TEC | Tecnológico de Costa Rica

INSTITUTO TECNOLÓGICO DE COSTA RICA
INGENIERÍA EN COMPUTACIÓN
PRINCIPIOS DE SISTEMAS OPERATIVOS

Tarea 2 : Frogger Booteo UEFI

Joel Vega Godínez

Profesor:

Kevin Moraga García

Sede Inter-Universitaria Alajuela
6 de Abril 2021

Contents

1 Introducción

Frogger (Furoggā) es un juego de arcade desarrollado por Konami y originalmente publicado por Sega. El objetivo del juego es guiar a las ranas a sus hogares, una por una, cruzando carreteras y rios con muchos peligros. Este juego ha encontrado su lugar en la cultura popular, incluyendo la televisión y la música. El juego originalmente se iba a llamar "Highway Crossing Frog", pero los ejecutivos de Sega pensaron que no capturaba la verdadera naturaleza del juego, así que fue cambiado a Frogger. Frogger es considerado como uno de los 10 mejores videojuegos de todos los tiempos según Killer List of Videogames. El original "Highway Crossing Frog" era de hecho una copia de un juego llamado "Freeway desarrollado en 1971 por el departamento de psicología de la Universidad de Washington en una minicomputadora IMLAC PDS-1, considerado como parte de un proyecto relacionado con el estudio de la memoria humana a corto plazo.

2 Ambiente de Desarrollo

La presente tarea fue desarrollada en ensamblador, específicamente NASM que un ensamblador libre para la plataforma Intel x86. Puede ser usado para escribir programas de 16-bit, 32-bit y 64-bit. Además, se utilizo qemu para realizar la virtualización del programa en el área de booteo.

3 Estructuras de datos utilizadas

3.1 Tablero

El tablero fue un arreglo que se utilizo parar representar la matriz al jugador imprimiéndolo en consola, esta estructura básicamente se reservaron 980100 bytes para su uso y representación.

3.2 Tablero_log

El tablero_log es una estructura del mismo tamaño que el tablero, pero esta fue utilizada para la lógica del juego y esta no se muestra en consola, se utilizo para ser recorrida cada movimiento y verificar cuando mover los troncos o autos.

3.3 Movimiento

Movimiento fue una estructura en la cual se le asignaron 2 bytes para recibir input del jugador, para lograr realizar movimientos dentro del tablero.

4 Funciones Utilizadas

4.1 Crear_matriz

Esta función se encarga de crear una matriz de 13x15, en donde se va a representar los autos, jugador, calles , agua y troncos.

4.2 Print

Esta función se encarga de imprimir el tablero después de realizar cada movimiento o jugada.

4.3 Preguntar_movimiento

Se encarga de recibir un input y almacenarlo en la estructura de datos movimiento, con esto conoce a donde se debe mover el jugador y representarlo en el tablero.

4.4 Colocar_jugador

Su función es colocar el jugador en su posición inicial en la matriz en el inicio del juego.

4.5 Colocar_automóviles

Su función es colocar los automóviles en sus respectivas posiciones iniciales en la matriz en el inicio del juego.

4.6 Colocar_troncos

Su función es colocar los troncos en sus respectivas posiciones iniciales en la matriz en el inicio del juego.

4.7 Mover_automóviles

Se encarga de realizar los movimientos de cada uno de los tipos de automóviles después de haber realizado un movimiento del jugador.

4.8 Mover_troncos

Se encarga de realizar los movimientos de cada uno de los tipos de troncos después de haber realizado un movimiento del jugador.

5 Instrucciones para ejecutar el programa

5.1 Compilar Programa

En la carpeta del programa se posee un makefile por consiguiente lo unico necesario para compilarlo es realizar el comando:

```
$ make
```

5.2 Ejecutar Programa

```
$ ./frogger
```

5.3 Compilar UEFI asm

En la carpeta en donde se encuentre el .asm realizamos los siguientes comandos en la terminal.

```
$ mkdir -p drive
$ fasm hello.asm drive/hello
```

5.4 Ejecutar Programa con QEMU

```
$ qemu-system-x86_64 -bios OVMF.fd -net none -drive
format=raw,file=fat:rw:drive/
```

6 Actividades Realizadas por el estudiante

Actividad	Descripción	Horas Invertidas	Fecha
Búsqueda de Información	Búsqueda de información necesaria para realizar booteo uefi e información que ayude a realizar el juego frogger.	4 horas	28/3/2021
Desarrollo de Código	Se realizó la implementación de el juego frogger en nasm y se intentó realizar el booteo con uefi.	6 horas	29/3/2021 30/3/2021 31/3/2021
Documentación del Código	Se realizó documentación interna y externa del código.	1.30 horas	4/4/2021

7 Autoevaluación

Rubro	Valor	Autoevaluación	Comentarios
Sector de Arranque	30%	25%	En el rubro de sector de arranque, se puede bootear de modo uefi pero no el programa principal, se bootea un archivo diferente al del frogger.
Frogger	50%	50%	El frogger contiene todo lo requerido por el rubro.
Documentación	20%	20%	La documentación posee todos las secciones requeridas por el rubro.

8 Lecciones Aprendidas

Con el presente proyecto comprendí diferentes aspectos acerca del uefi boot, el uefi boot permite ejecutar código en 32 y 64 bits, además, tiene una interfaz más moderna a comparación de otros métodos de boot en donde permite incluir animaciones y sonidos, y permite utilizar el mouse para interactuar con ella. Y como se hizo para este proyecto uefi se puede cargar en cualquier recurso de memoria no volátil, lo que permite que sea independiente de cualquier sistema operativo y esto nos permite bootear aplicaciones como la que se realizó en el presente proyecto.

Además, con el uso de NASM, fue bueno para recordar la manera en la cual se debe manejar los registros, para no perder ningún valor y ejecutar la acción que queremos. Además, la manera de trabajar arreglos en ensamblador y el pedir entrada al usuario, fue de gran utilidad para comprender la asignación de memoria en bytes que se debía hacer para lograr estas acciones.

9 Bibliografía

Boot Sequence - OSDev Wiki. (2021). Recuperado 28 marzo 2021, de https://wiki.osdev.org/Boot_Sequence

Getting started with bare-metal assembly. (2020). Recuperado 28 marzo 2021, de https://johv.dk/blog/bare-metal-assembly-tutorial.html

UEFI Programming - First Steps. (2021). Recuperado 28 marzo 2021, de http://x86asm.net/articles/uefi-programming-first-steps/