NesOS

Un kernel emulador

Aldana Ramirez
Martín Villagra

21 de septiembre de 2014

Resumen

poner una foto de la consola arriba de esto
El objetivo del trabajo es construir un emulador de la videoconsola Nintendo Entertaiment System(también conocida como Family) que se ejecute sin ningún sistema operativo de por medio.

Índice general

Capítulo 1

Introducción

algo historia de la consola, que fue la más exitosa de su época, como trascendio, que era barata,etc

porque la elegimos: es una de las consolas mas investigadas y toqueteadas que alguna vez existio.

modo de trabajo: se busca no reinventar la rueda, el projecto se construyo tomando como guia otros kernels y emuladores ya existentes

la mayoría se va a hacer en c, solo se usa assembler si es necesario *se asume conocimientos en c por parte del lector*

1.0.1. Estructuración

En un principio dividimos en dos partes principales el projecto.

Una es el **emulador** propiamente dicho, es decir la que se encarga de hacer todo lo que hacía internamente la consola.

La otra parte es el **kernel** encargada de inicializar todo lo necesario para que el emulador funcione, así como proveerle funciones de bajo nivel tales como escribir en pantalla o reservar memoria. Al no tener un sistema operativo detrás, funciones como malloc y free que cualquier programador de C supone siempre presentes deben ser implementadas por el kernel.

Capítulo 2

El Kernel

Empecemos por decir que lo que aquí se presenta está muy lejos de un sistema operativo completo.

Se priorizó la simplicidad recortando todo lo que estaba de más, reduciendo al mínimo las capacidades del sistema. Por ejemplo la mayoría de los sistemas operativos pueden leer un programa y ejecutarlo. Nuestro querido kernel carece de esa posibilidad.

Lo único que se puede ejecutar actualmente en nuestro sistema es el emulador, y eso sólo porque está embebido en el kernel. Podría verse al emulador como un solo programa standalone, que no necesita ningún sist. operativo para ejecutarse.

Y entonces que necesita como mínimo nuestro emulador? Se muestra, en orden de importancia: 0) Iniciar el sistema y ejecutar el emulador (booteo) 1) Modificar libremente pixeles de la pantalla 2) Detectar pulsaciones del teclado 3) Poder reservar memoria dinámica *explicar que es* 4) Cargar de alguna forma los juegos 5) Ejecutar sonido 6) Escribir en algún medio persistente el estado actual del juego(Guardar la partida)

Pues bien, el kernel tiene que ser capaz de proveer funciones que faciliten cada una de estas tareas. En las siguientes secciones se detallaran cada una de ellas.

2.1. Booteo

Para evitar tener que lidiar con la BIOS y otras interfaces de bajo nivel, se eligió utilizar un bootloader ya existente, GRUB. *poner que carajos es GRUB*. En particular existe un standard llamado multiboot que especifica como estructurar la cabecera[1] de un kernel para que el mismo pueda ser car-

[1] header, la primera parte

gado por GRUB(o por cualquier otro bootloader que implemente multiboot). En esta cabecera se determina que modo se prefiere(texto o consola) y que función va a ser la 1era en ser llamada. A si mismo GRUB se comunica con la BIOS entre otras cosas y recolecta información de la máquina que luego es recibida convenientemente por nuestro kernel. De esta forma al encender la máquina se iniciará GRUB, el mismo va a poder detectar nuestro kernel y lo va a ejecutar.

El problema es que al momento en que GRUB ejecuta el kernel el stack pointer no está inicializado por lo que no es posible que la función inicial sea en C. Debemos comenzar en assembler, inicializar el stack pointer y ahí si pasar e C.

Bibliografía

[AC] A Cottrell, *Word Processors: Stupid and Inefficient*, www.ecn.wfu.edu/~cottrell/wp.html

[BR] Visit www.dur.ac.uk/library/using/guides/ and click on '*Writing your bibliography and citing references*'.

[ESL] L Truss, *Eats, Shoots and Leaves*, Profile Books 2003 (ISBN 1-86197-612-7).

[GRM] M Goossens, S Rahtz and F Mittelbach, *The LaTeX Graphics Companion*, Addison-Wesley, 1997 (ISBN 0-201-85469-4).

[IM] *ImageMagick*, www.dur.ac.uk/its/software/application/... ...?application=ImageMagick

[LAT] *LaTeX stuff*, maths.dur.ac.uk/Ug/projects/resources/latex/

[MEM] *Memoir document class*, www.ctan.org/tex-archive/macros/latex/contrib/memoir/

[MG] F Mittelbach and M Goossens *et al.*, *The LaTeX Companion*, Addison-Wesley, 2nd ed. 2004 (ISBN 0-201-36229-6).

[MKT] *MikTex Project Page*, www.miktex.org

[NSS] T Oetiker, H Partl, I Hyna and E Schlegl, *The Not So Short Introduction to LaTeX 2$_\varepsilon$*, www.ctan.org/tex-archive/info/short

[PS] *Photoshop*, www.dur.ac.uk/its/software/application/... ...?application=Adobe+Photoshop

[TXC] *TeXnicCenter*, www.toolscenter.org

[WDT] *WinEdt*, www.winedt.com

[WL] *Wikibook on LaTeX*, en.wikibooks.org/wiki/Latex

[WO] *Controlling widows and orphans*,
www.tex.ac.uk/cgi-bin/texfaq2html?label=widows

[WSH] *WinShell*, www.winshell.de