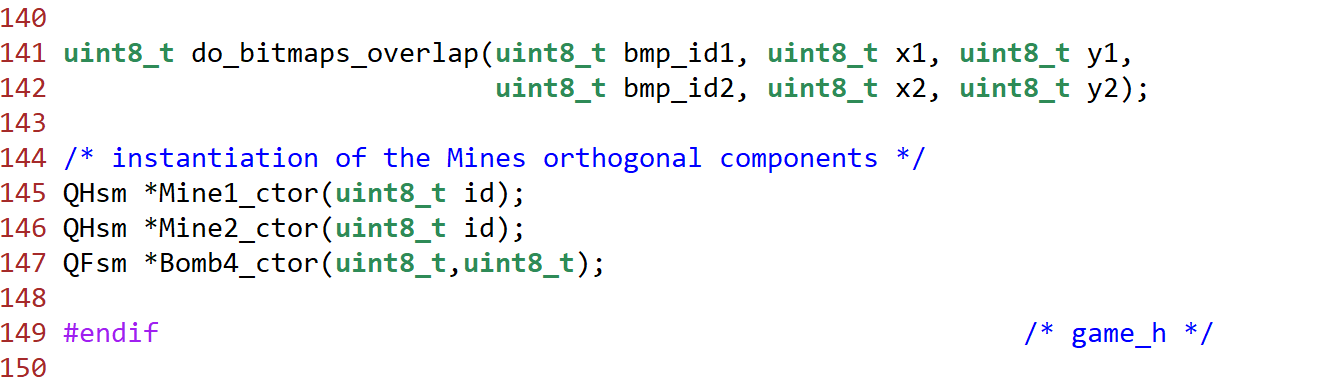








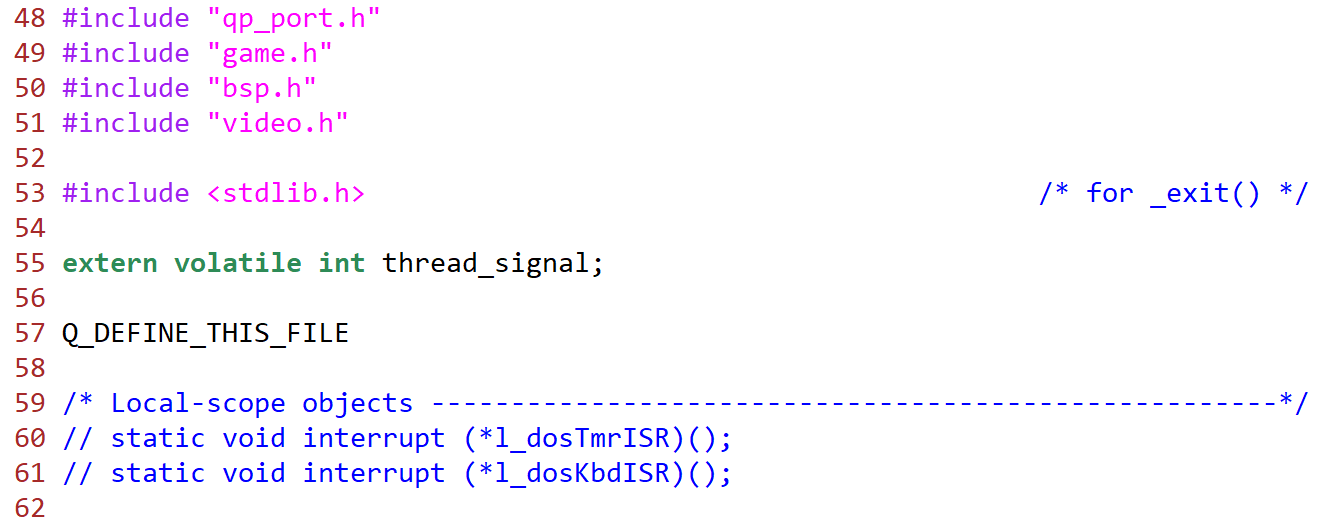


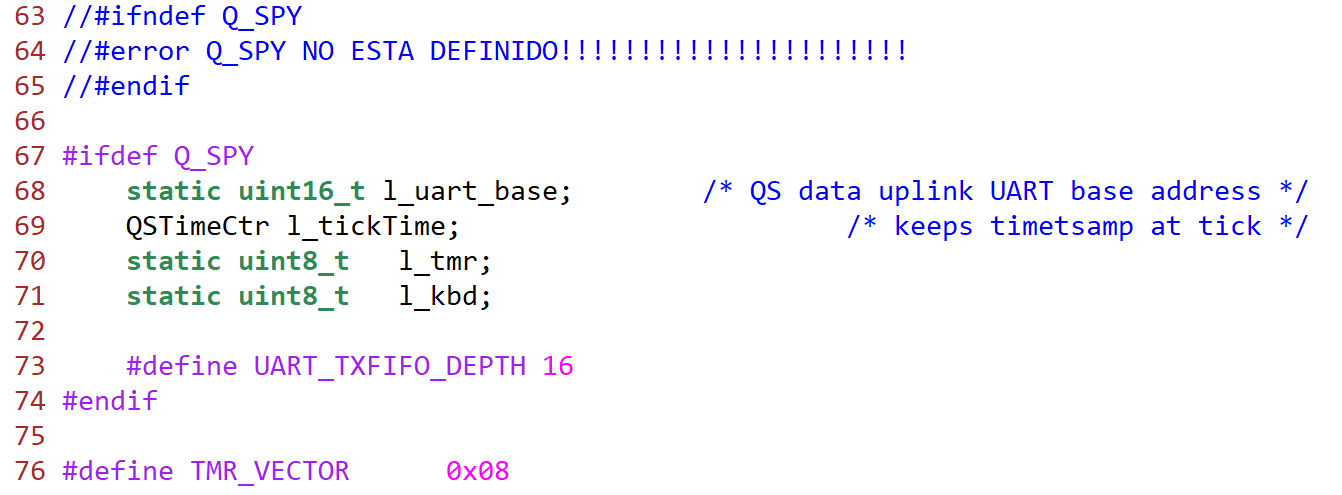


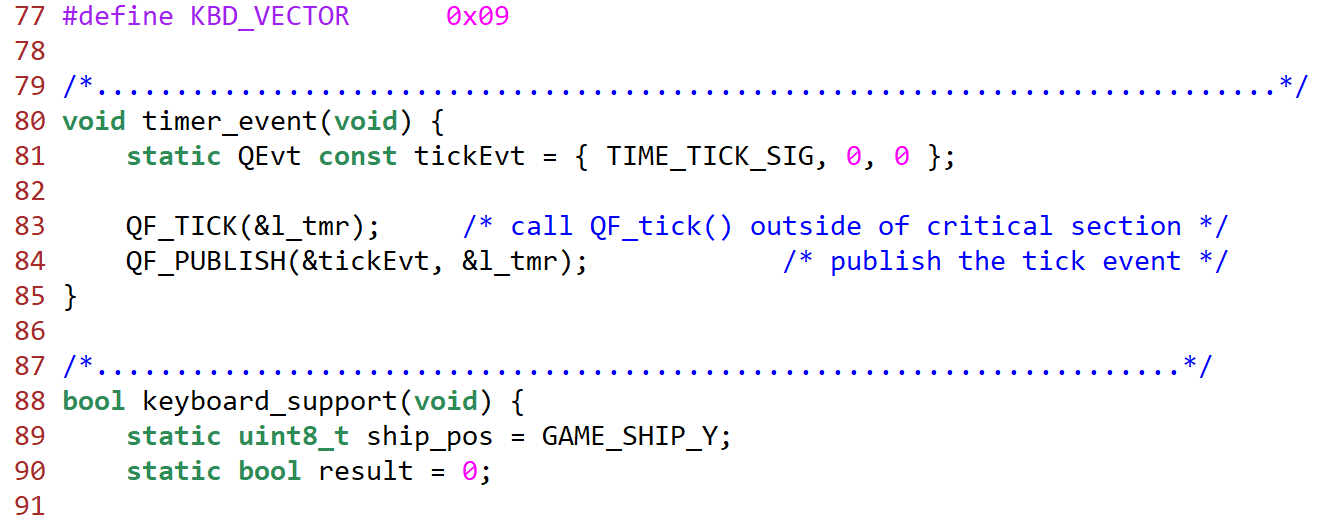
Archivo game/game.h

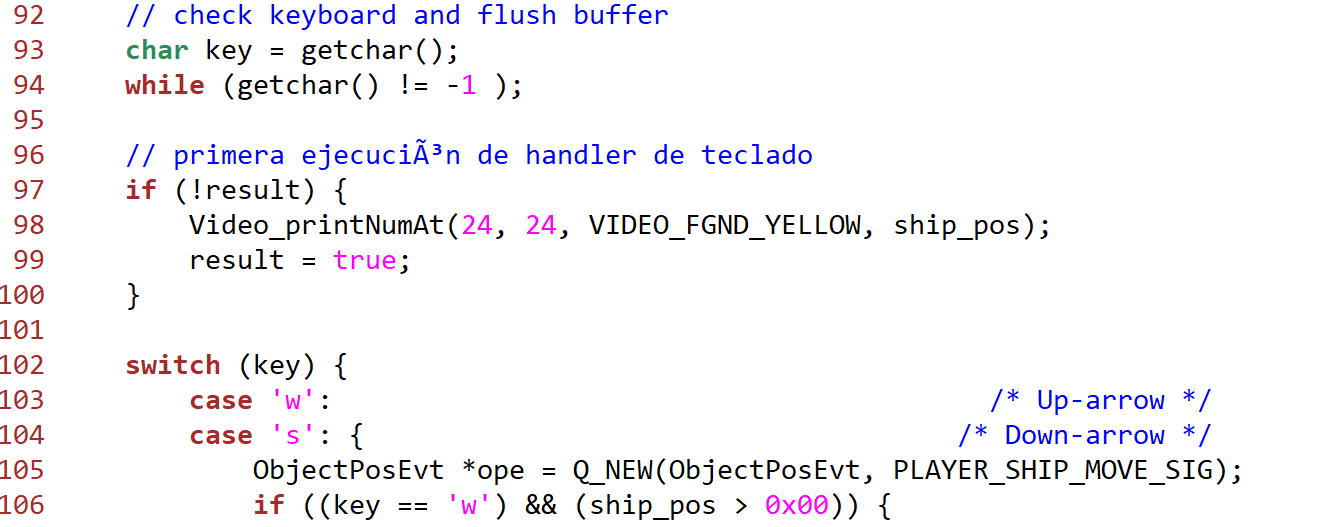
En el listado anterior, la señal UP\_SIG está en la línea 45. La estructura BombEvt está declarada en las líneas de la 102 a 105.

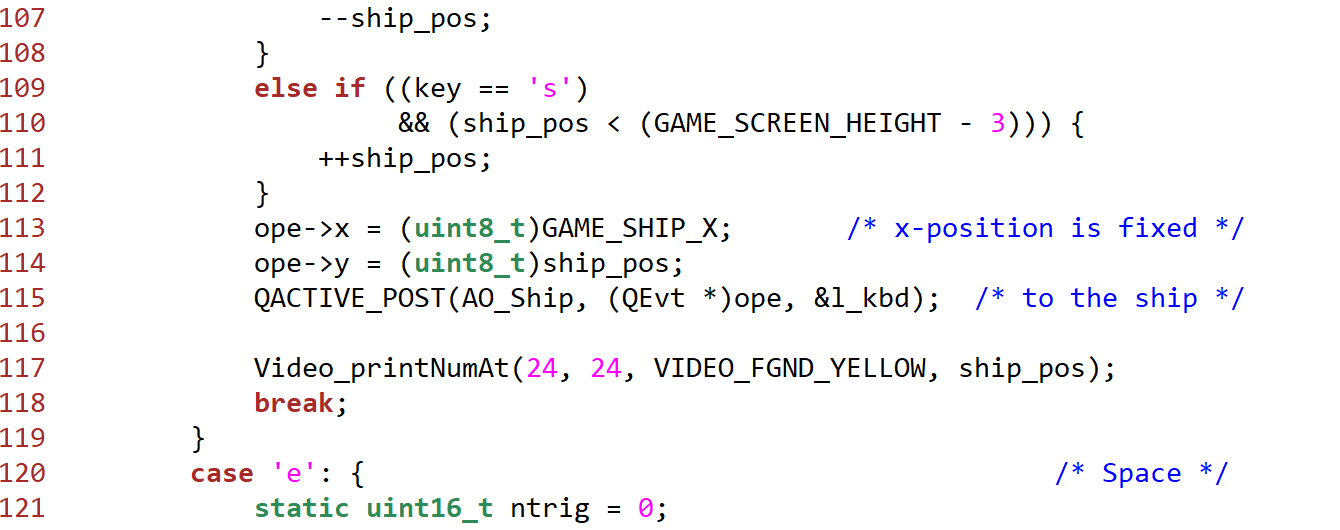
En el archivo bsp.c, en la función keyboard\_suppot(), se agrega el código para crear el evento BombEvt cuando se presione la tecla ‘u’, y enviar el evento creado al objeto activo AO\_Tunnel.

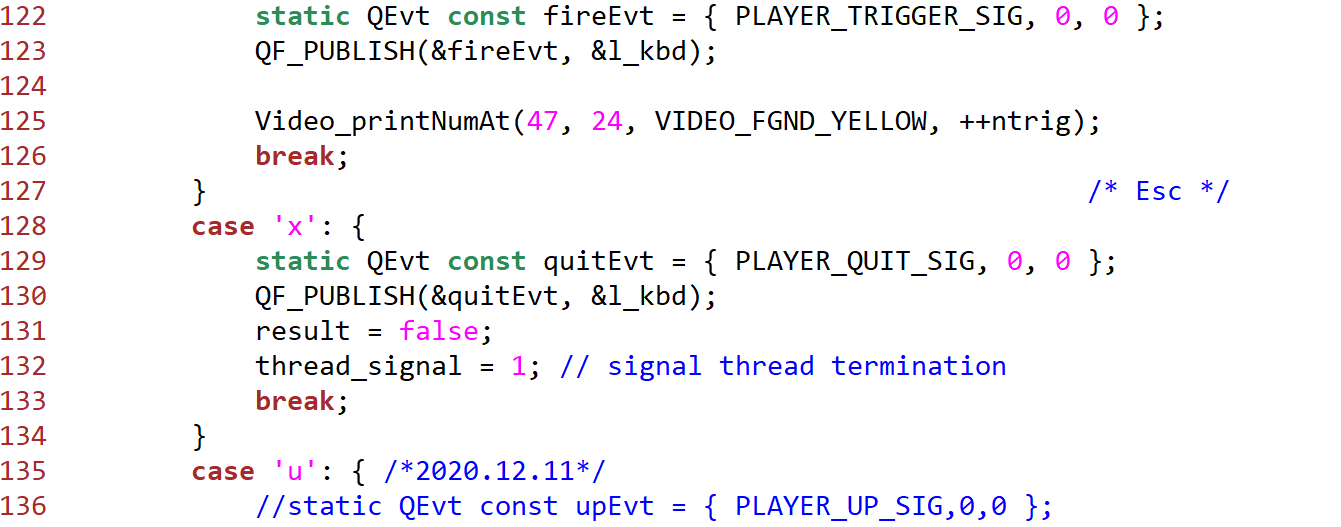


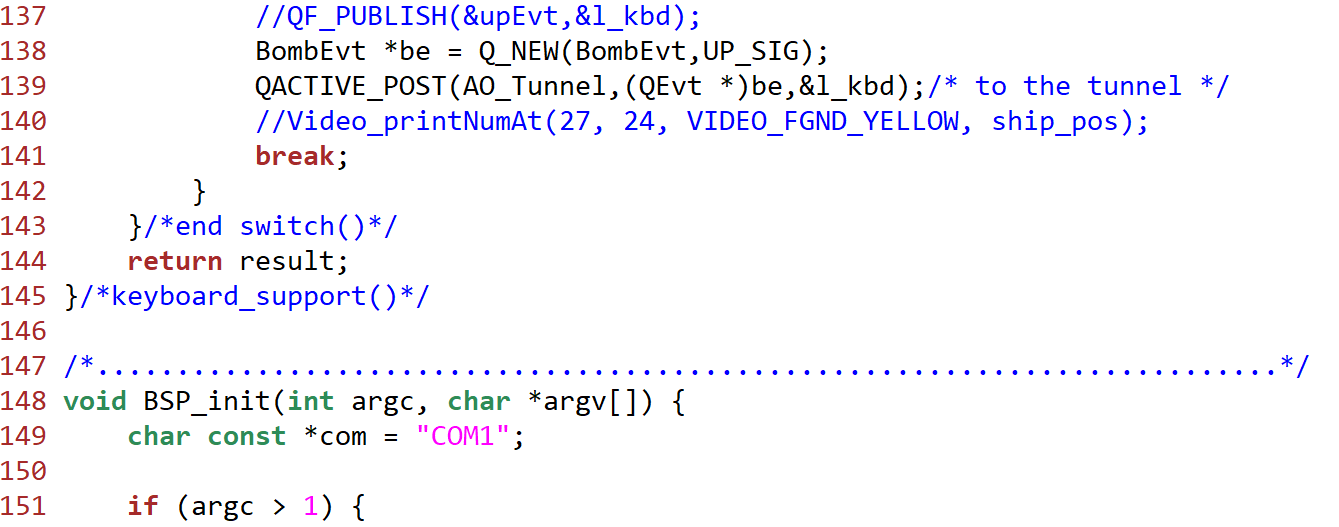


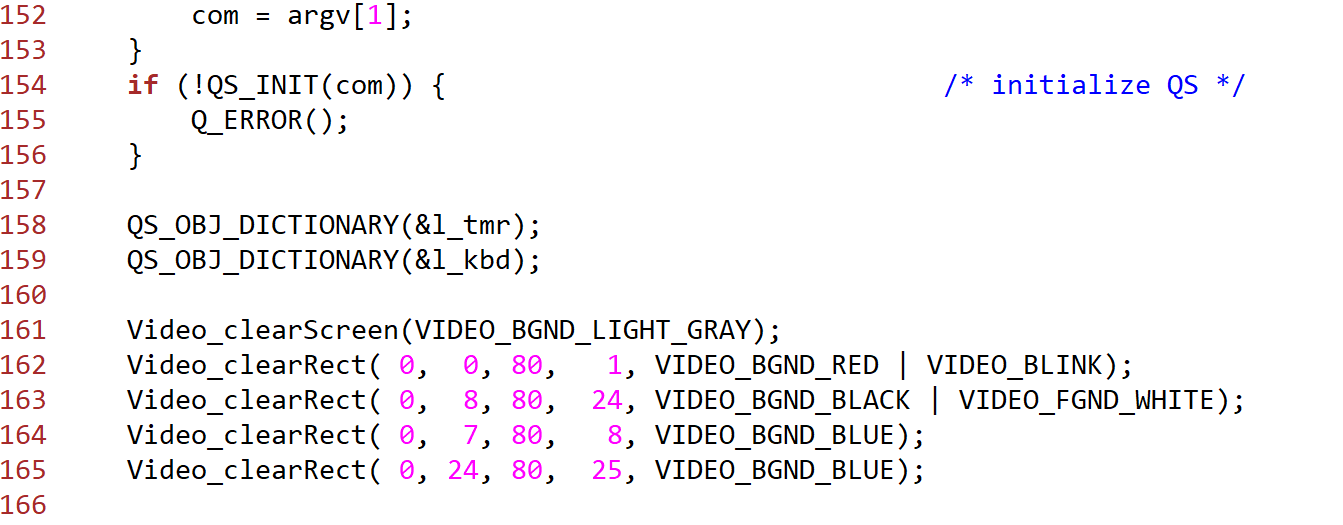


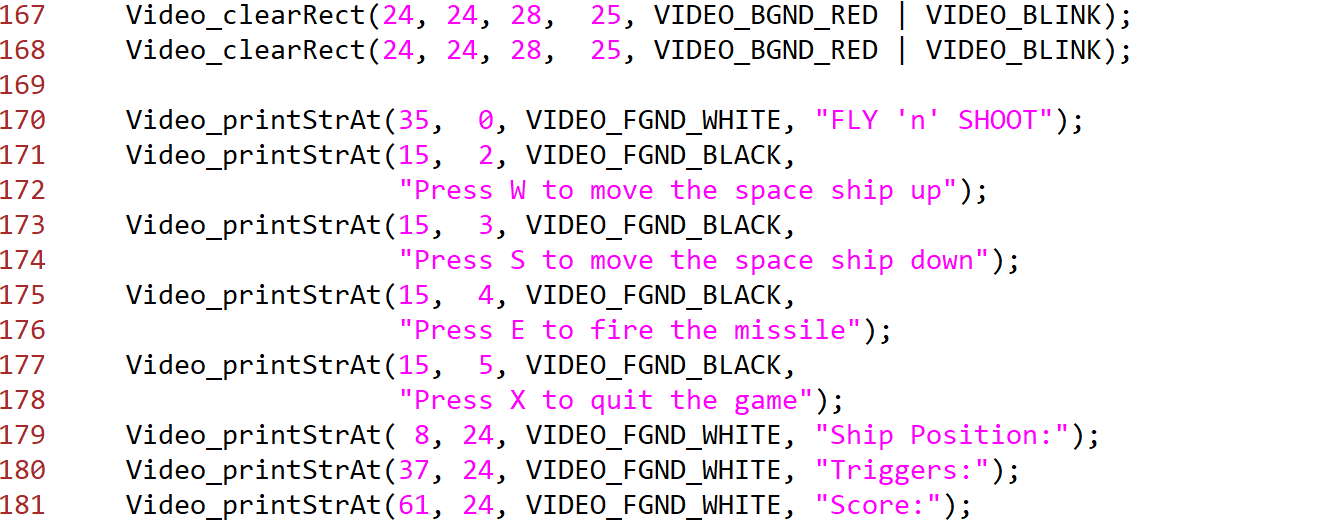


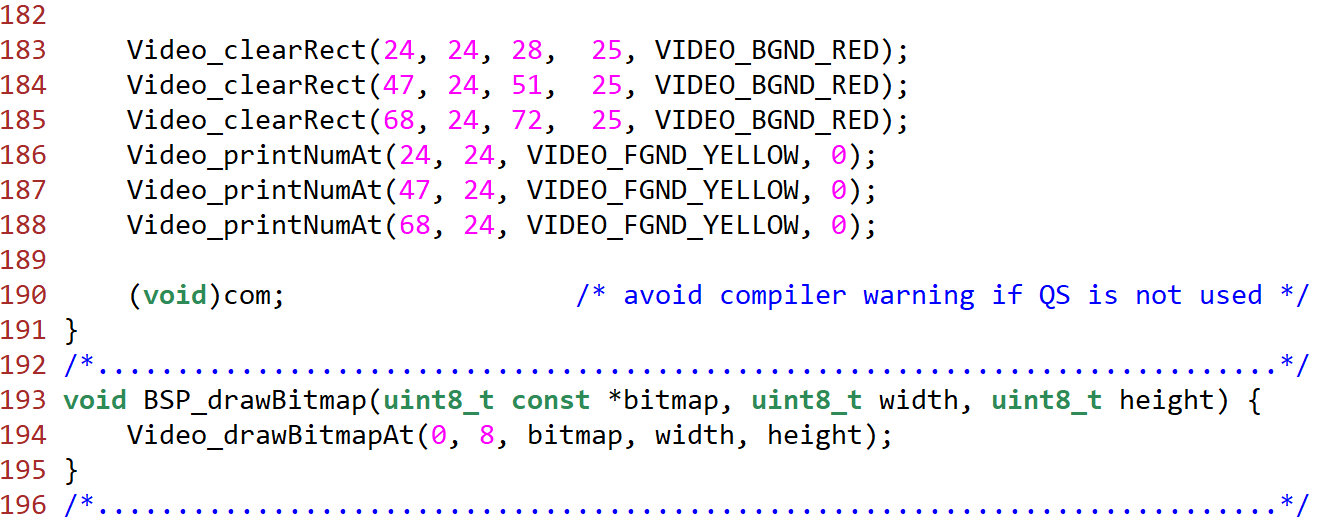


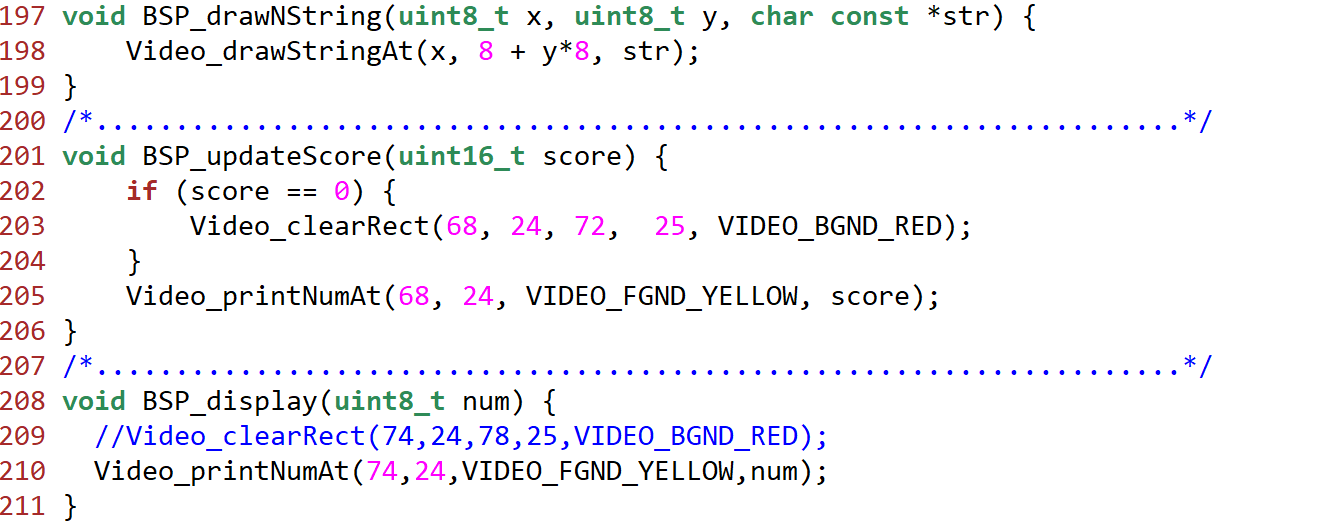


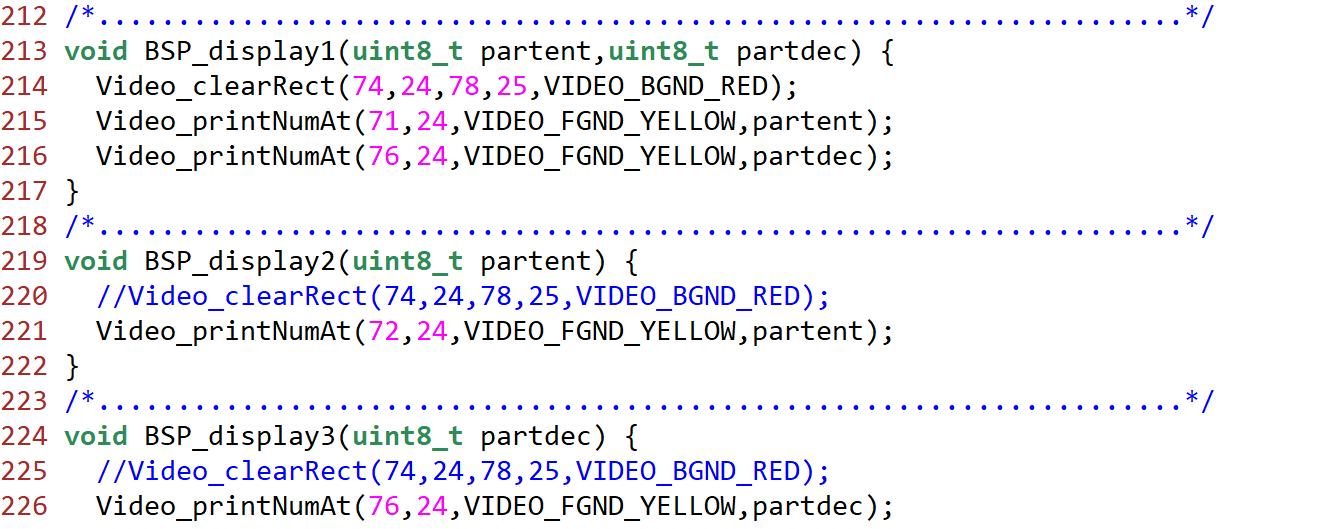


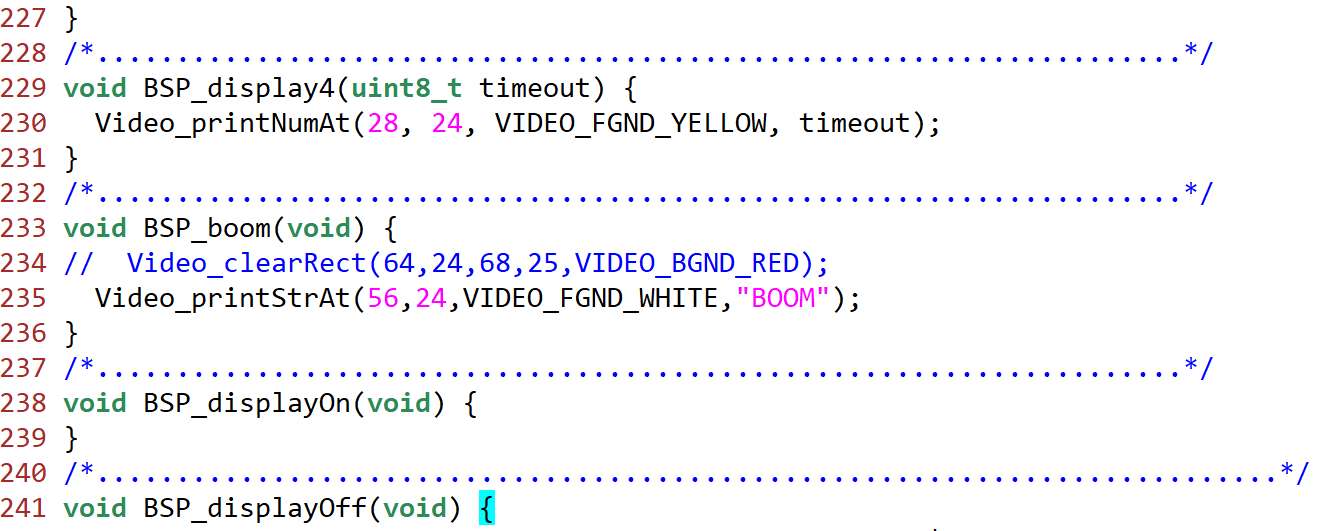


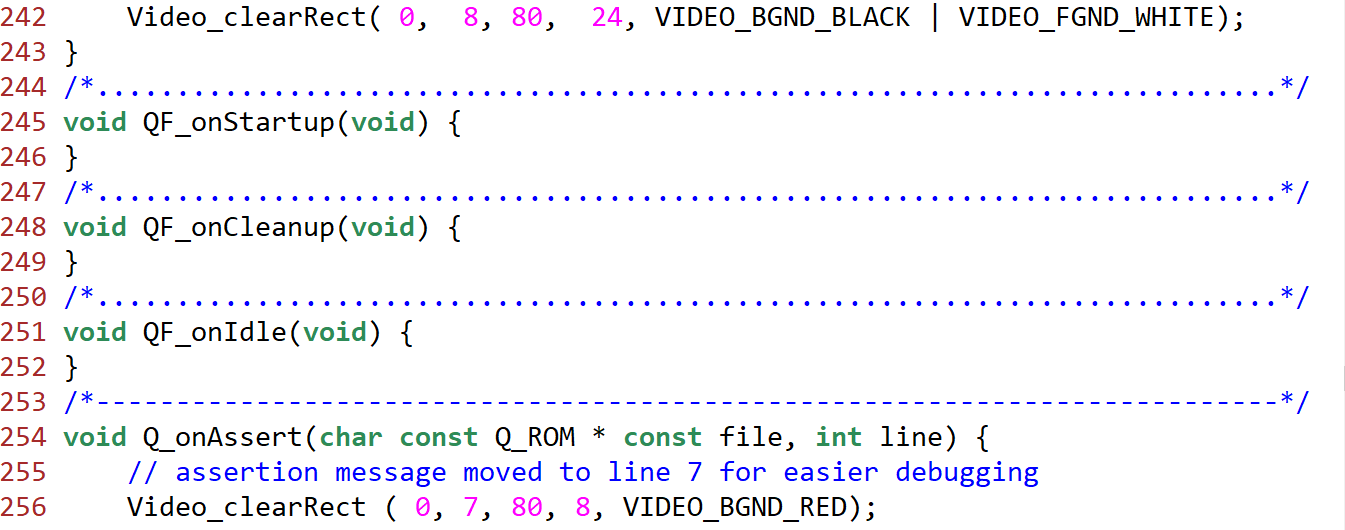


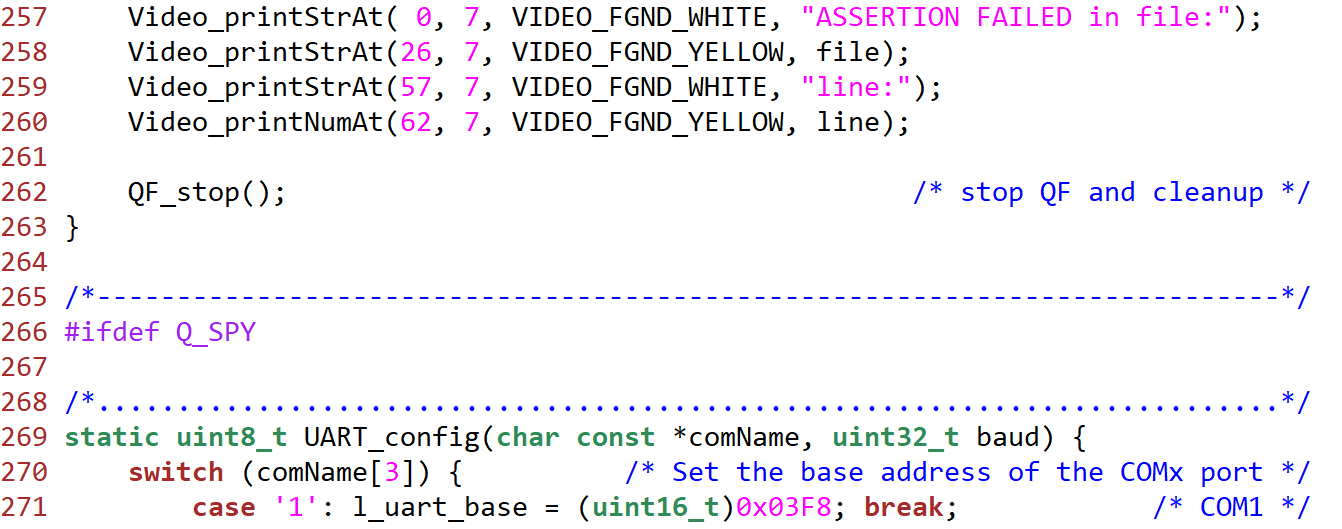


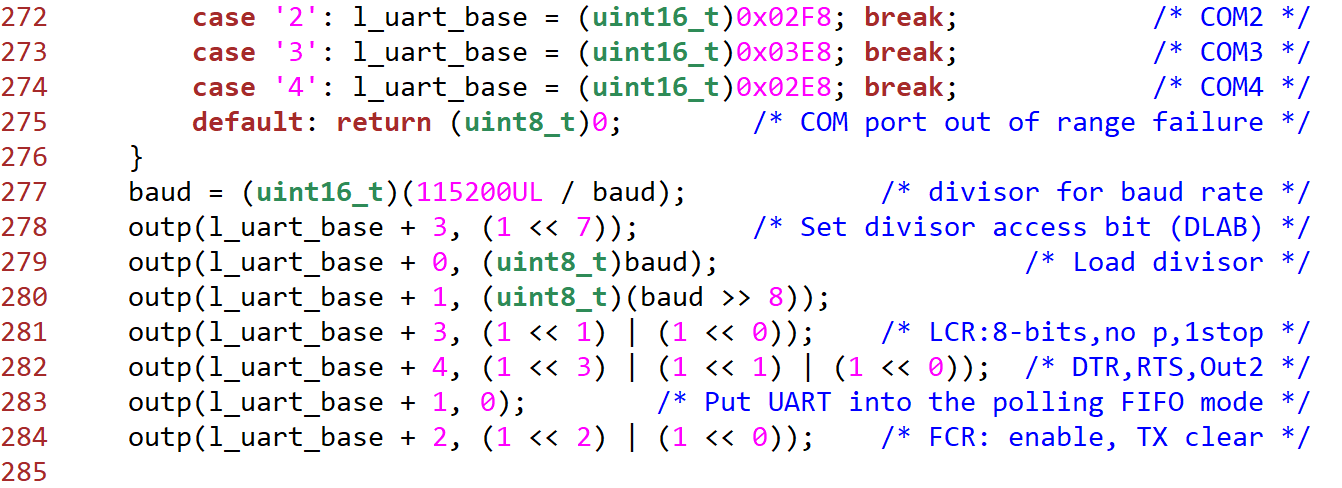


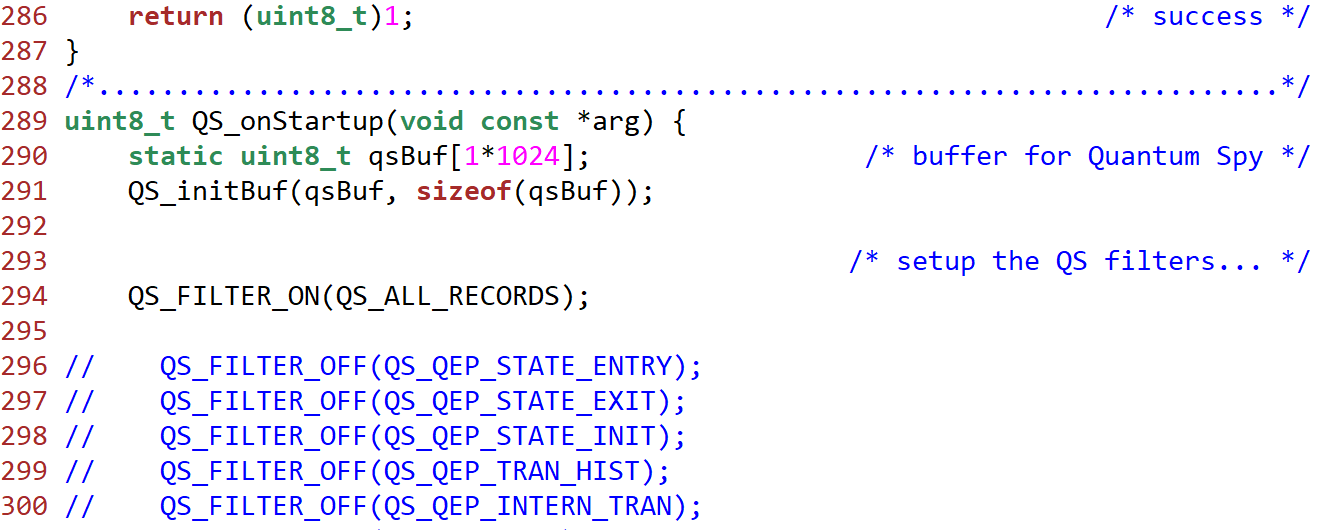


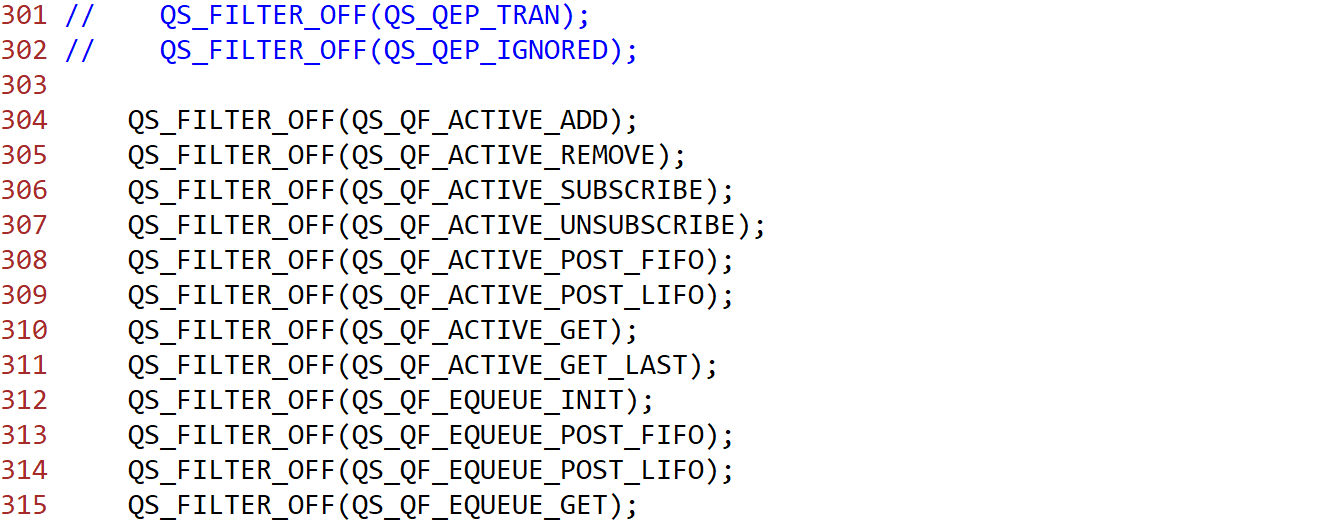


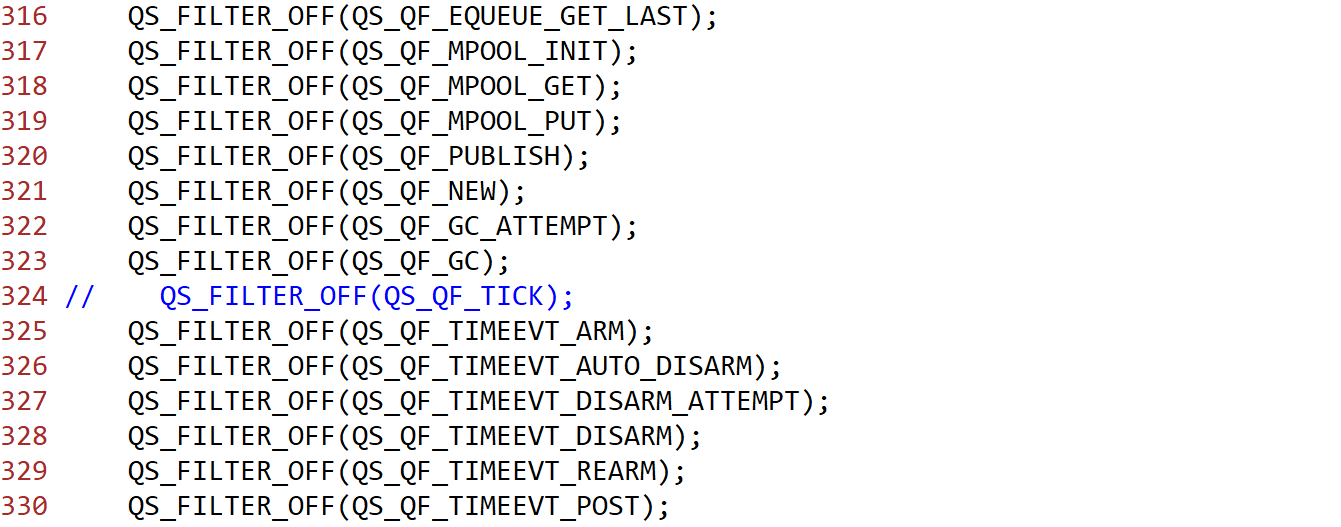


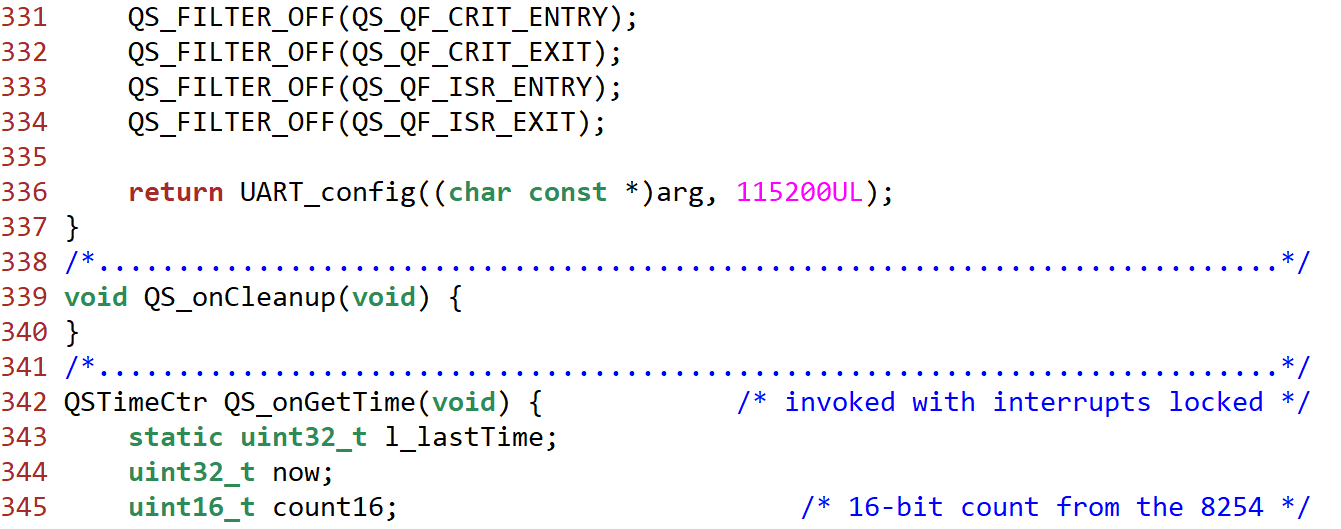


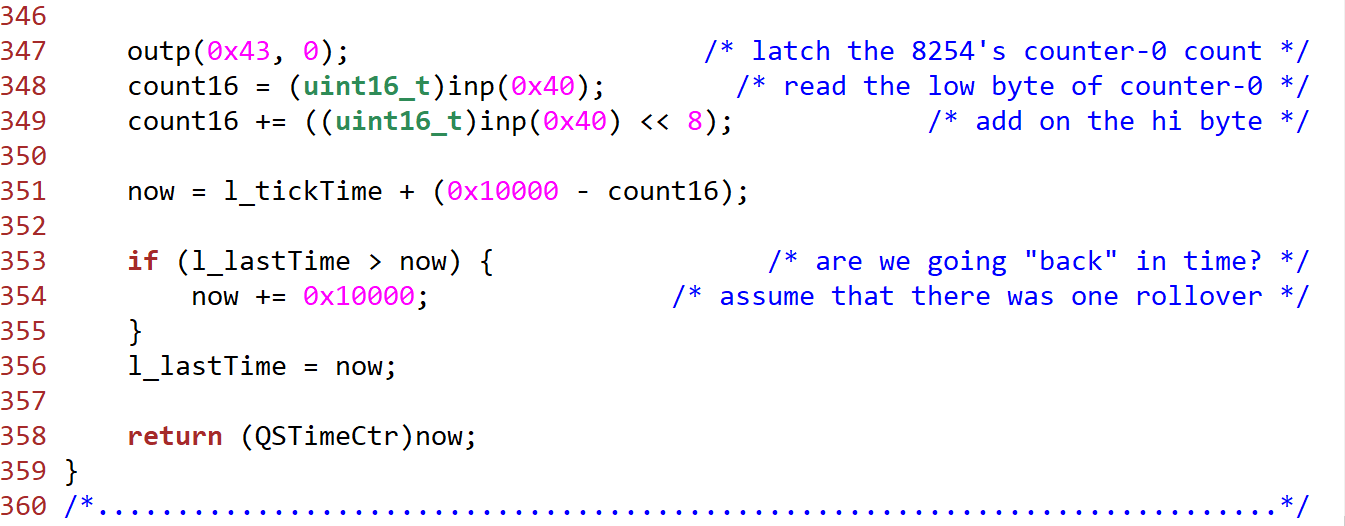


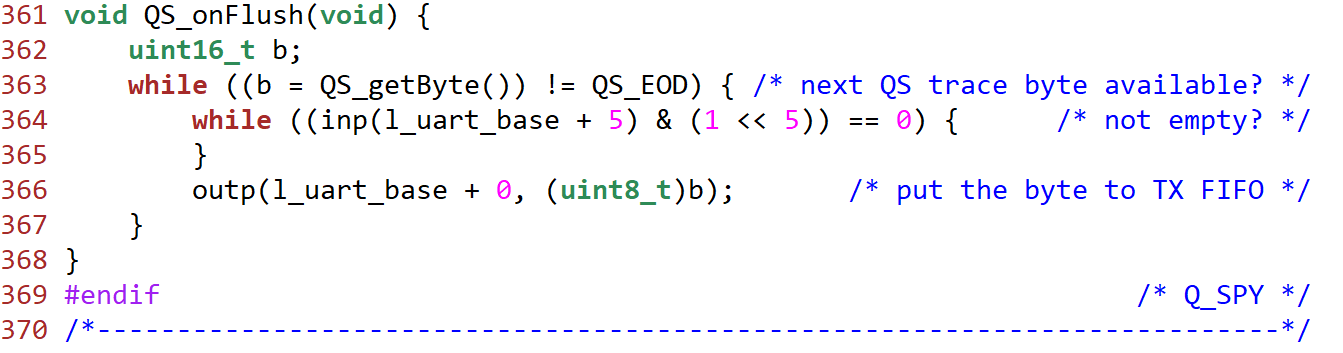








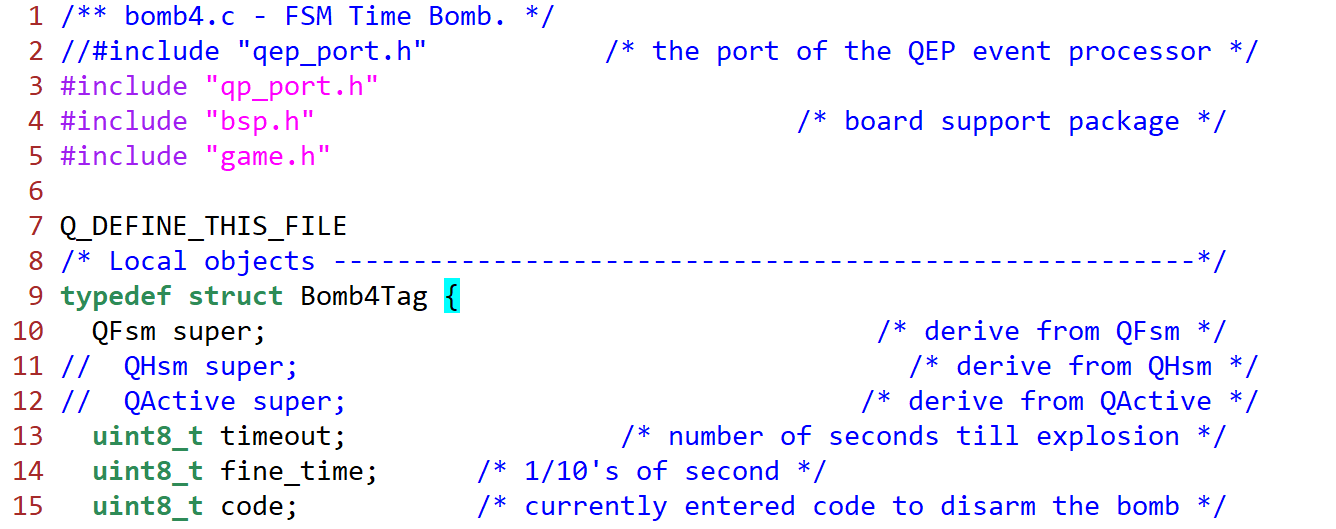


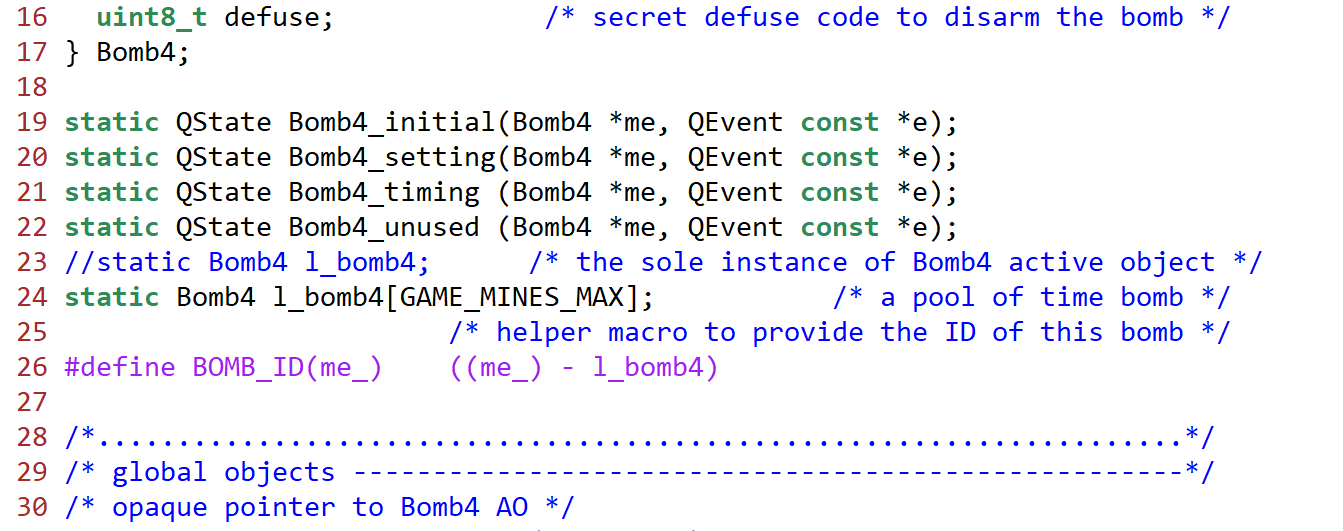


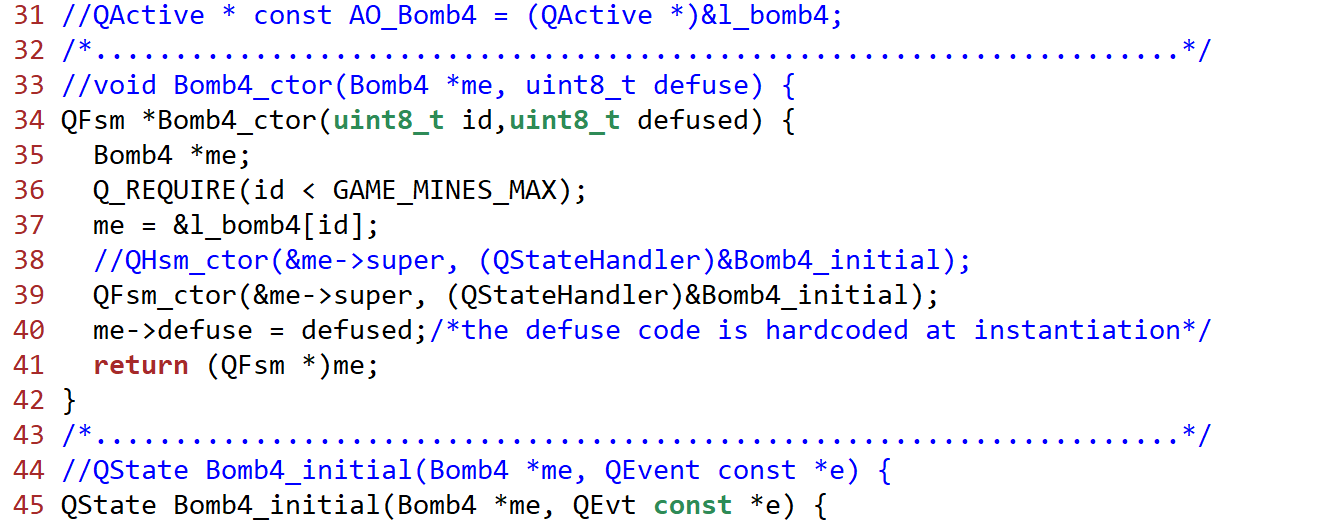
Archivo game/bsp.c

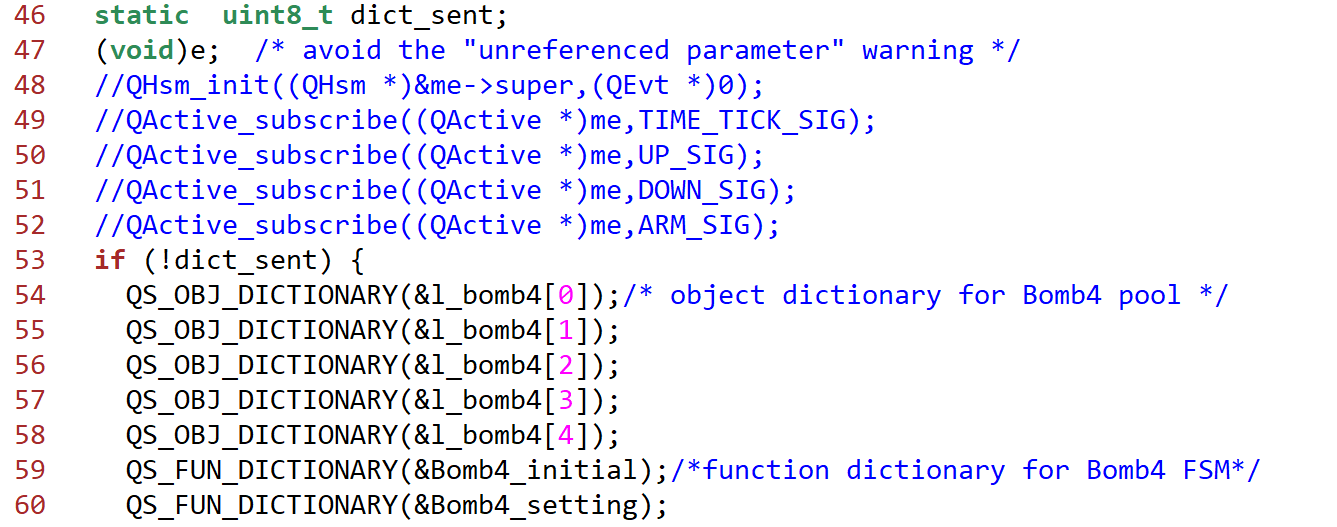
En el listado anterior, el código para crear el evento BombEvt cuando se presiona la tecla ‘u’ y enviarlo al objeto activo AO\_Tunnel está en las líneas 135 a la 142. En la línea 138 se crea el evento BombEvt con la macro Q\_NEW, pasándole el tipo de dato BombEvt y la señal UP\_SIG. En la línea 139 con la macro QACTIVE\_POST, se envía el evento BombEvt al objeto activo AO\_Tunnel.

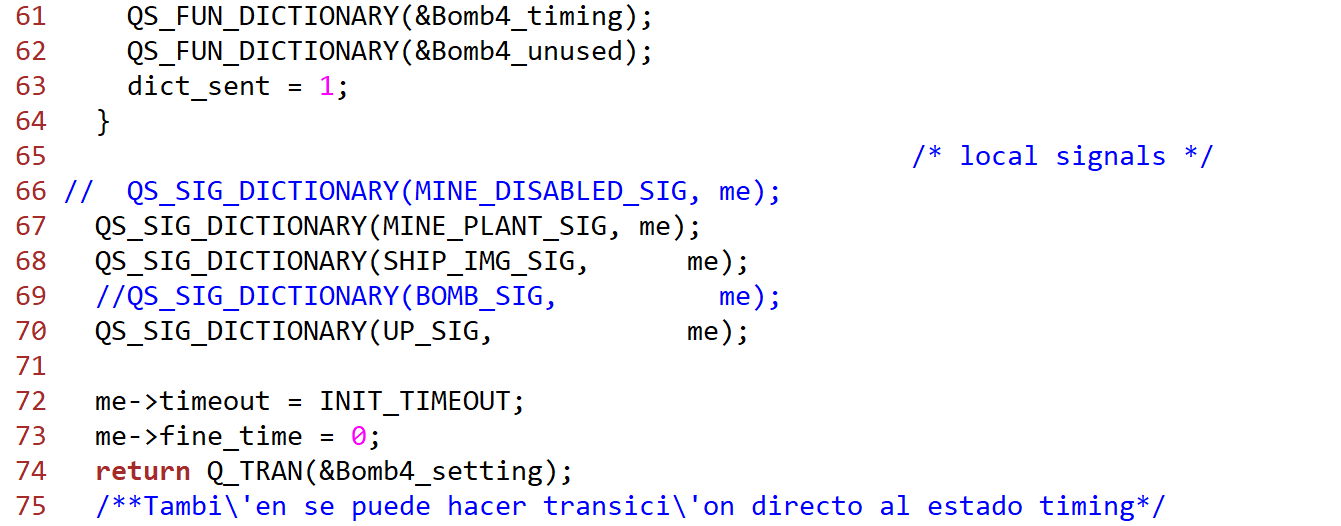
En el archivo game/bomb4.c, en las funciones manejadoras de estado Bomb4\_initial() y Bomb4\_setting(), se agrega el código para responder al evento BombEvt.

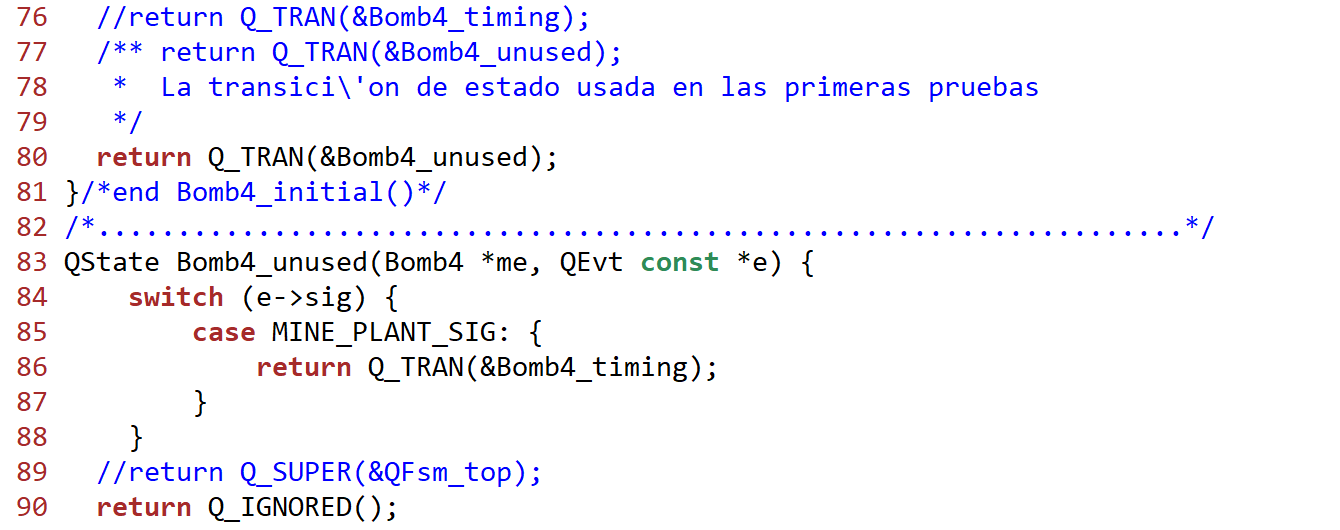


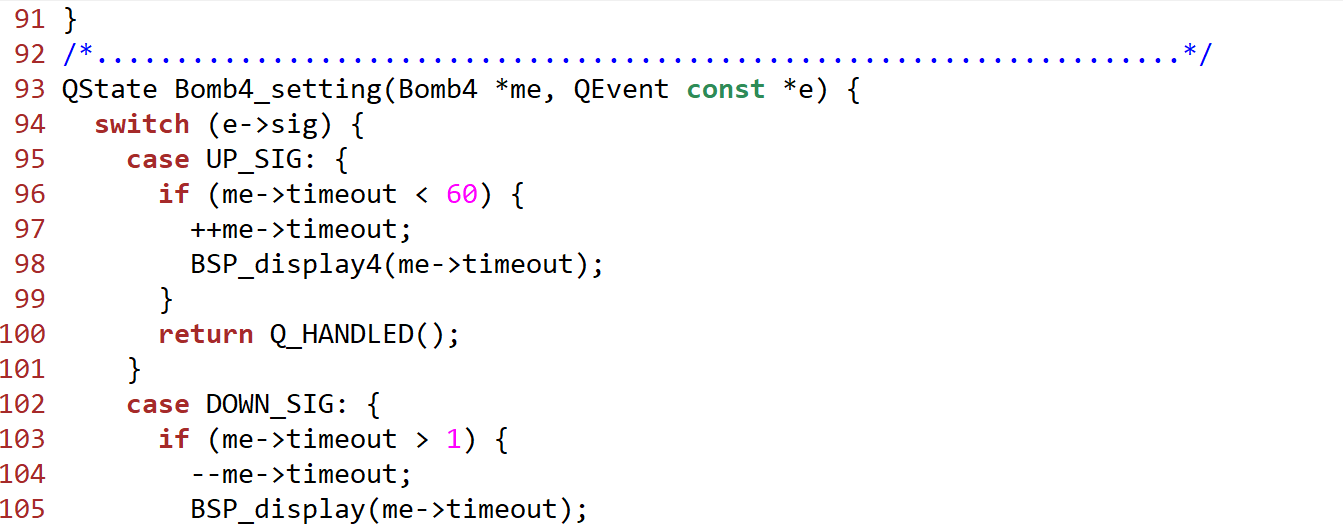


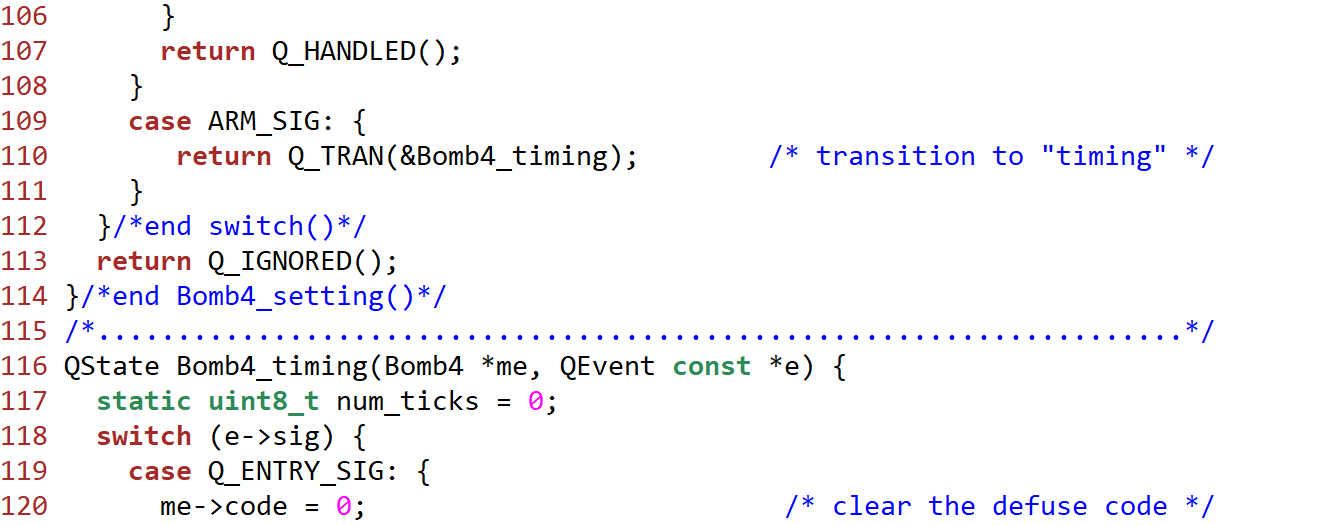


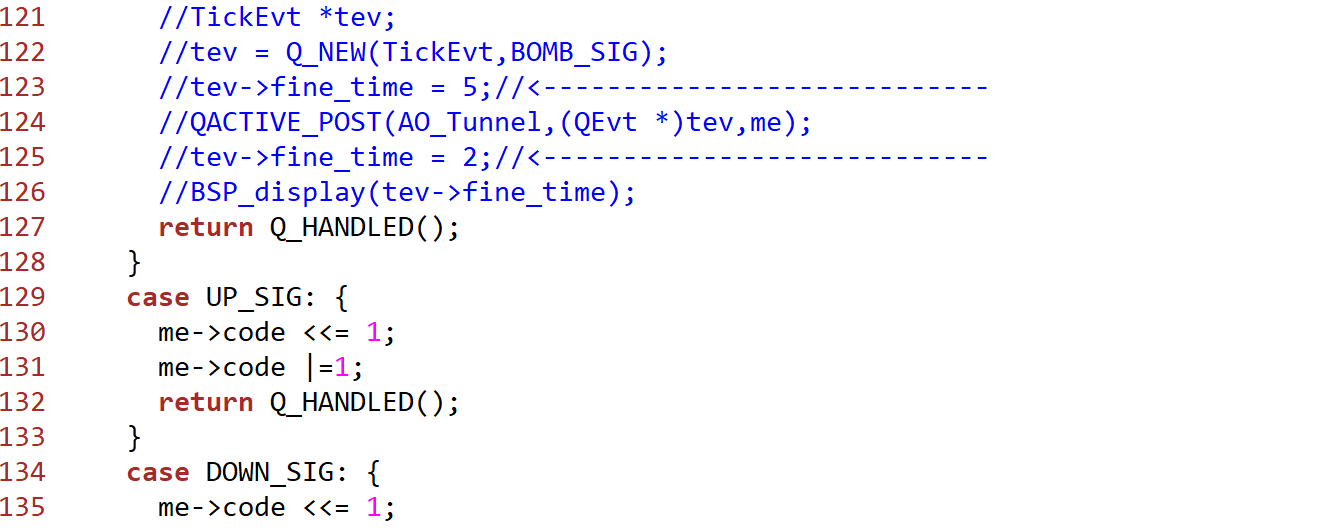


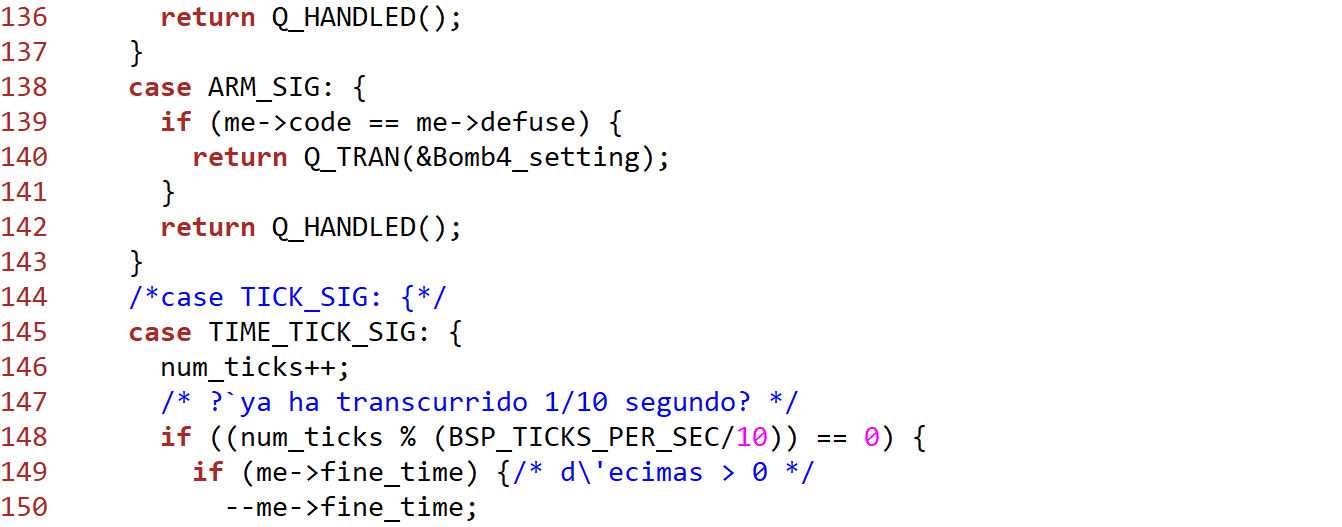


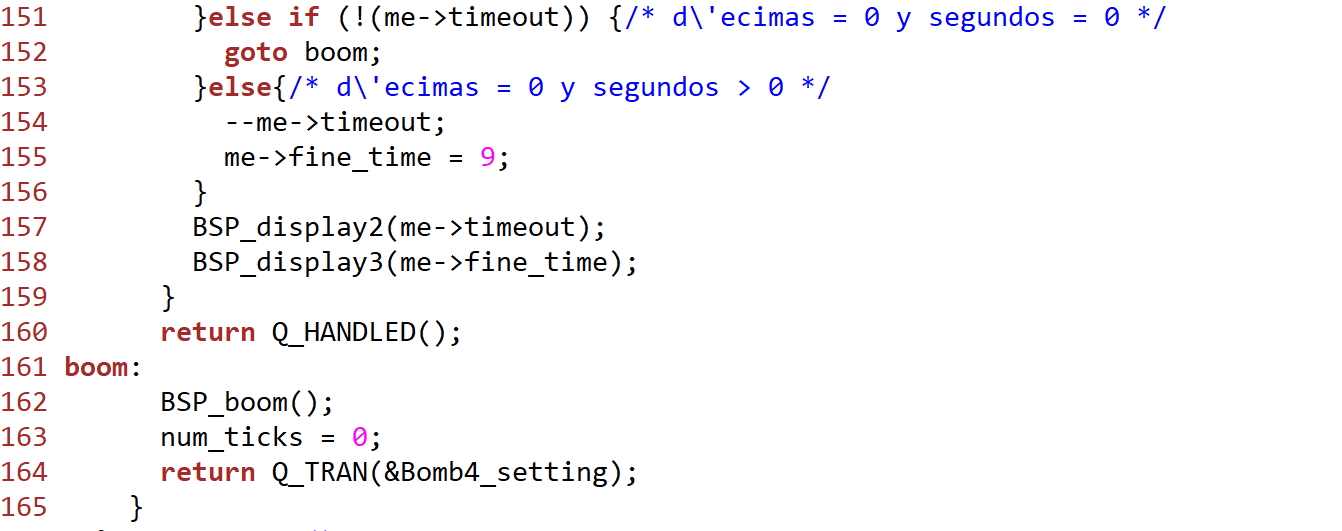














Archivo game/bomb4.c

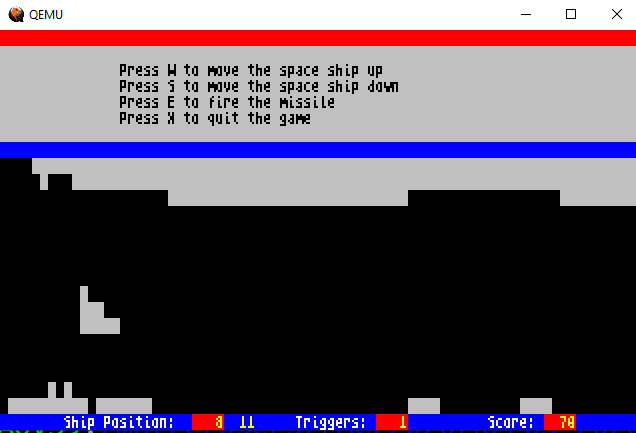
En el listado anterior, se agrega la señal UP\_SIG al diccionario de señales, dentro de la función manejadora de estado Bomb4\_initial() en la línea 70. También, en la línea 74, se hace una transición de estado al estado setting.

En la función manejadora de estado Bomb4\_setting(), el código para procesar el evento BombEvt (con la señal UP\_SIG), se encuentra entre las líneas 95 y 99.

Después de construir el kernel flynshoot.exe, se ejecuta el programa con el comando

$ qemu-system-i386 -kernel flynshoot.exe

Una muestra de la salida del programa se incluye a continuación



(Después de presionar la tecla ‘e’ para iniciar el juego), cuando se presiona la tecla ‘u’ por primera vez, el valor del atributo timeout de la máquina de estado finito se incrementa de 10 a 11, este valor es el que se muestra en la parte inferior de la pantalla de la ejecución de qemu-system-i386.