Practica 1

Instalación de un Sistema Operativo en Tiempo Real

La instalación de un sistema operativo en una computadora personal consiste en colocar algunos archivos especiales en un medio de almacenamiento, de manera tal que cuando se energice la computadora, luego de algunas operaciones intermedias, el kernel del sistema operativo sea colocado en cierta localidad de memoria y se le transfiera el control del sistema de cómputo. Para colocar el kernel del sistema operativo en cierta localidad de memoria de una computadora personal, se utiliza un pequeño programa conocido como cargador de arranque.

Descarga de los archivos fuente

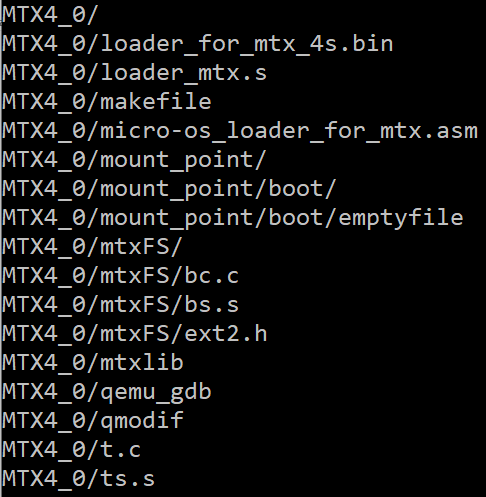
Primero descargamos el archivo MTX4\_0.tar con el comando

$ wget <https://raw.githubusercontent.com/sotrteacher/dirtywork/master/MTX4_0.tar>

Una vez descargado el archivo tar, extraemos su contenido con el comando

$ tar xvf MTX4\_0.tar

Con este comando obtendremos el directorio MTX4\_0 y su contenido



Extracción del contenido del archivo MTX4\_0.tar

Cargador de Arranque

En esta primera etapa de la práctica, se dan las indicaciones para construir un programa cargador de arranque llamado loader, el cual será colocado en el sector 1 de un archivo imagen de disco floppy, llamado mFDimage.

Construcción del archivo imagen de disco floppy

El archivo mFDimage lo vamos a construir con el comando genext2fs. Primero ingresamos al directorio MTX4\_0/, para ello, usando el comando

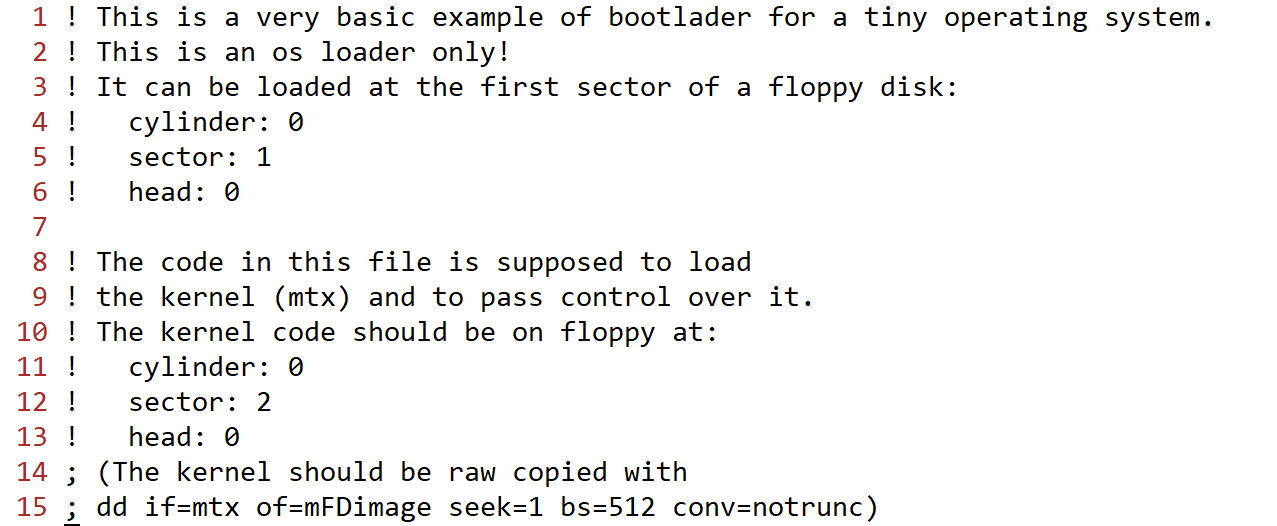
$ cd MTX4\_0/

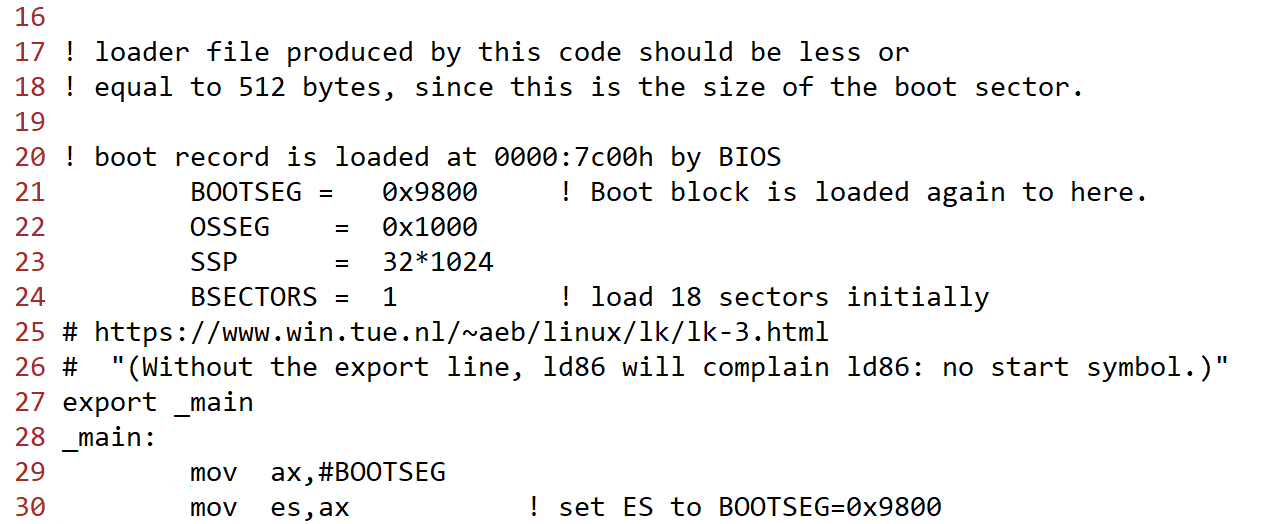
Y ejecutamos el comando

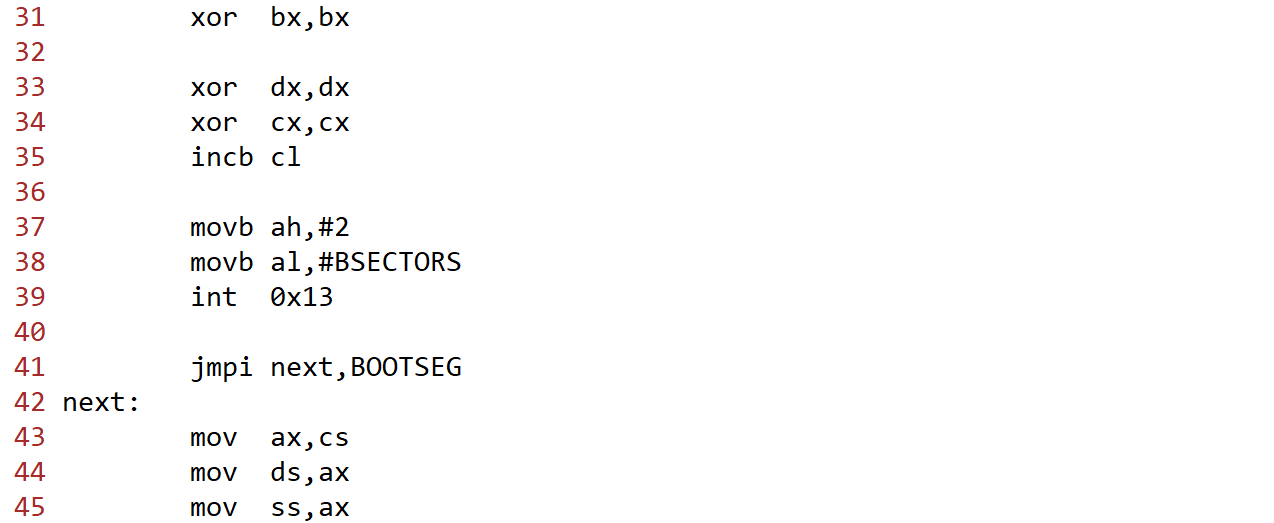
$ genext2fs –b 1440 –d ./mount\_point/ mFDimage

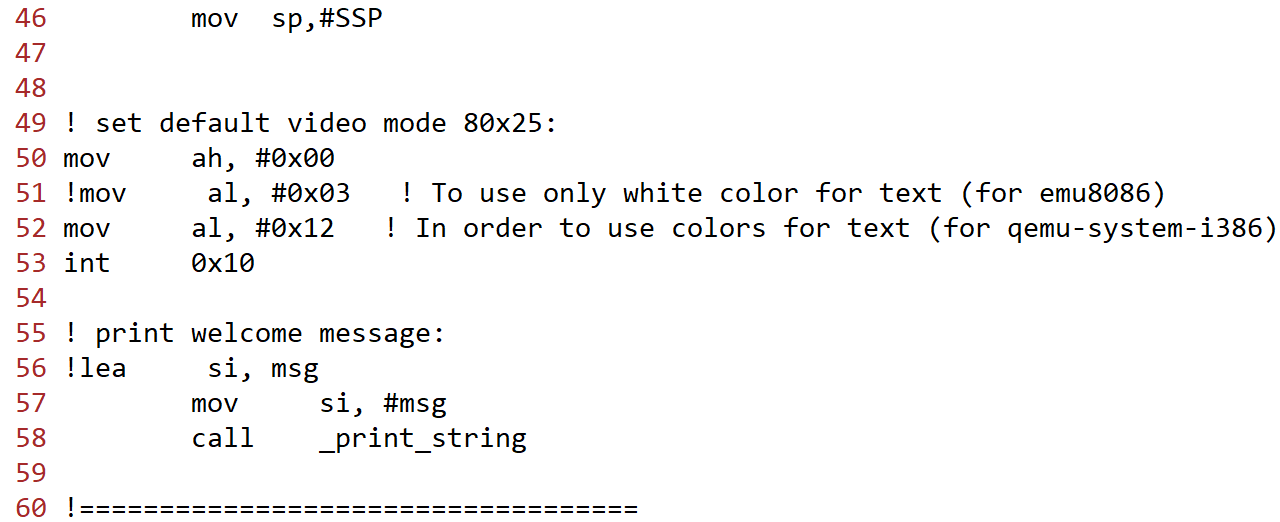
Construcción del cargador de arranque

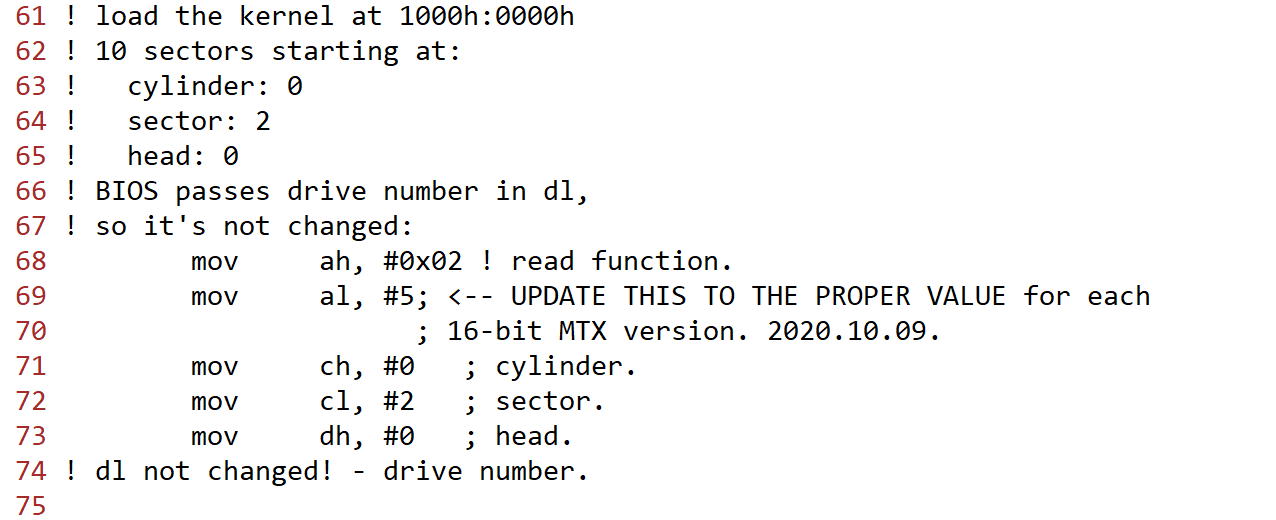
El archivo fuente para el cargador de arranque se muestra a continuación, se trata de un archivo en lenguaje ensamblador que está escrito de acuerdo con la sintaxis del ensamblador as86.

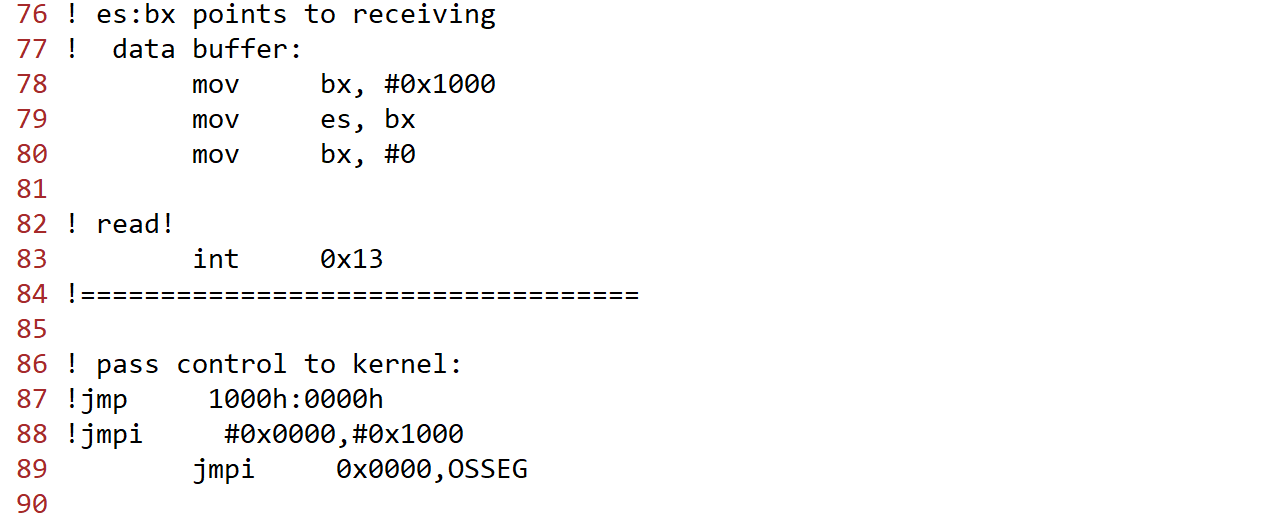


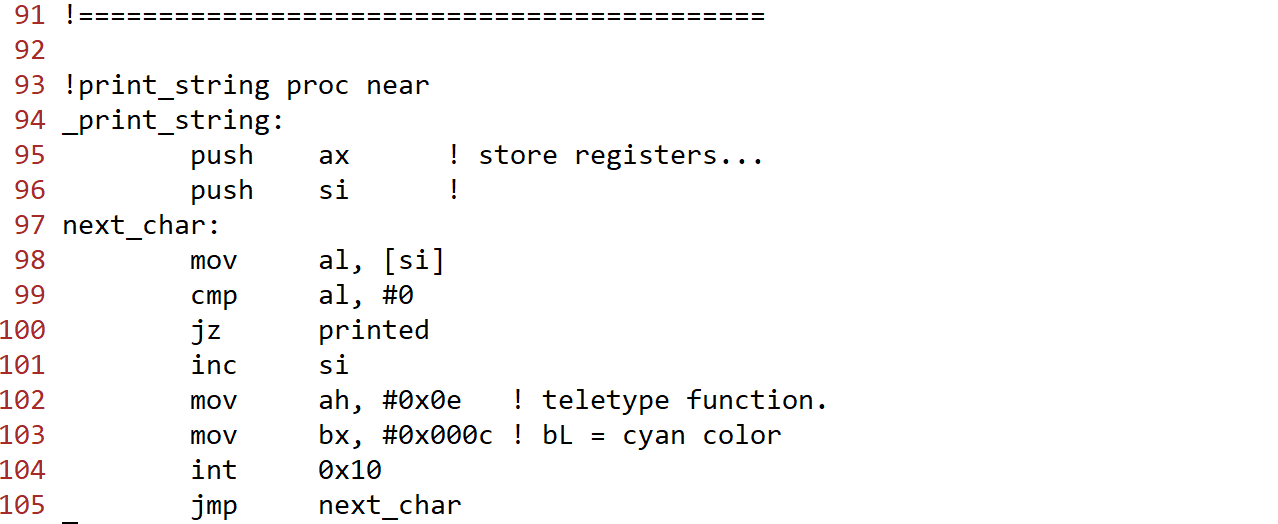


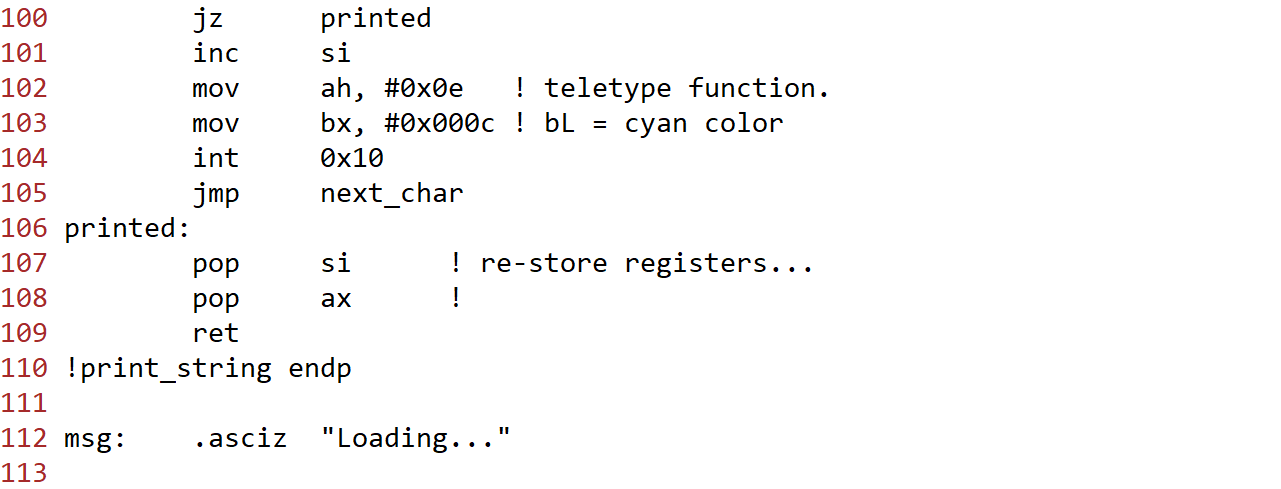












Archivo loader\_mtx.s

Construya el cargador de arranque con los comandos

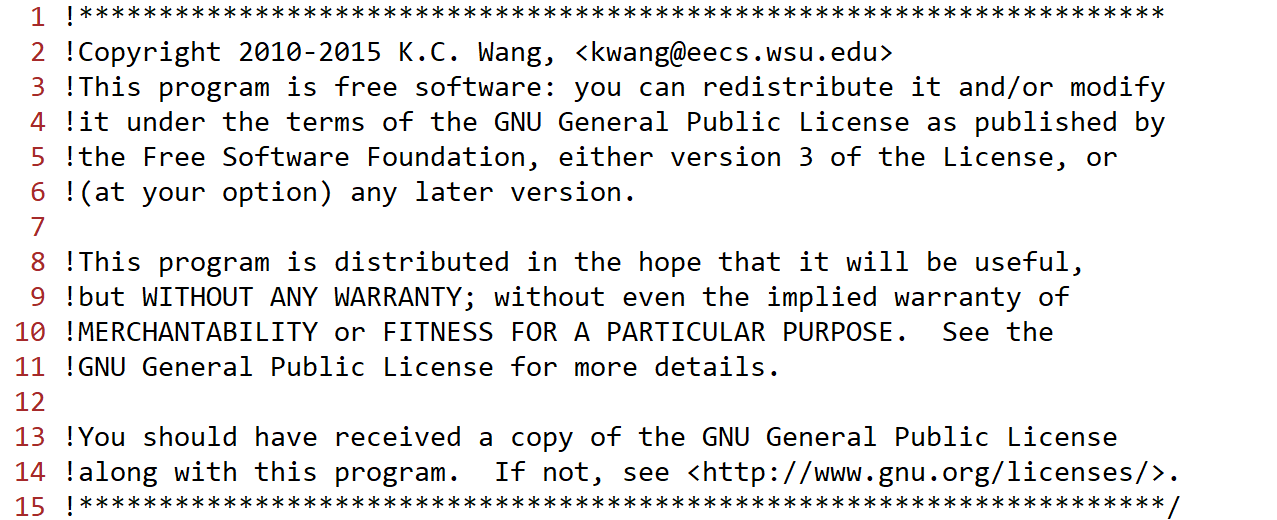
$ as86 –o loader\_mtx.o loader\_mtx.s

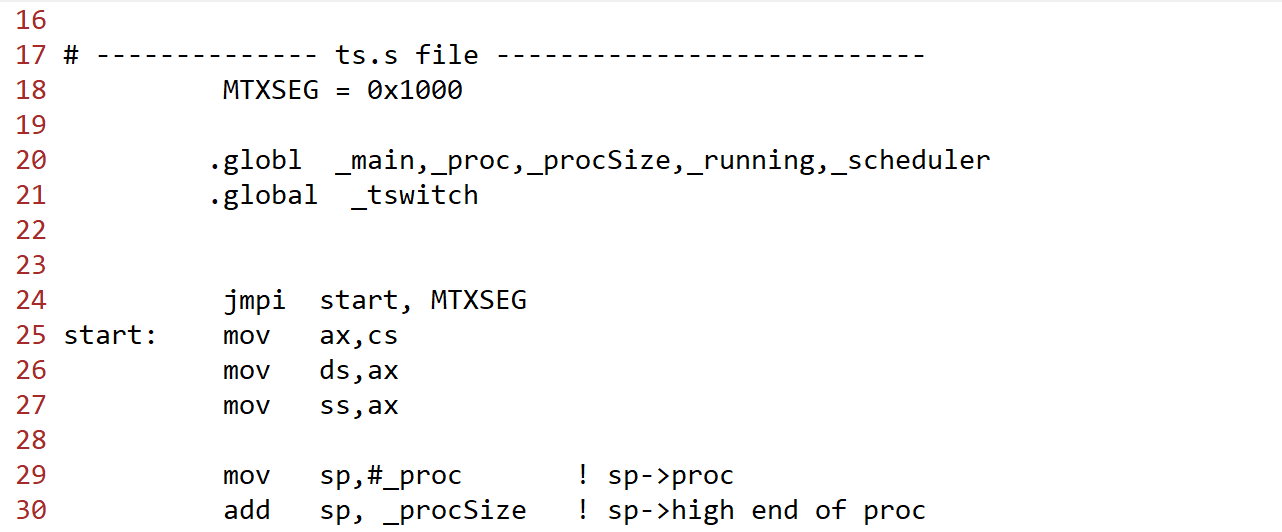
$ ld86 –d –o loader loader\_mtx.o

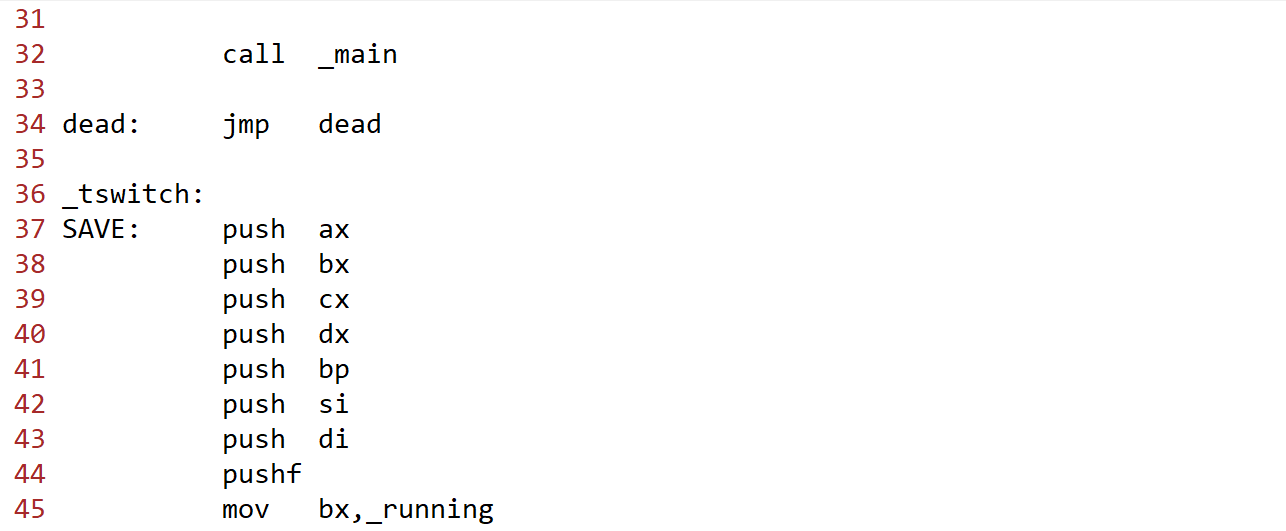
Con los comandos anteriores se debe obtener el cargador de arranque en el archivo loader (el cual debe ser de un tamaño menor a 512 bytes).

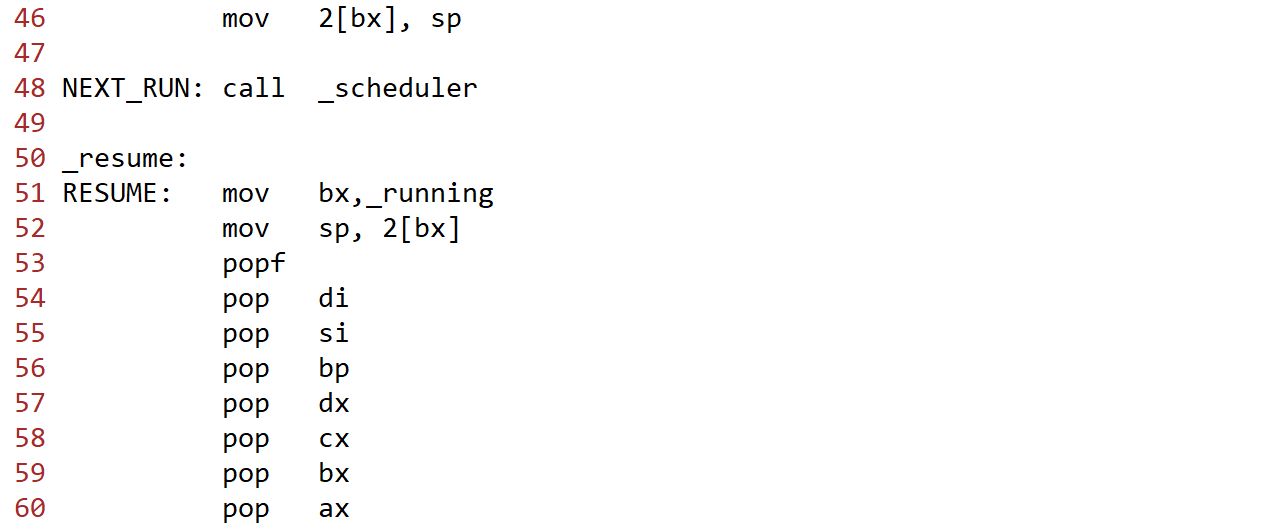
Construcción del kernel MTX4.0

El kernel MTX4.0 está formado por solamente dos archivos, ts.s y t.c. Ambos archivos se muestran a continuación



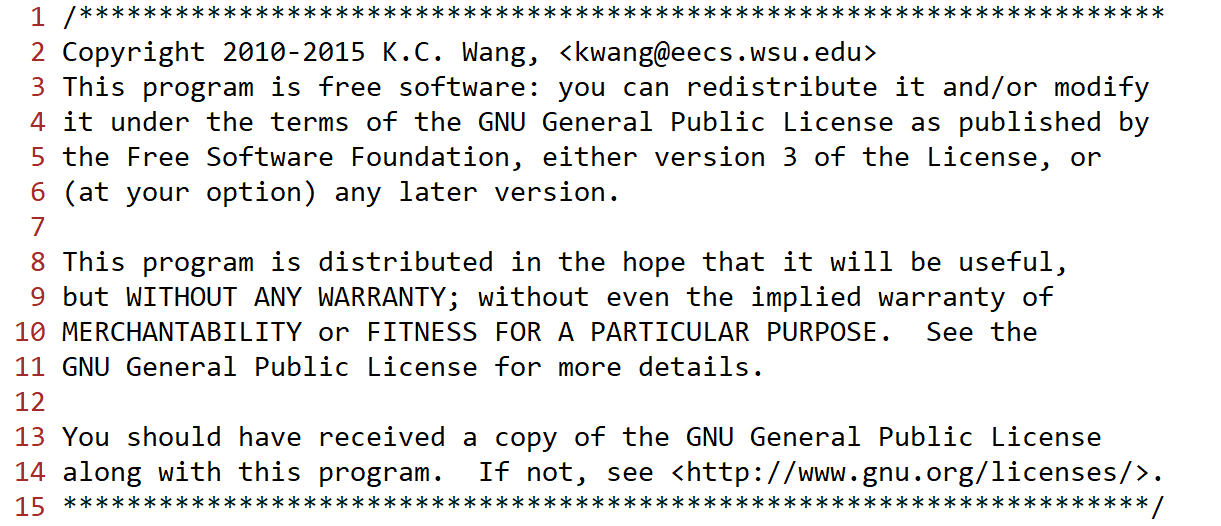


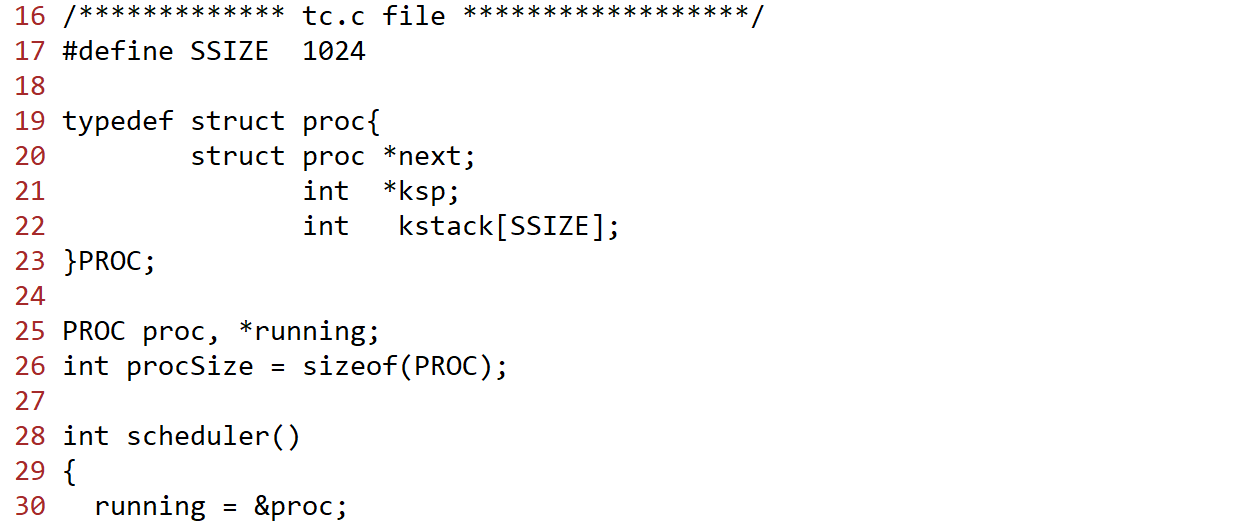


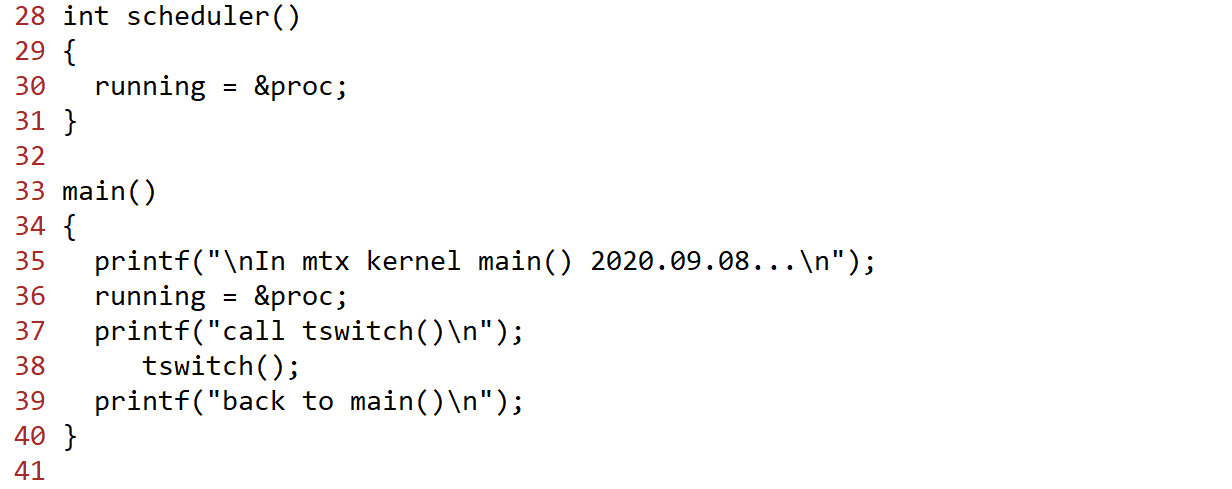




Archivo ts.s







Archivo t.c

Construya el kernel MTX4.0 con los comandos

$ as86 –o ts.o ts.s

$ bcc –o t.o -c –ansi t.c

$ ld86 –d –o mtx ts.o t.o mtxlib /usr/lib/bcc/libc.a

Con los comandos anteriores se debe haber creado el kernel MTX4.0 en el archivo mtx.

Creación del disco floppy virtual de arranque

Colocación del cargador de arranque en el primer sector del archivo mFDimage

$ dd if=loader of=mFDimage bs=512 count=1 conv=notrunc

Colocación del kernel mtx a partir del segundo sector del archivo mFDimage

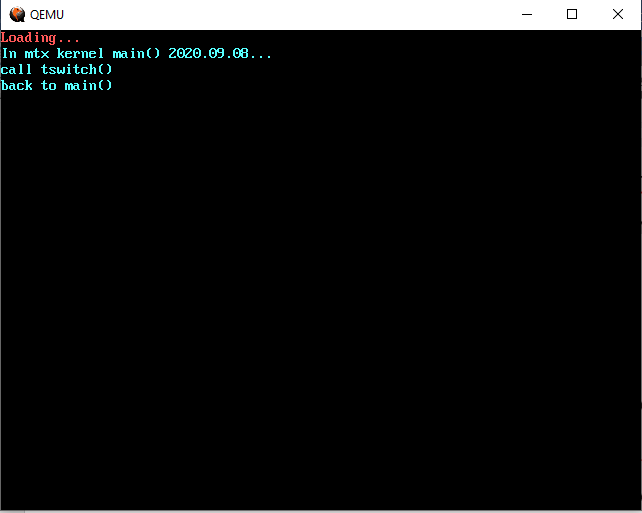
$ dd if=mtx of=mFDimage seek=1 bs=512 conv=notrunc

Booteo del kernel MTX4.0 en qemu-system-i386

Para correr el kernel mtx, ejecute el siguiente comando:

$ qemu-system-i386 –fda mFDimage –no-fd-bootchk

La ejecución del kernel MTX4.0 debe observarse como se muestra en la siguiente imagen.

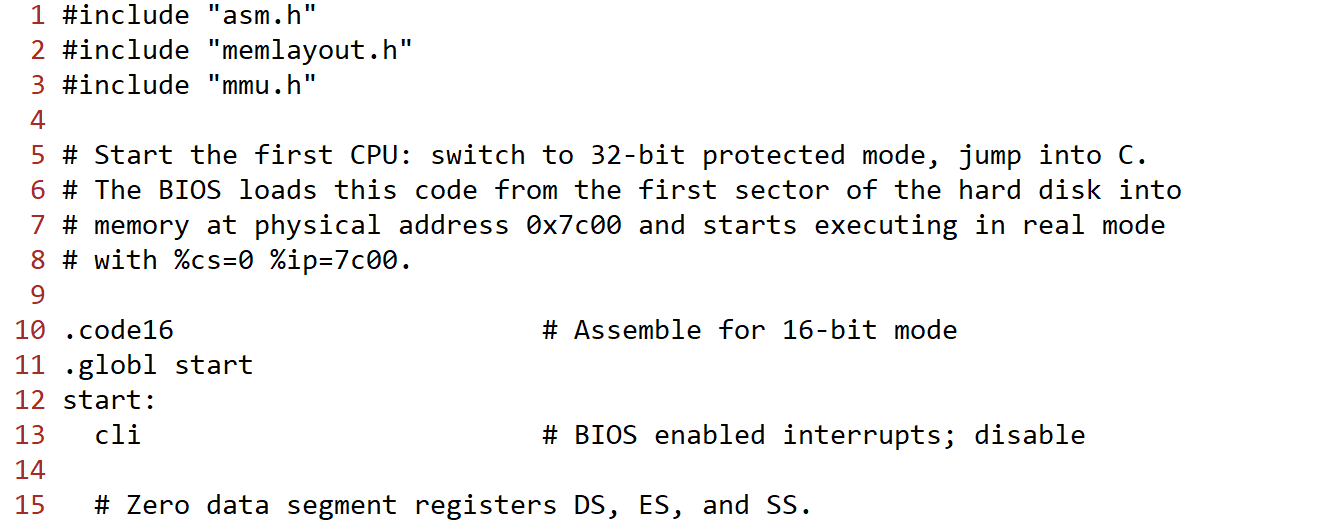


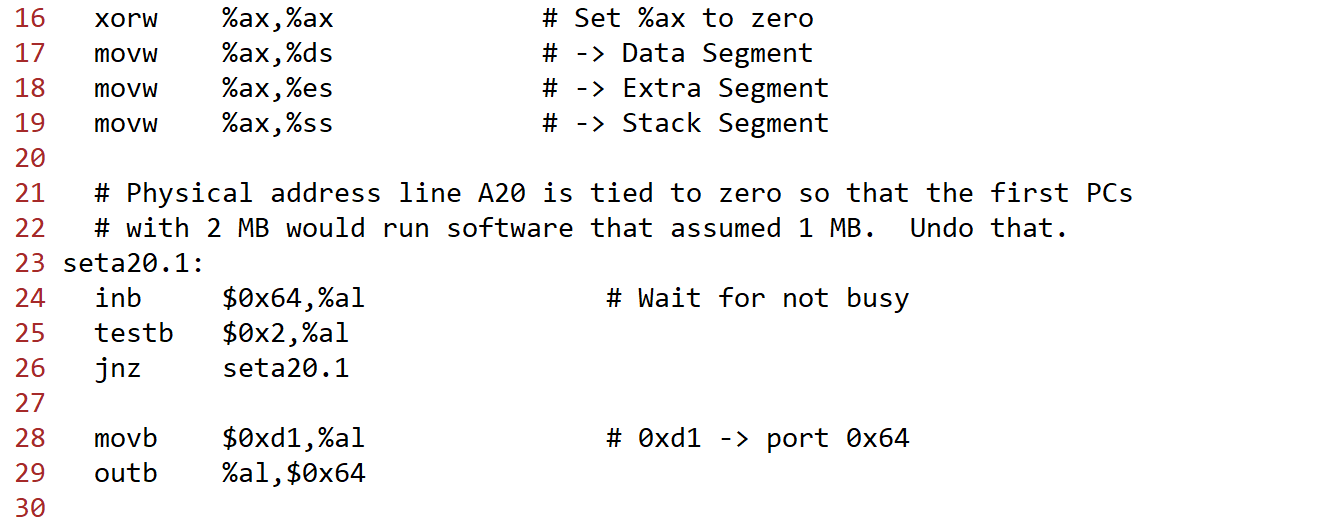
Ejecución del kernel MTX4.0 sobre qemu

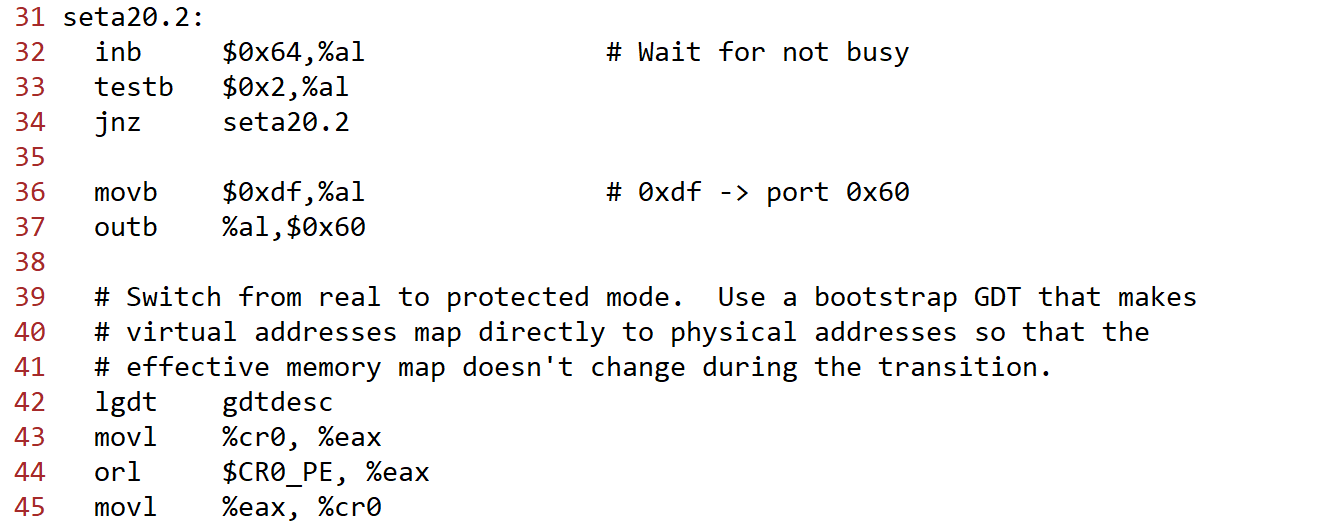
Instalación de un kernel ejecutable en el modo protegido de 32 bits

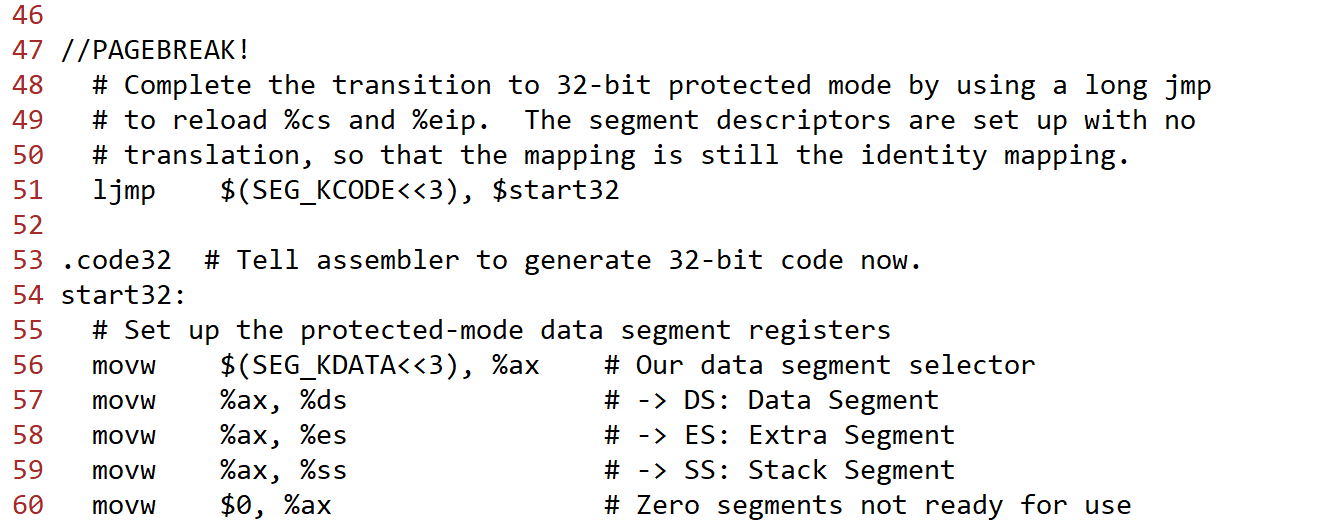
En esta sección de la práctica se dan indicaciones para crear un archivo de disco duro virtual para utilizarlo como dispositivo de arranque en qemu-system-i386.

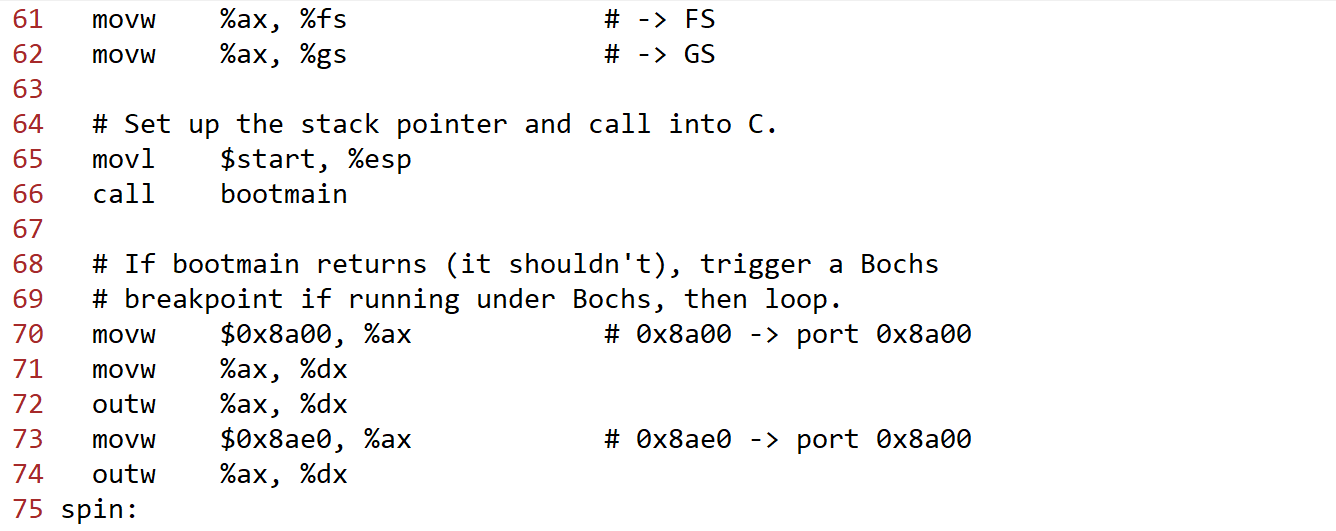
Igual que en el caso del archivo de disco floppy virtual, comenzamos con los archivos fuente del cargador de arranque, se trata de los archivos bootasm.S y bootmain.c (ambos archivos fueron tomados de la distribución del sistema operativo xv6, repositorio xv6-public en <https://github.com>).

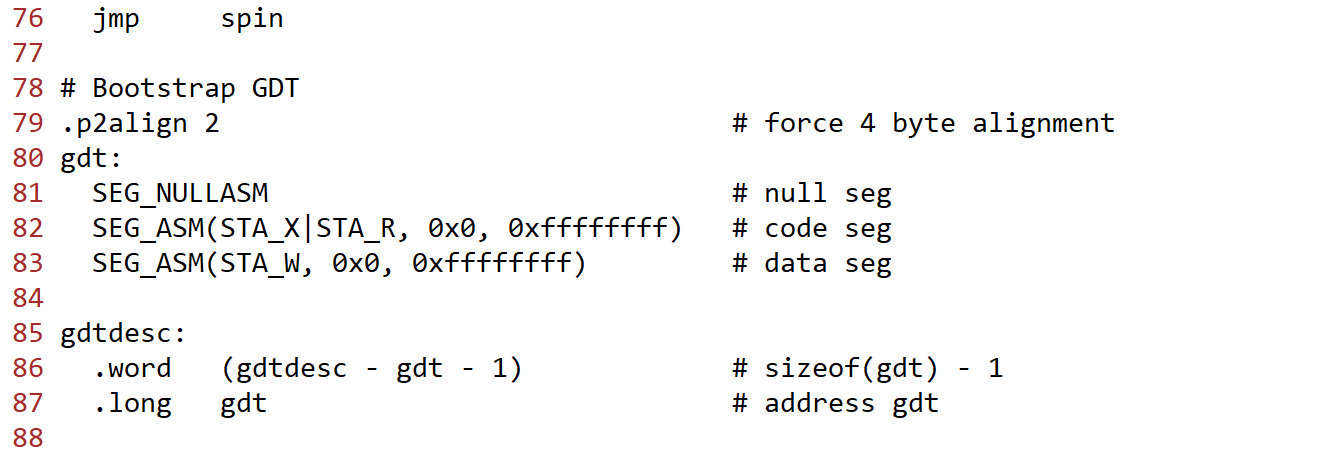




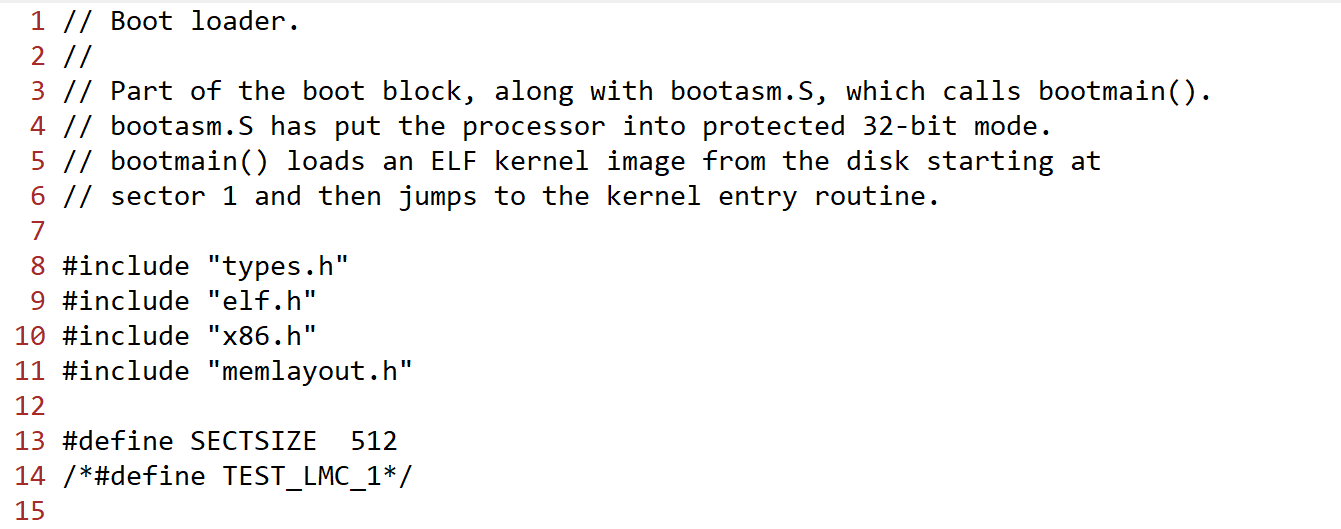


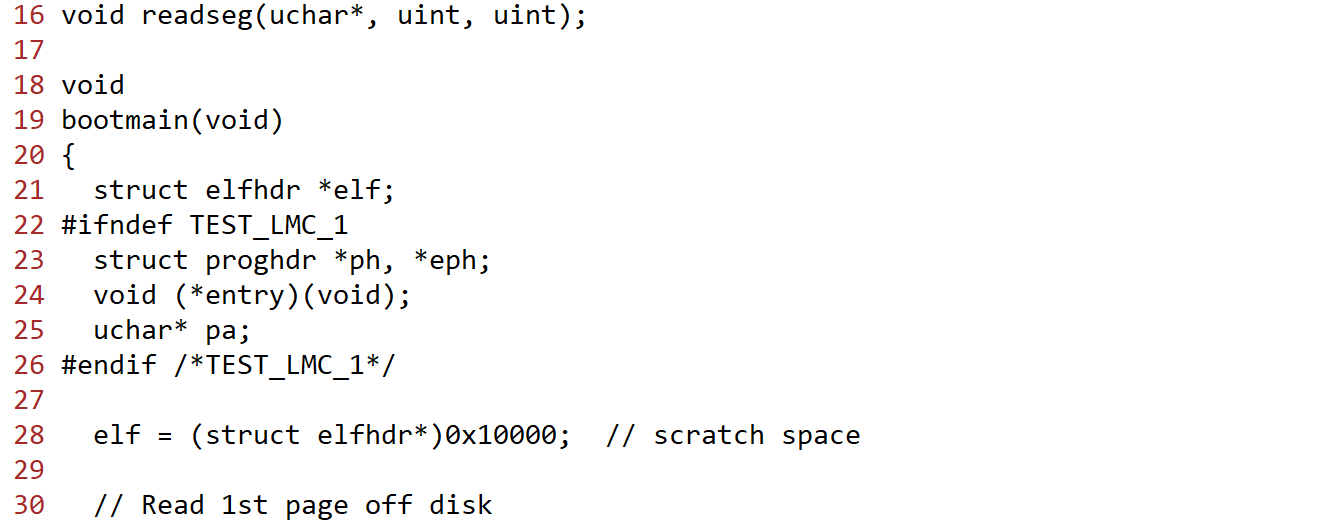


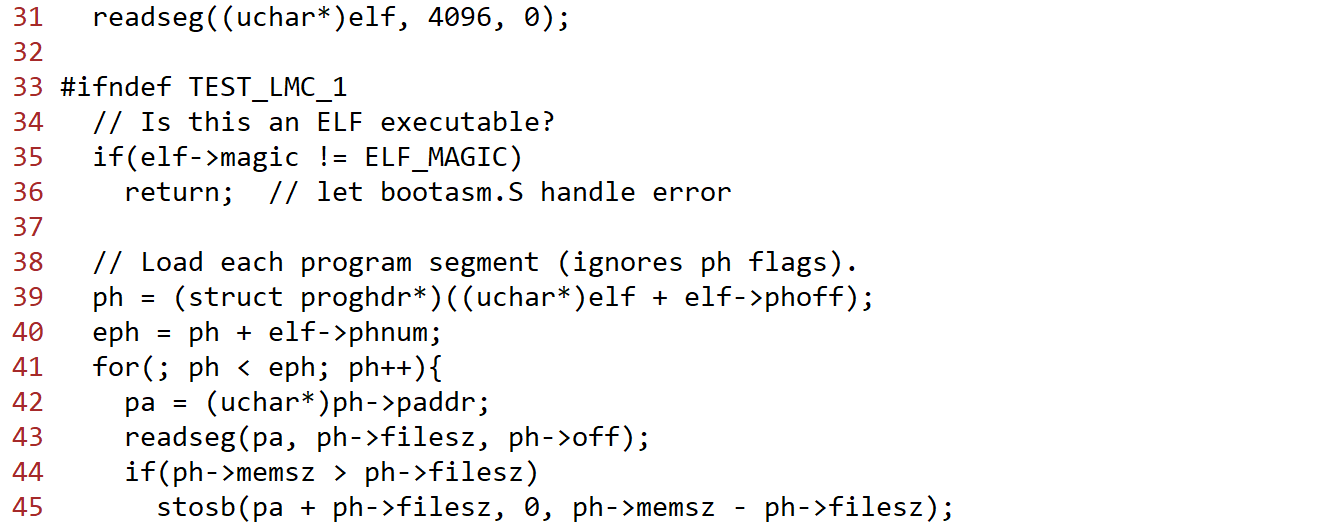


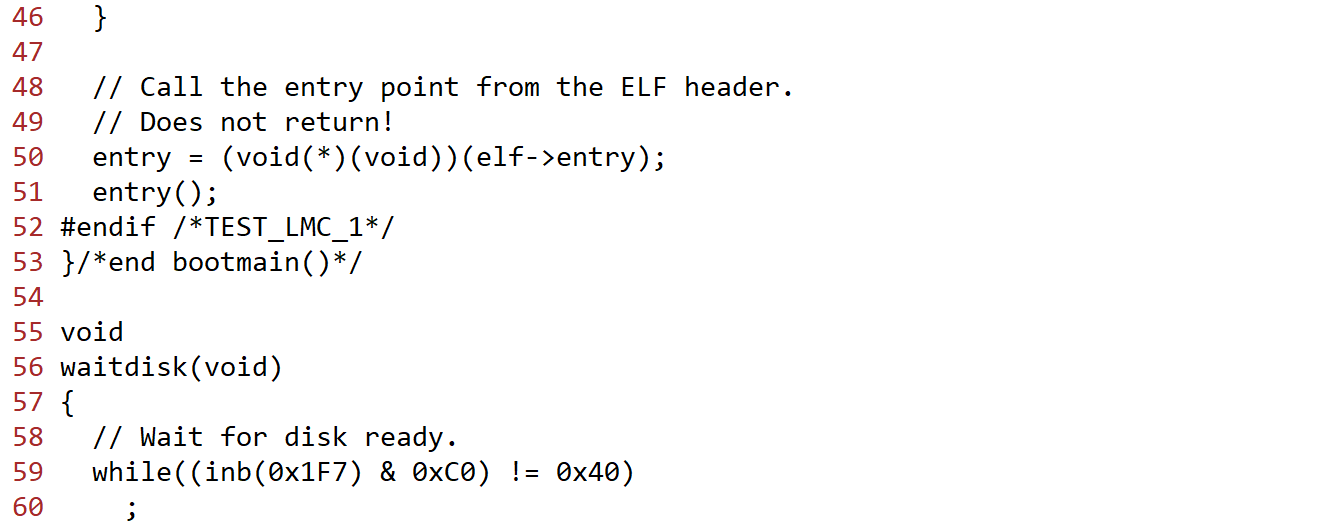


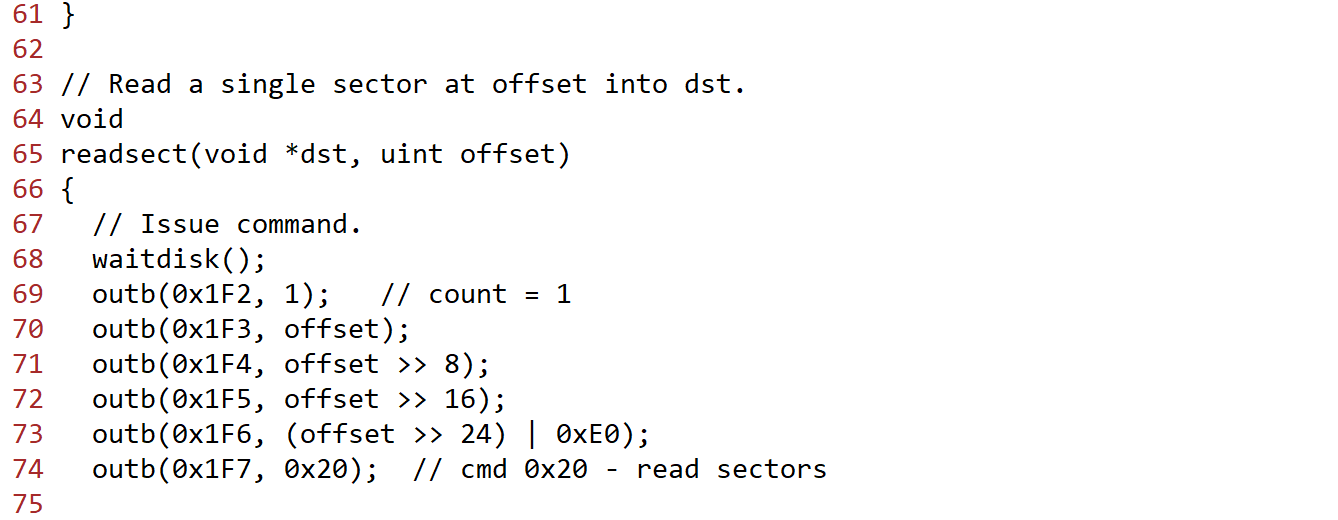
Archivo bootasm.S

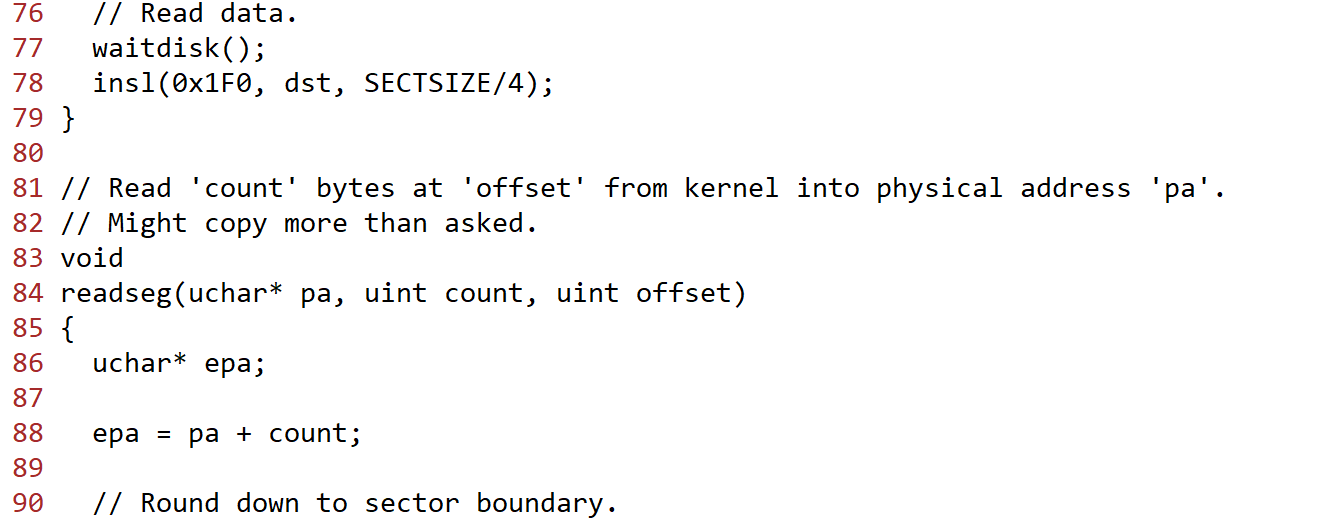


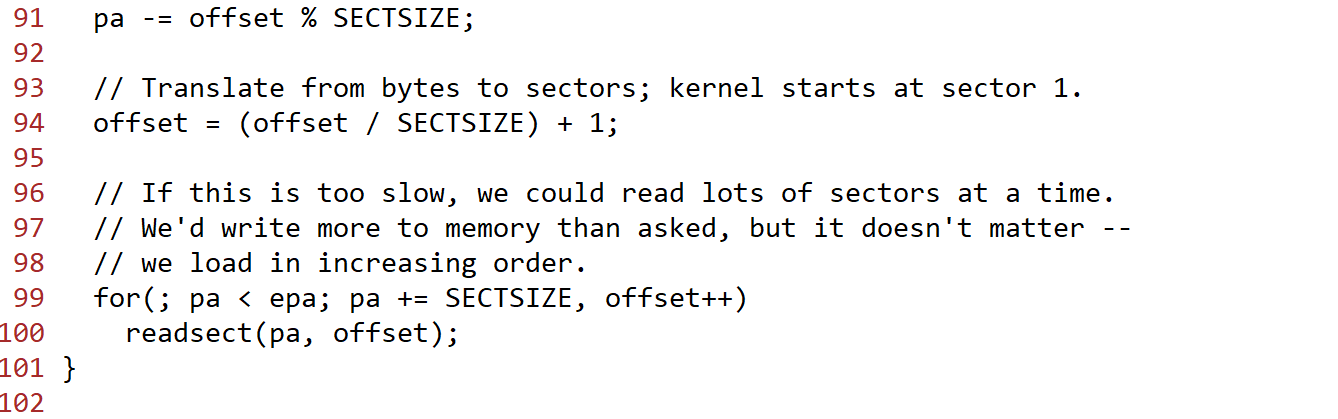






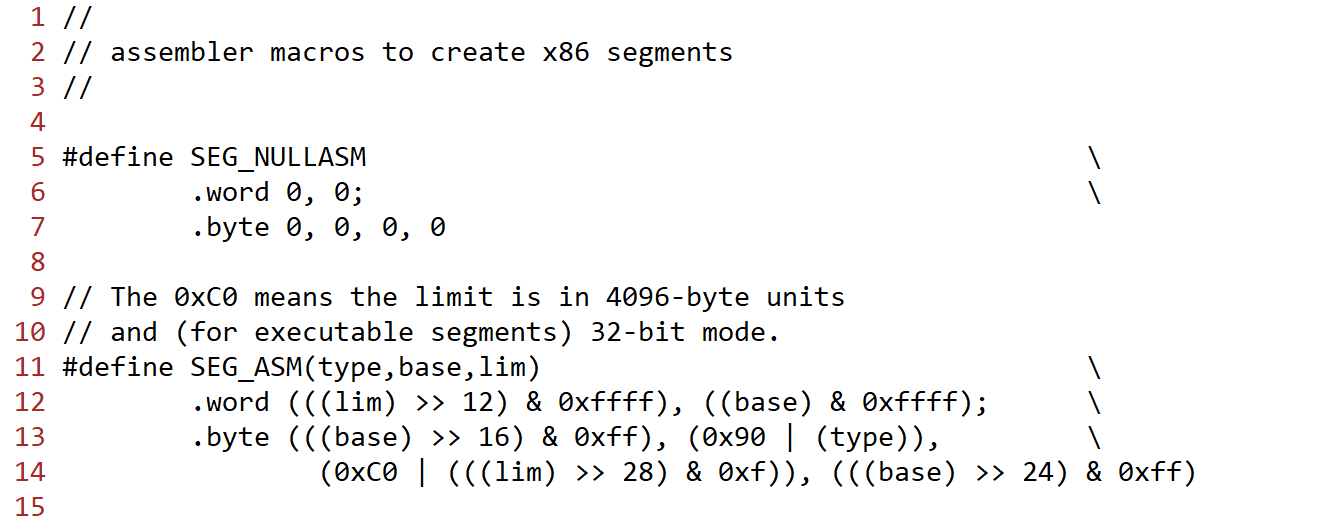


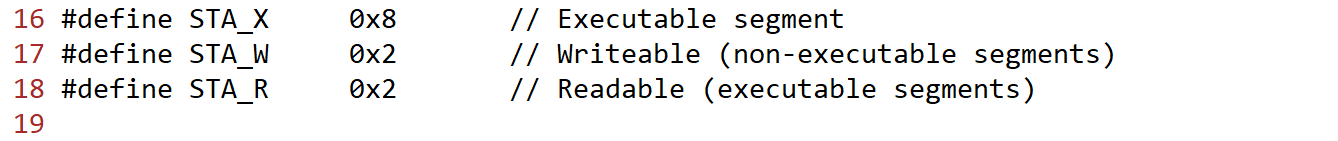




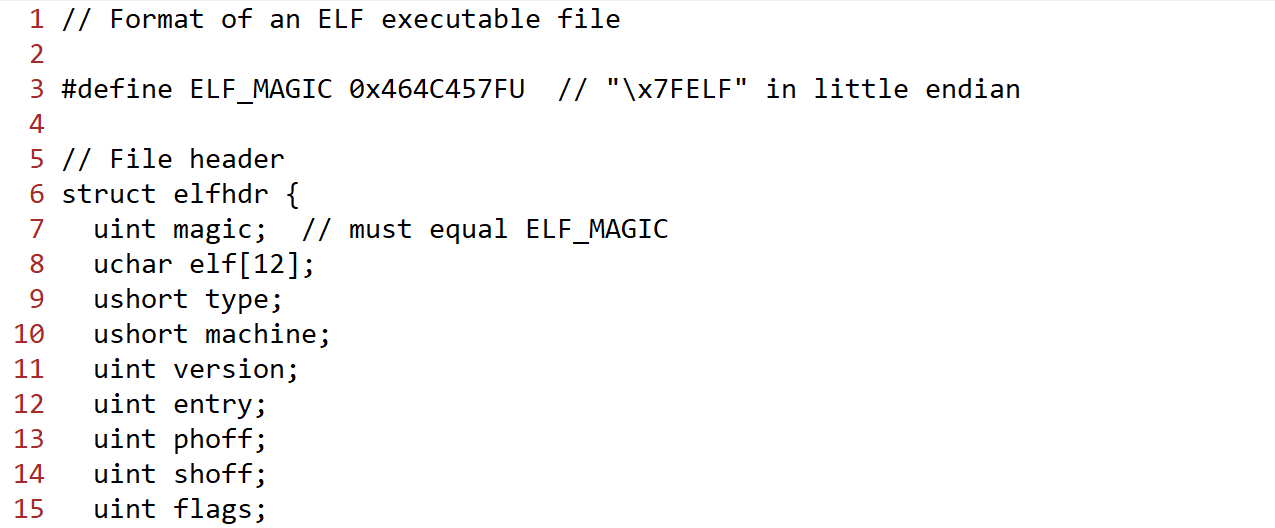
Archivo bootmain.c

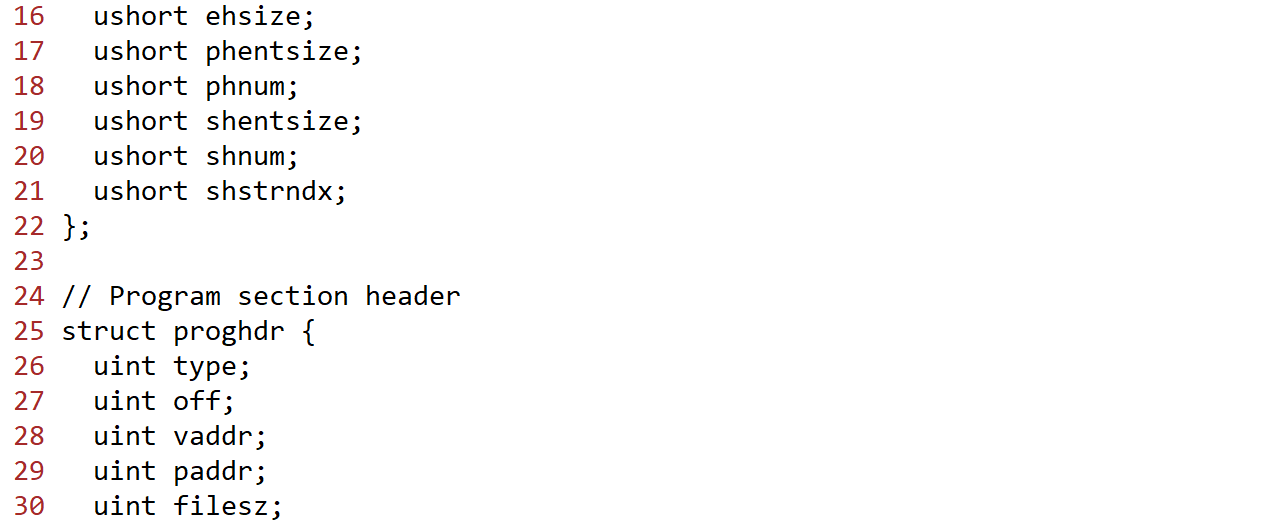
Los archivos de cabecera utilizados son asm.h, elf.h, memlayout.h, mmu.h, types.h, x86.h. Los cuales se incluyen a continuación.

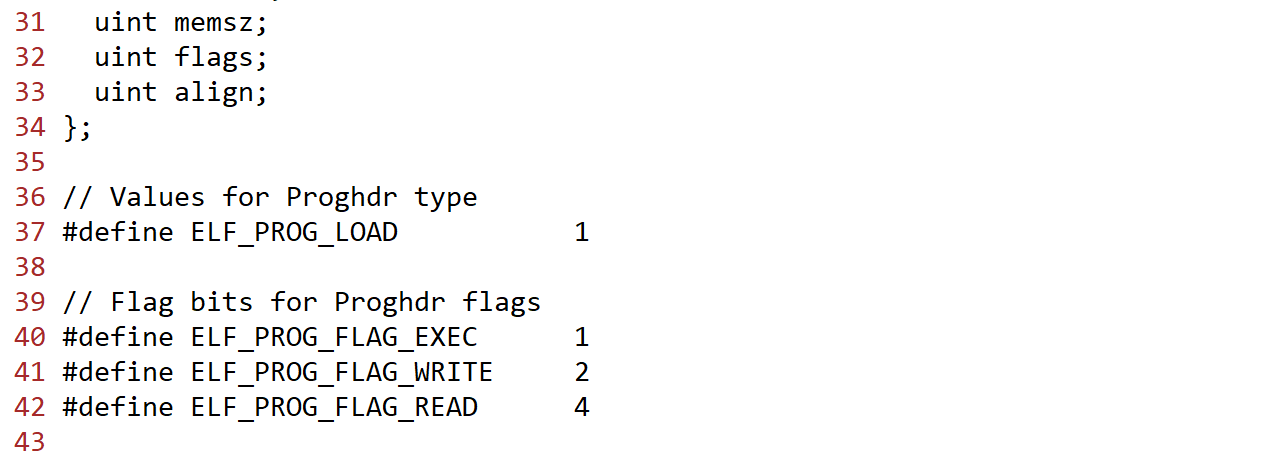




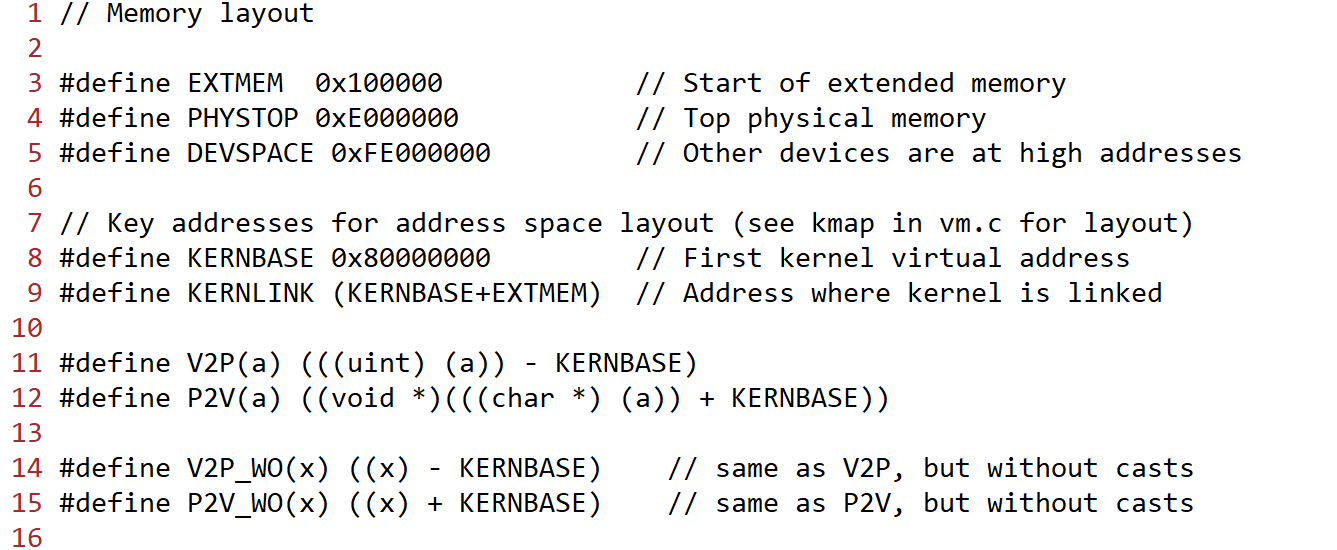
Archivo asm.h



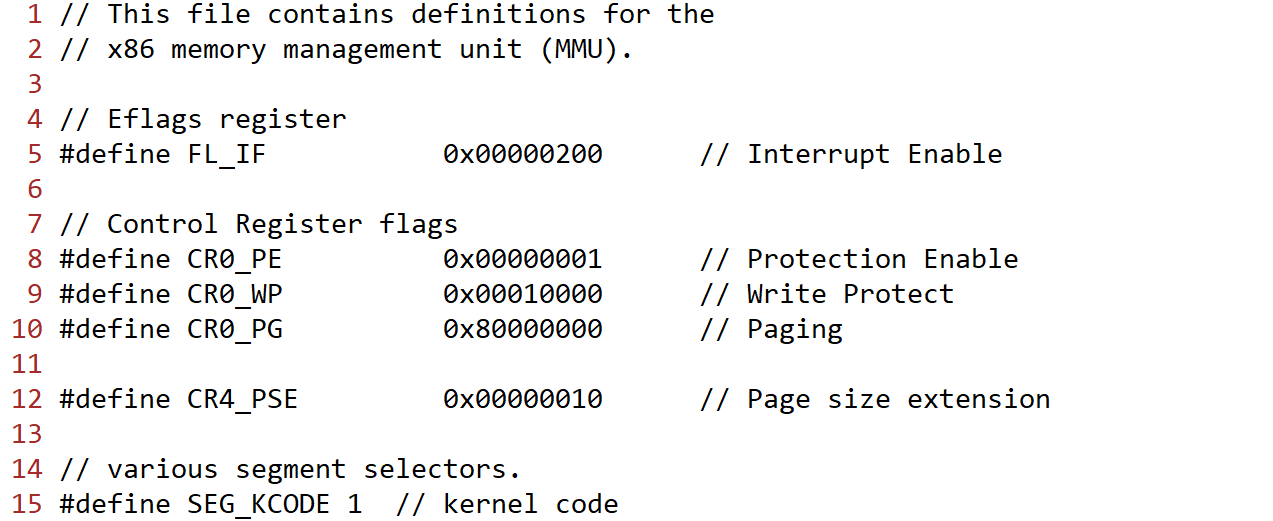


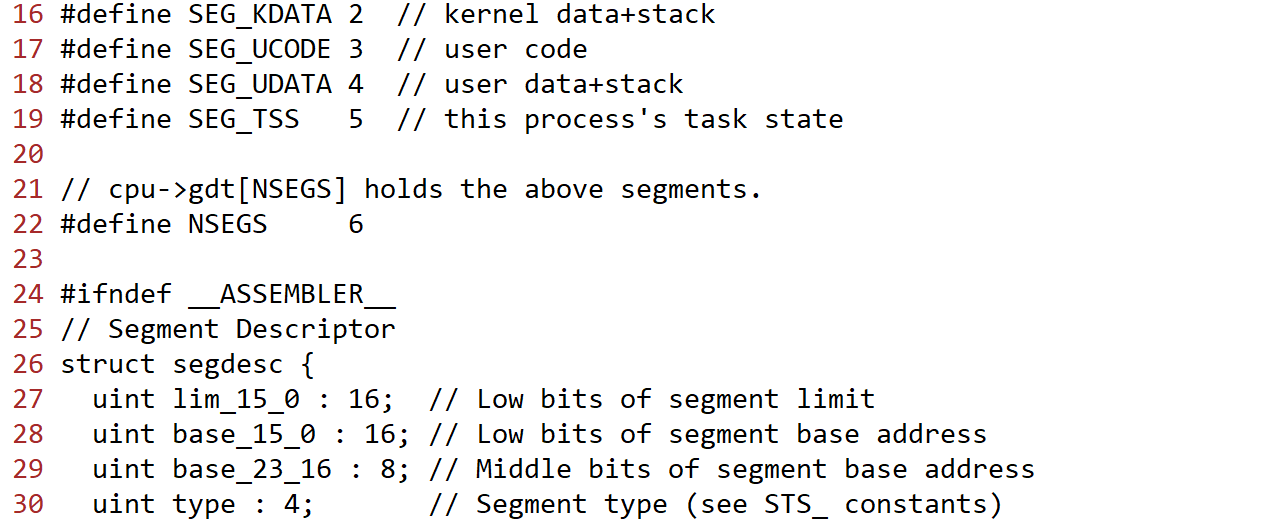


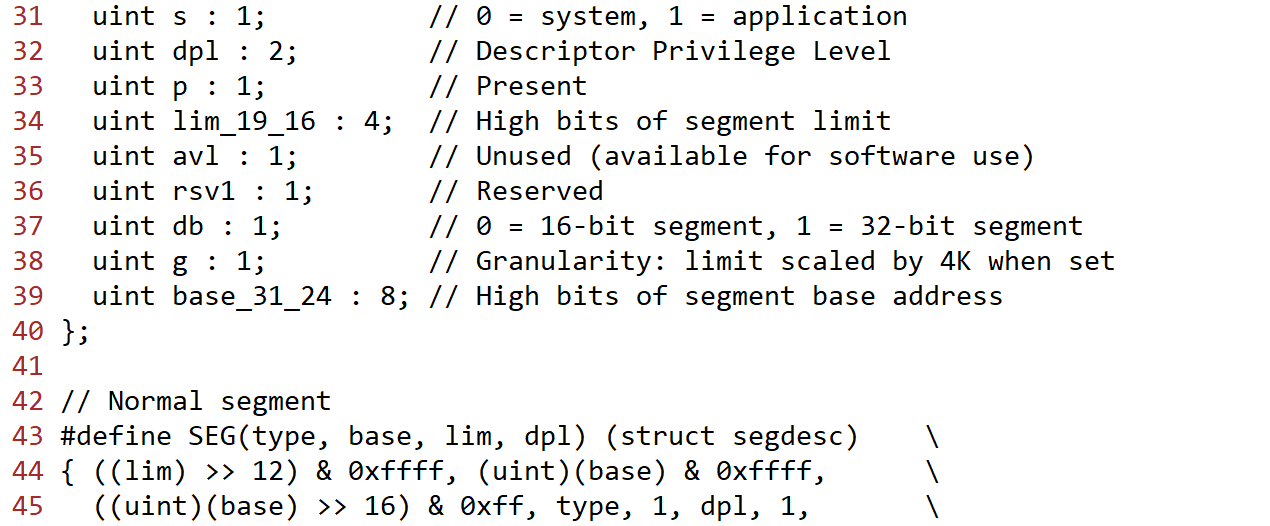
Archivo elf.h

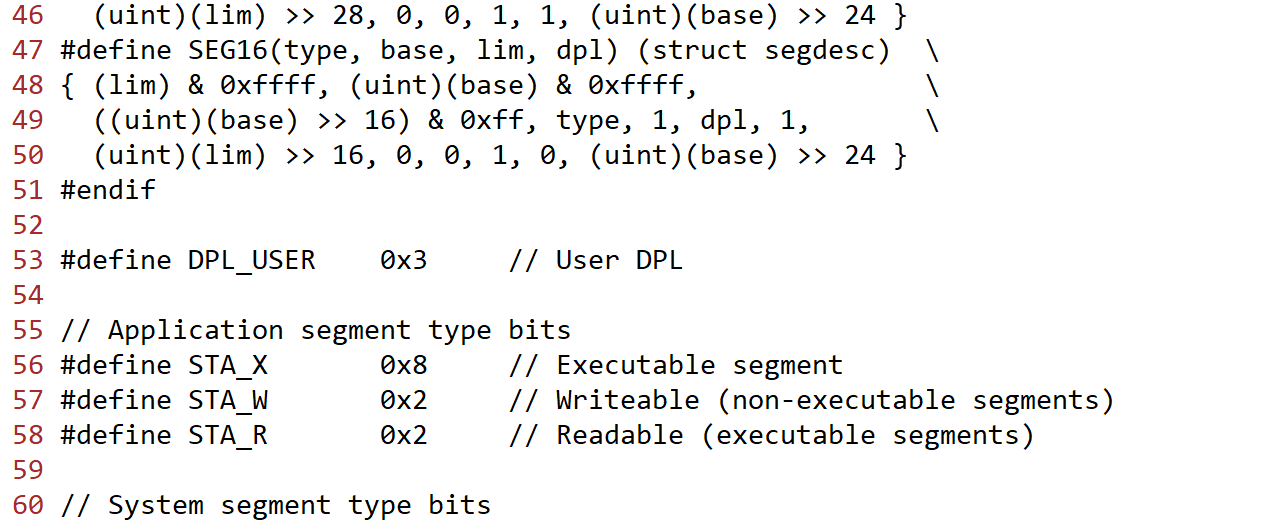


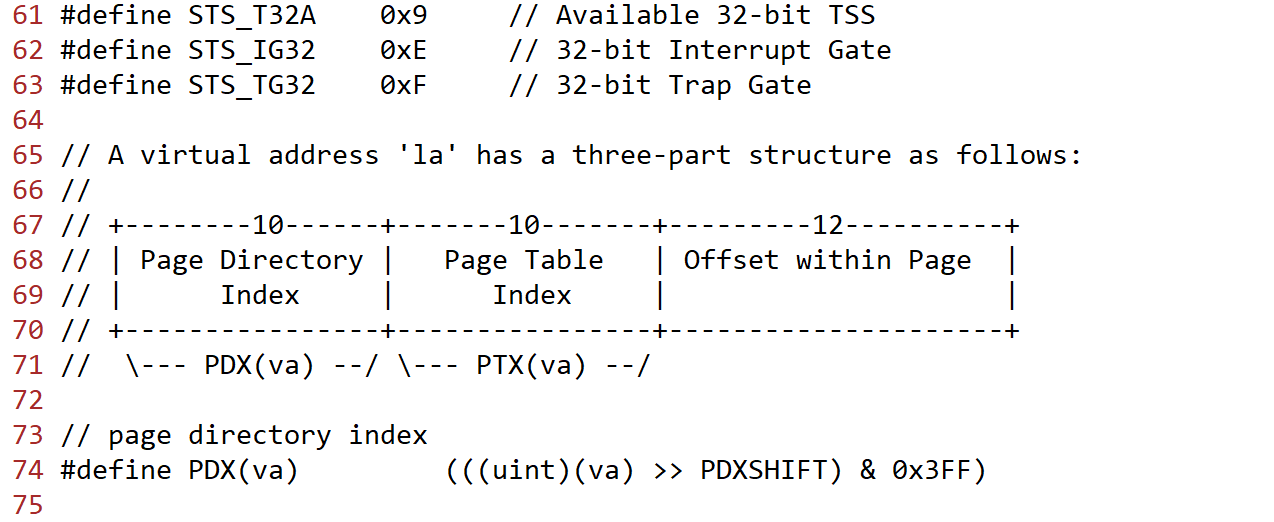
Archivo memlayout.h

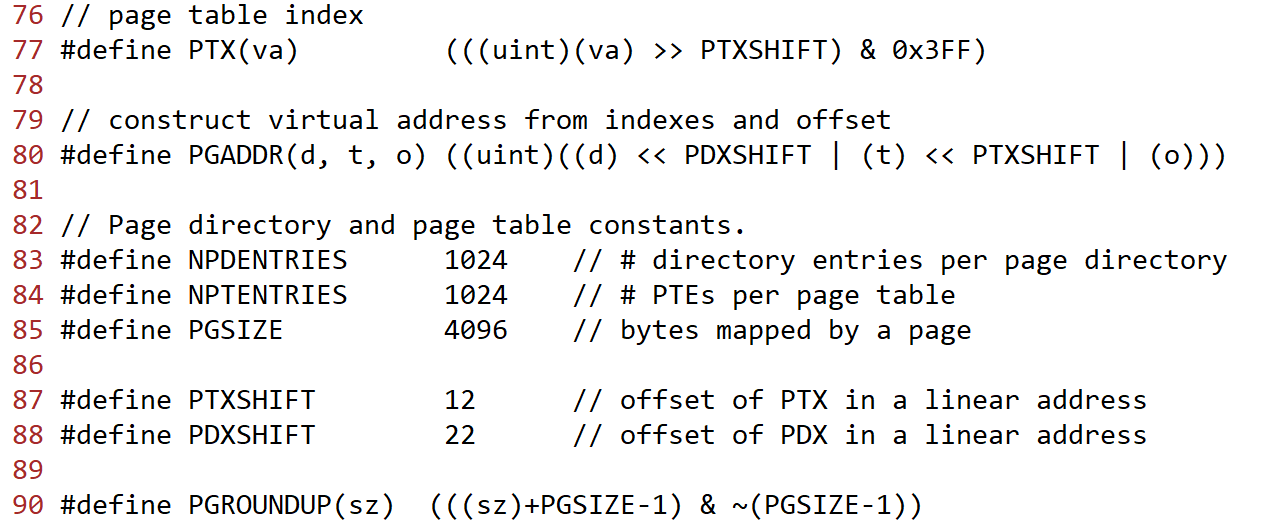


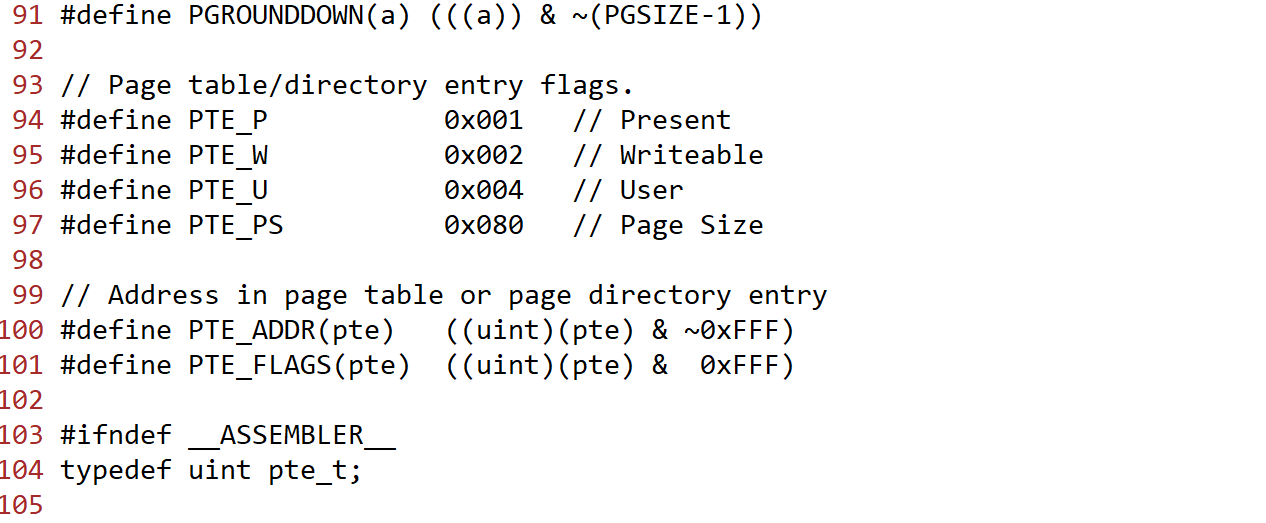


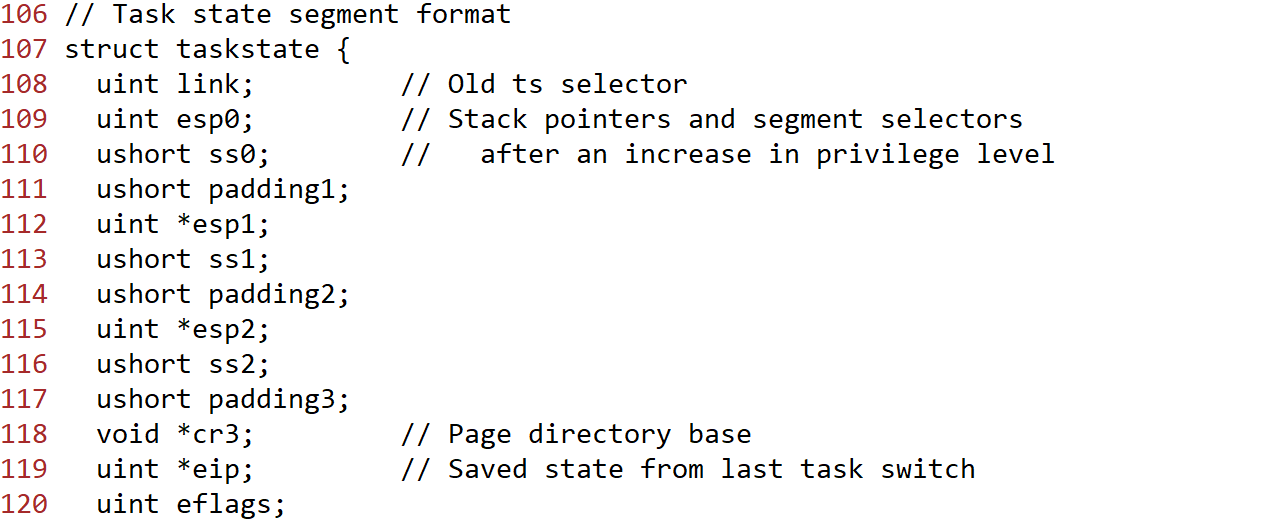


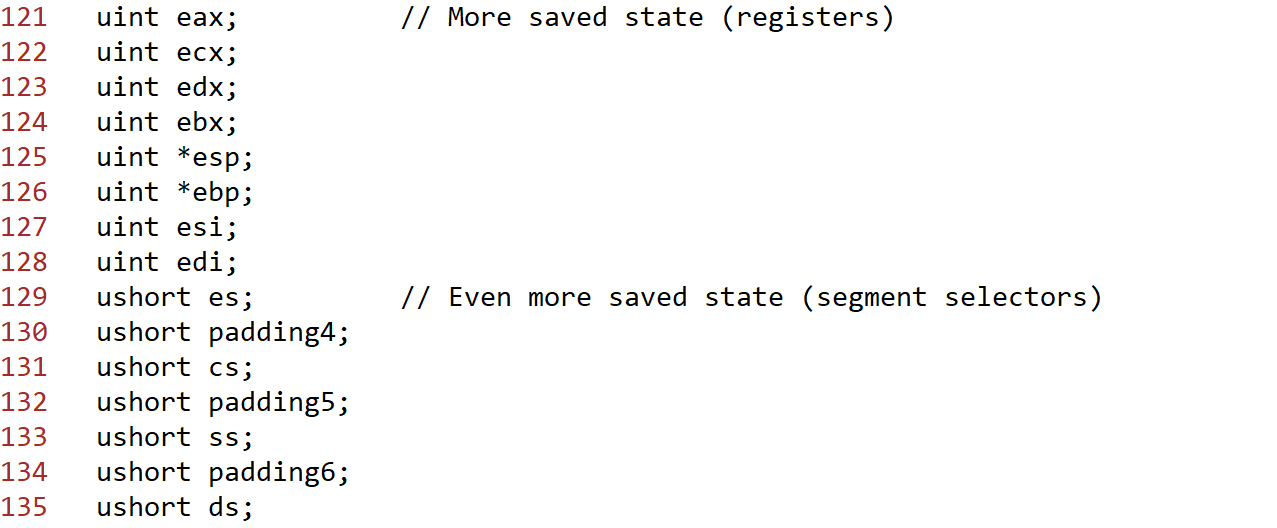


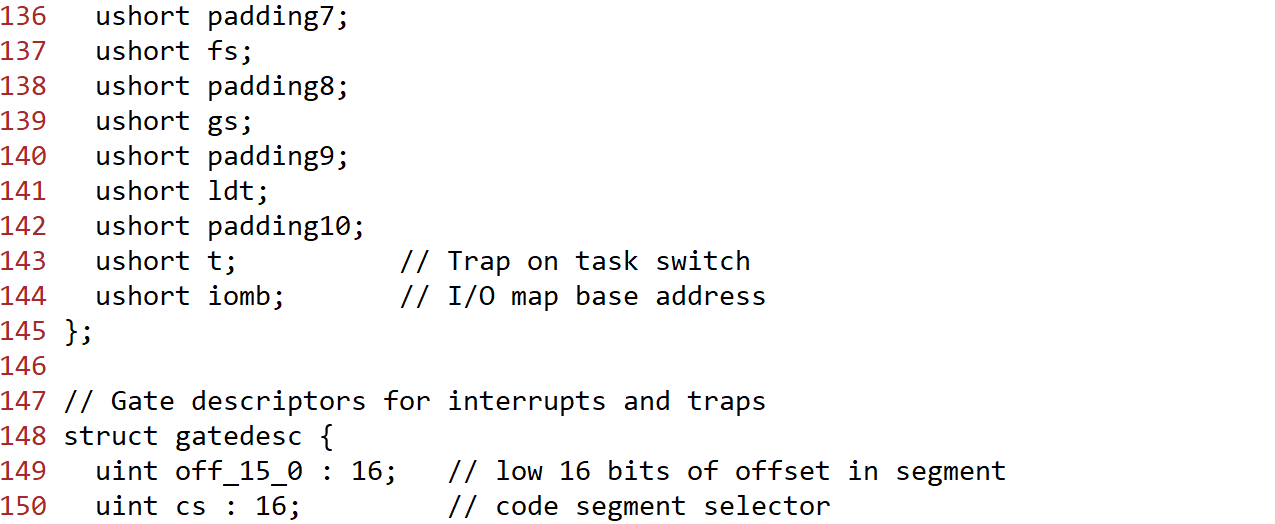


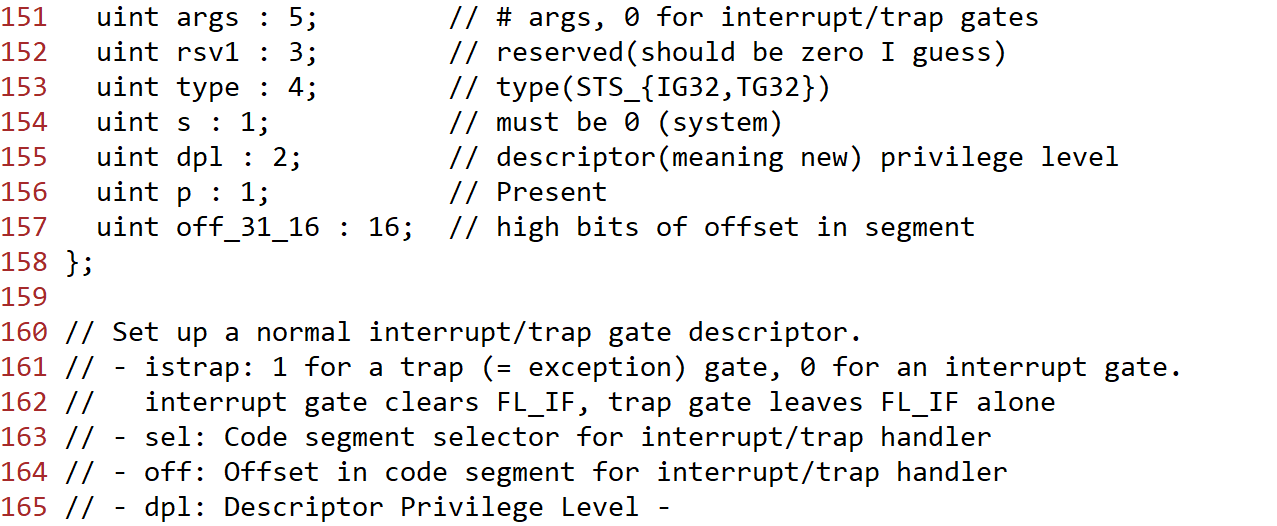


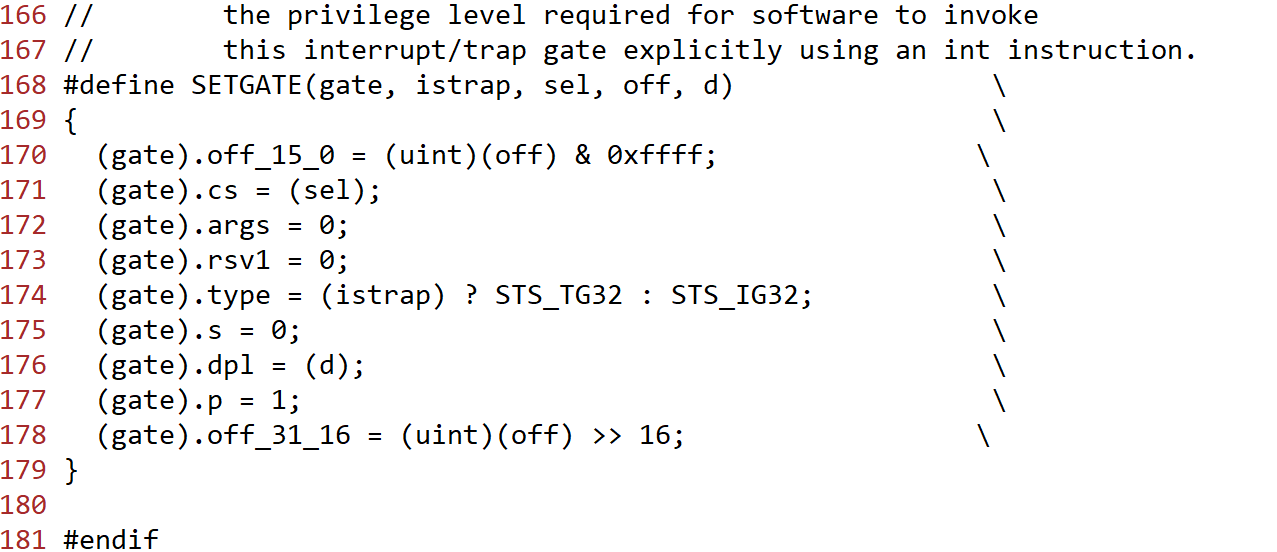








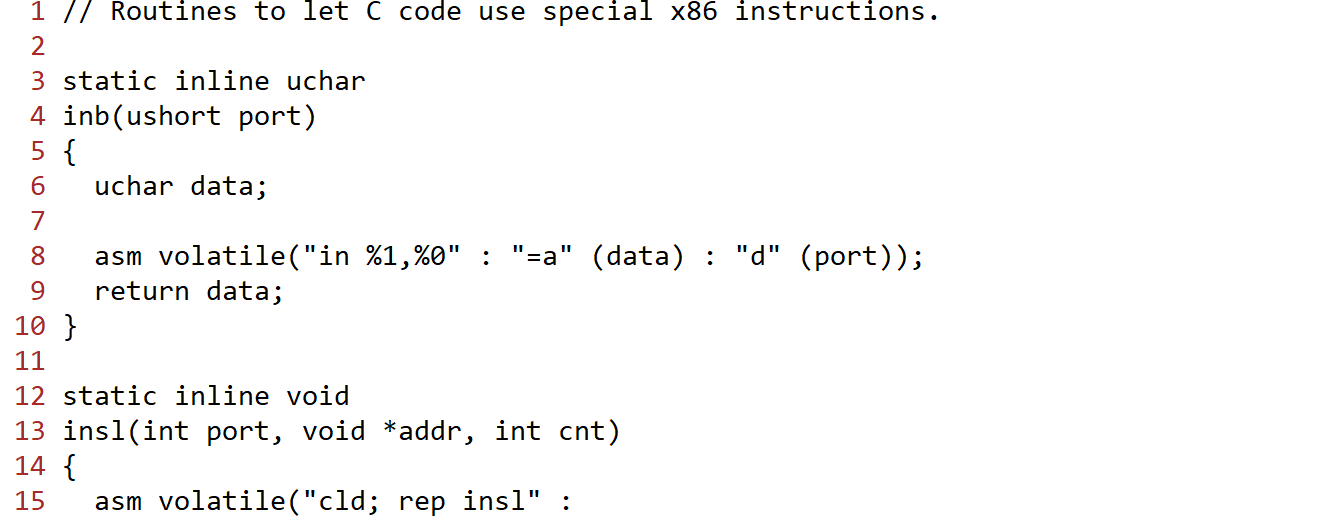


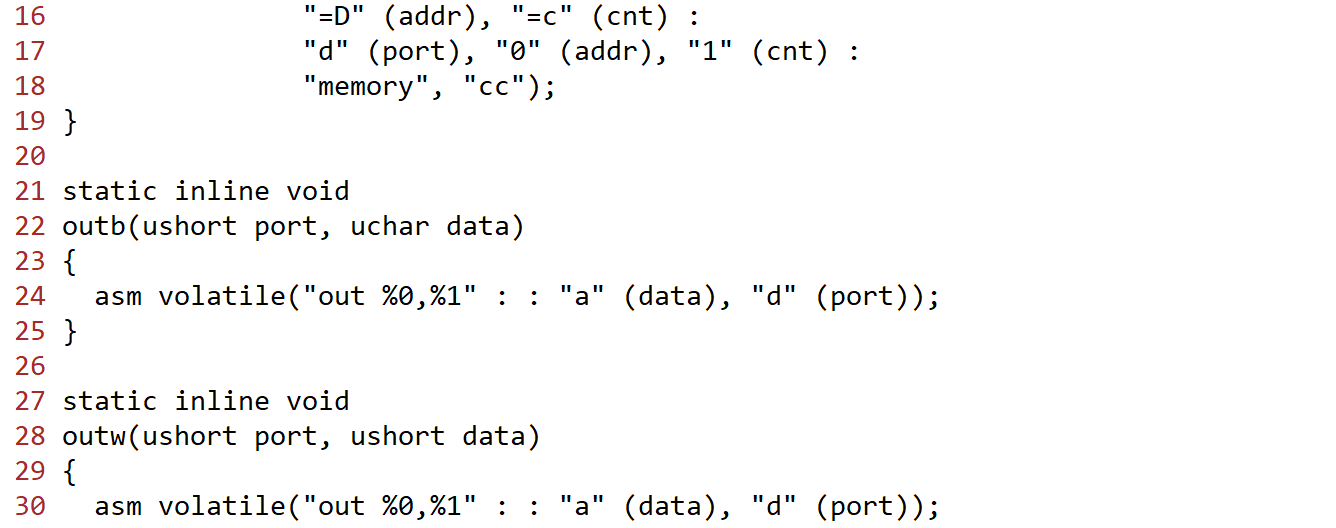


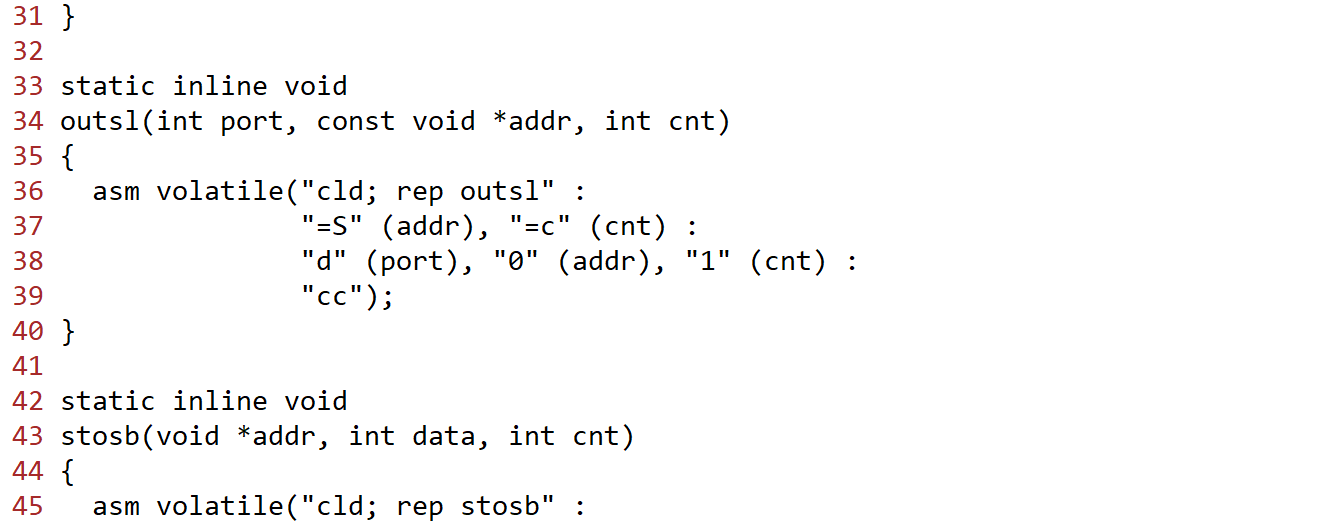
Archivo mm.h

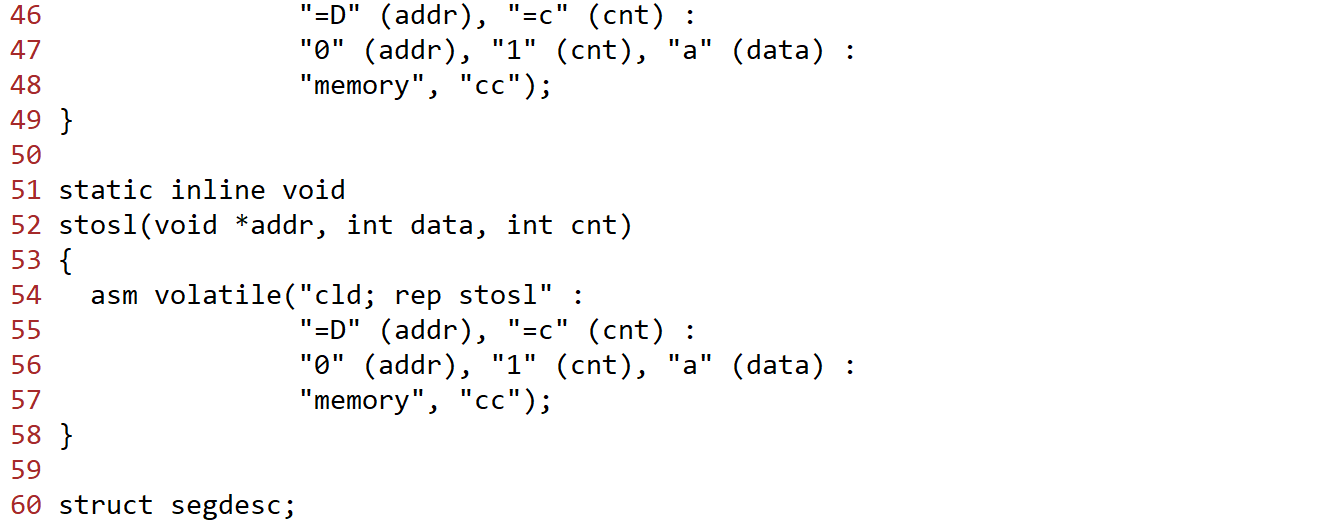


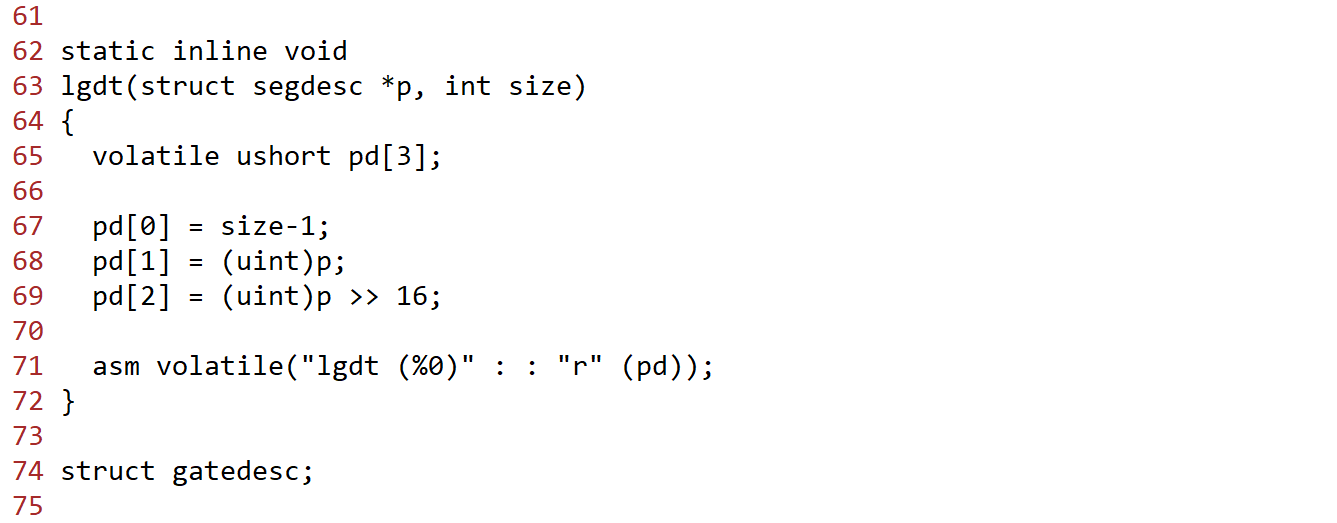
Archivo types.h

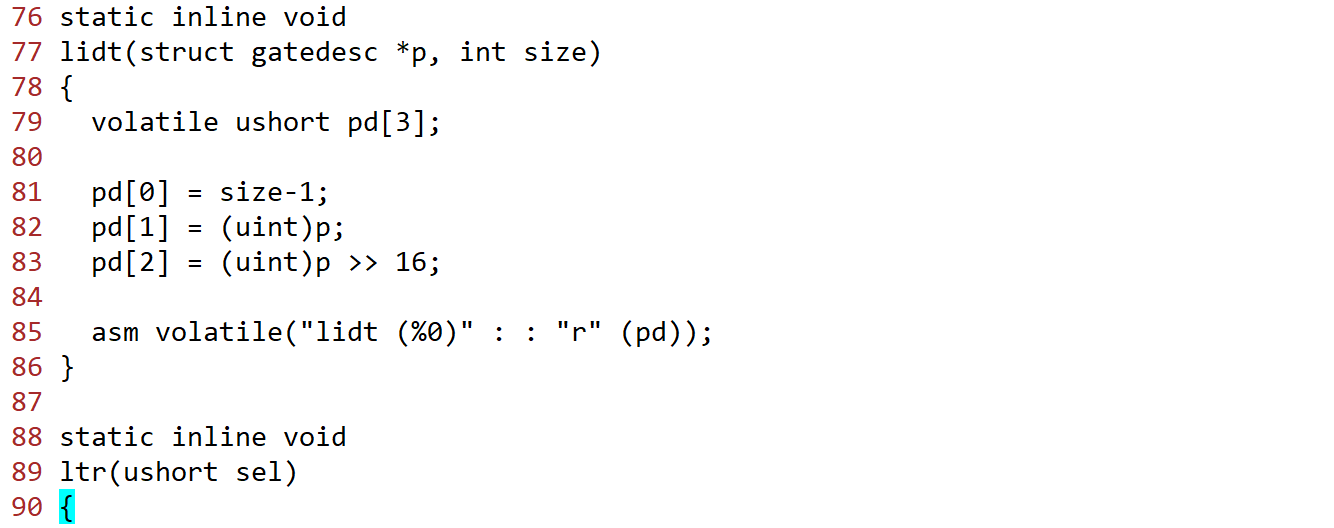


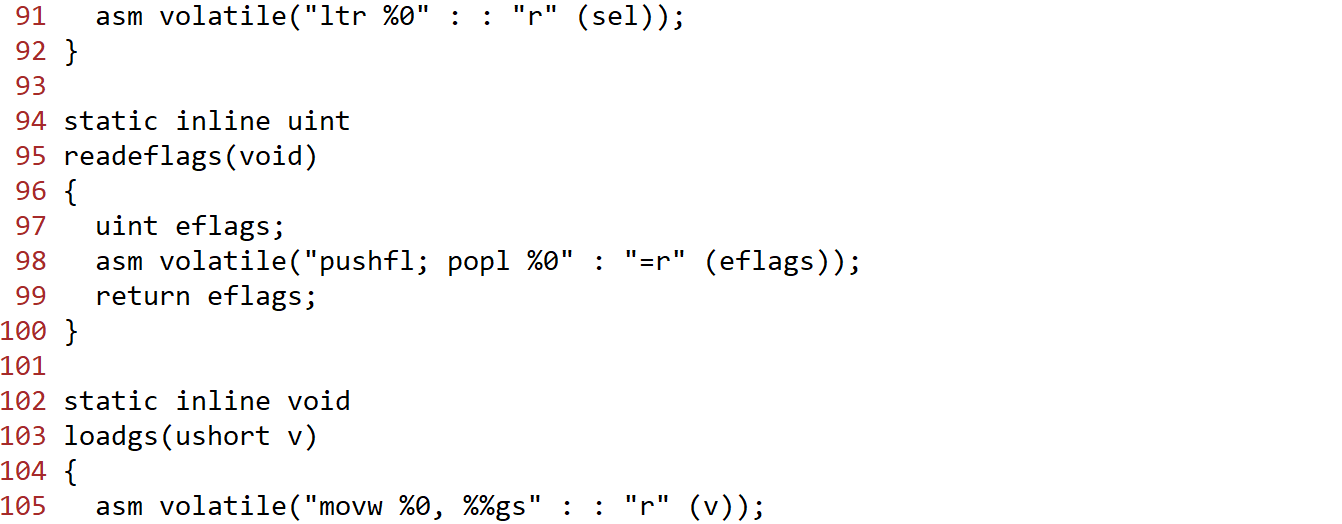


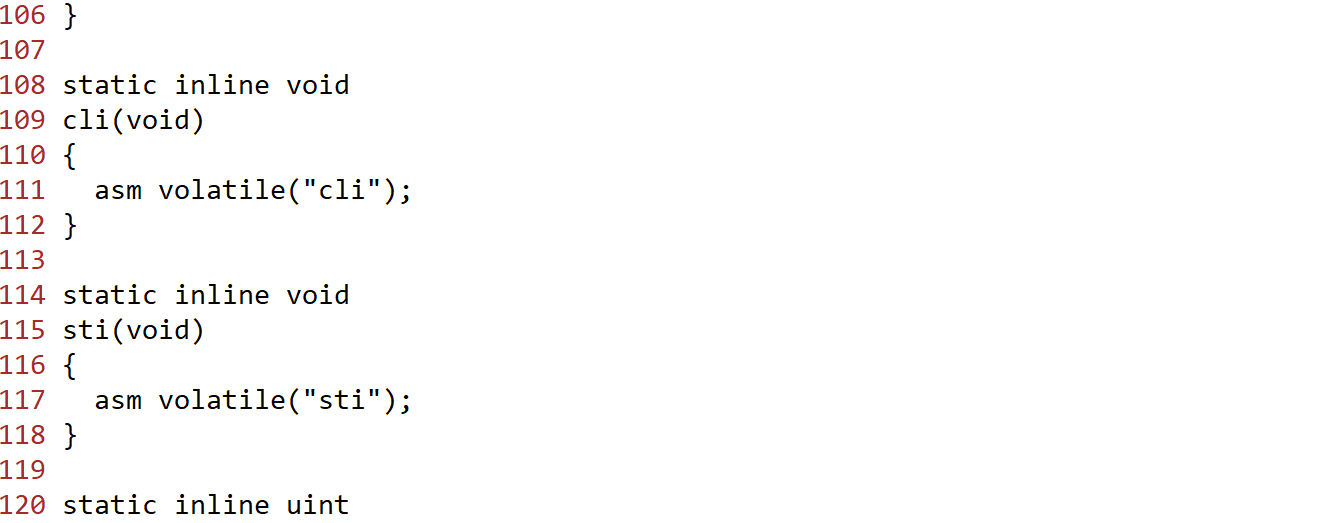


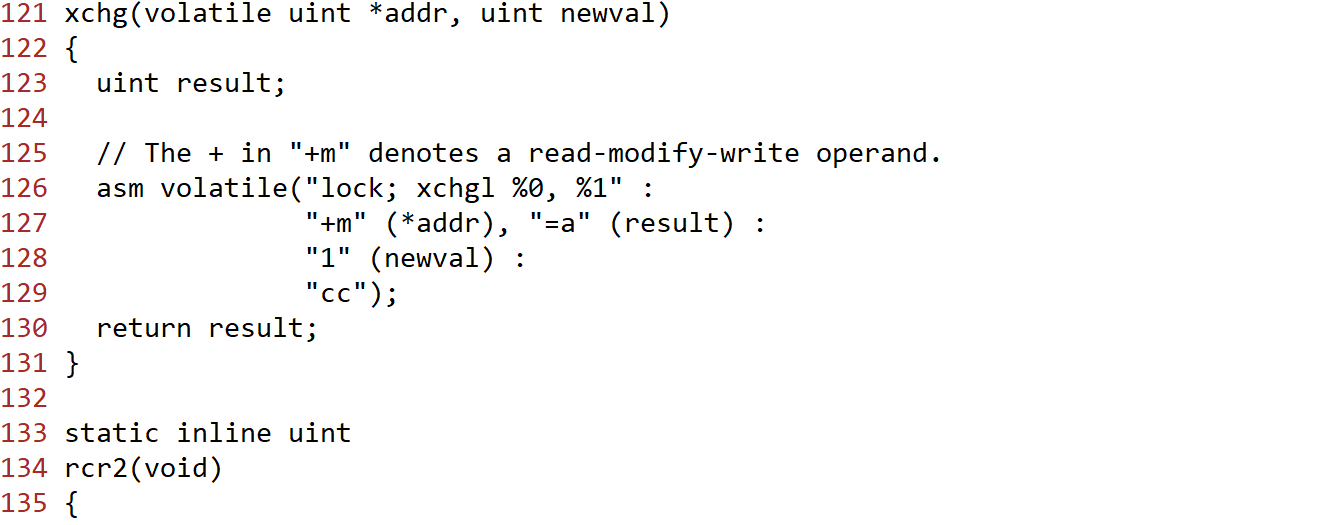


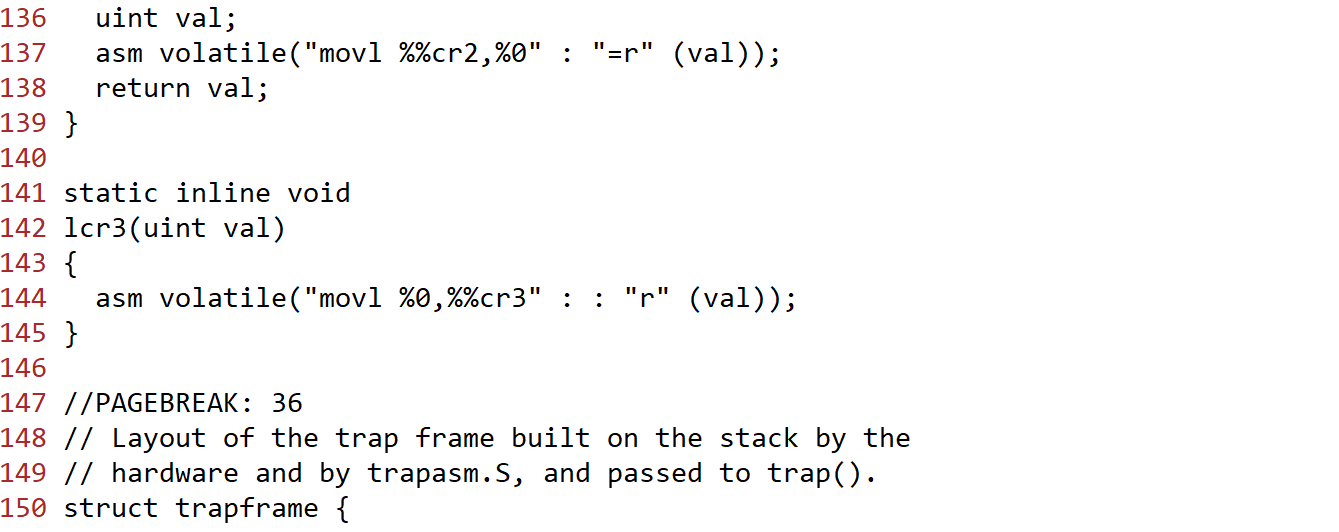


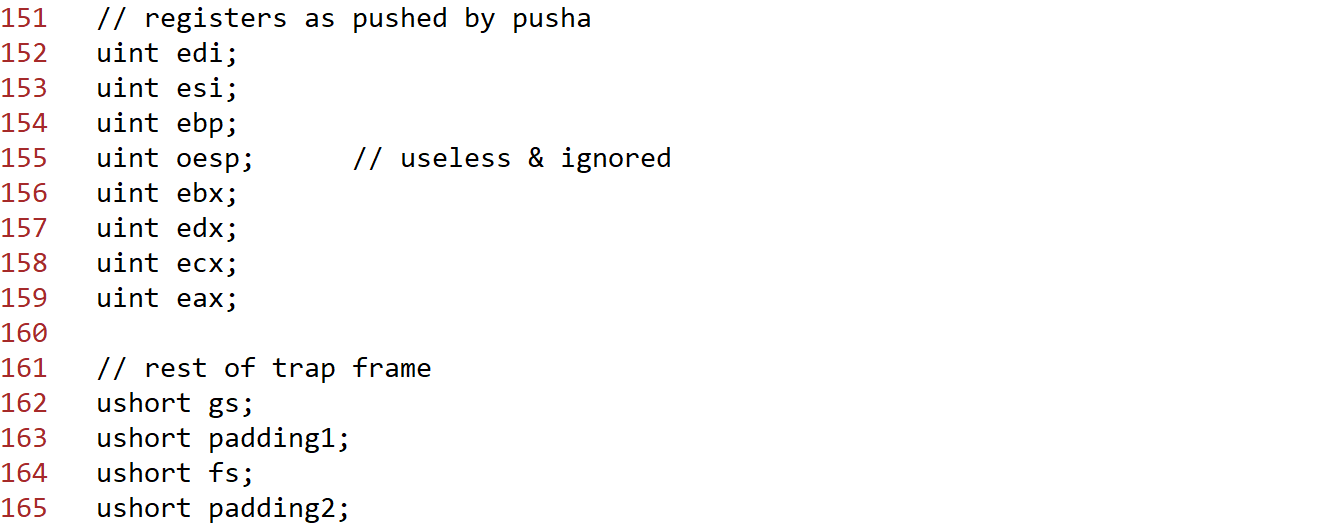


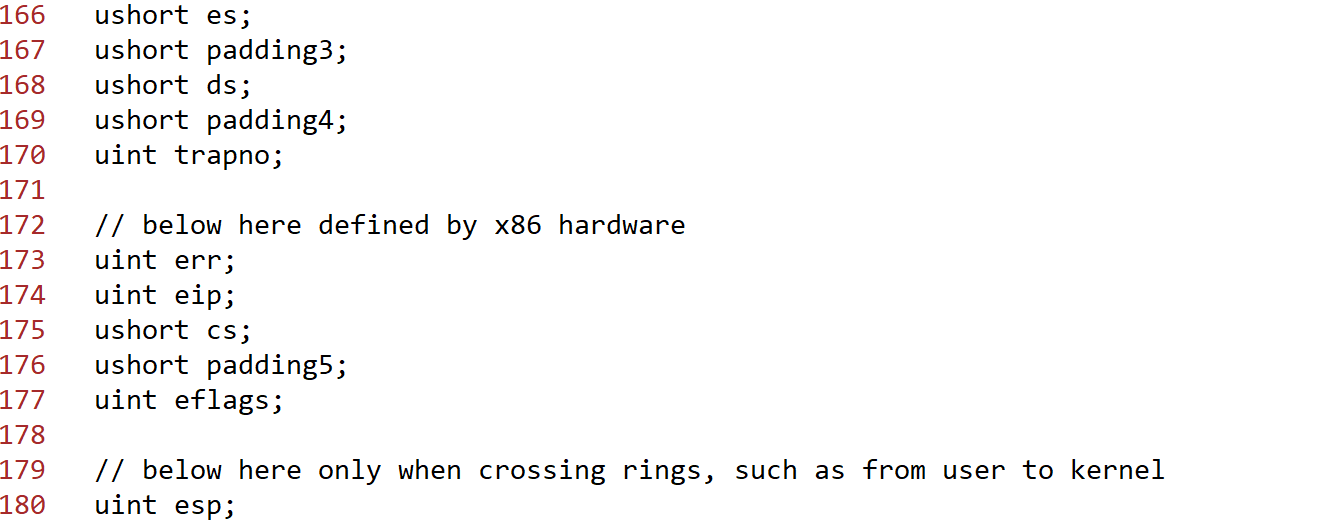








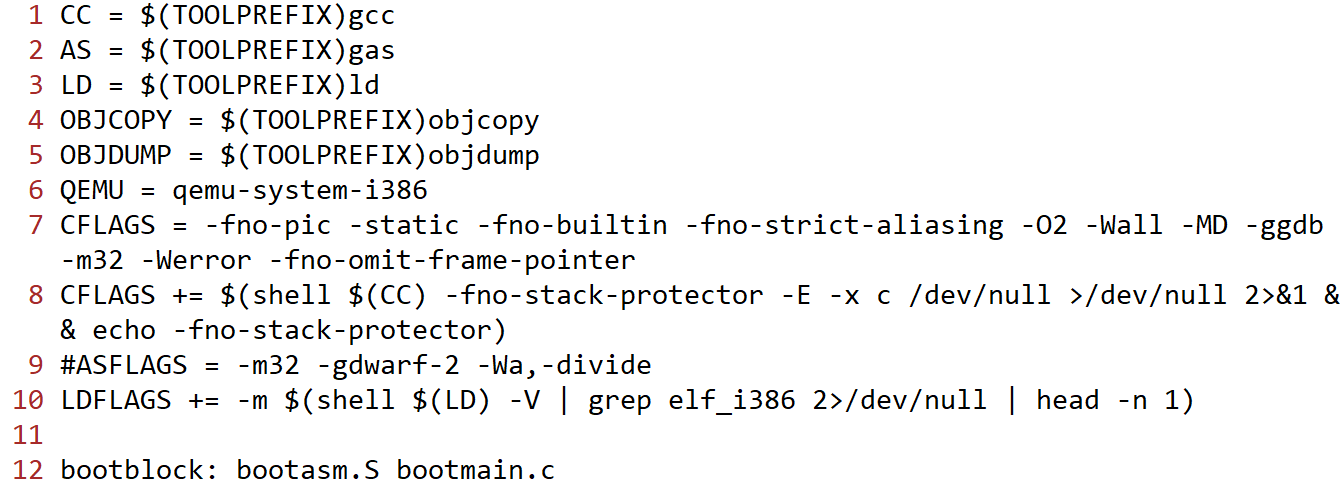


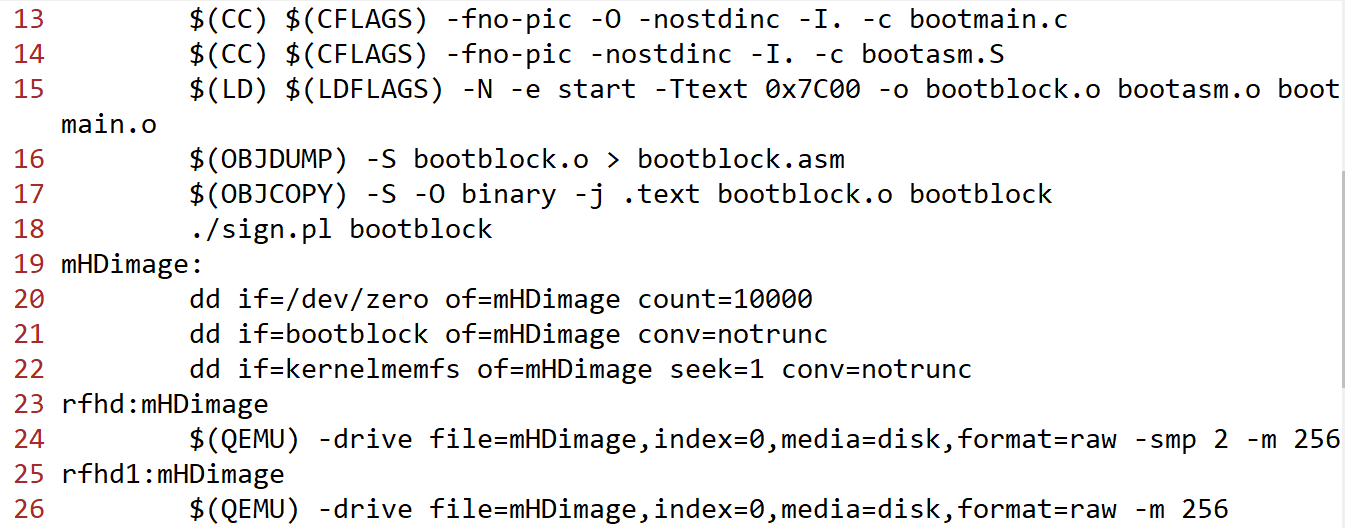


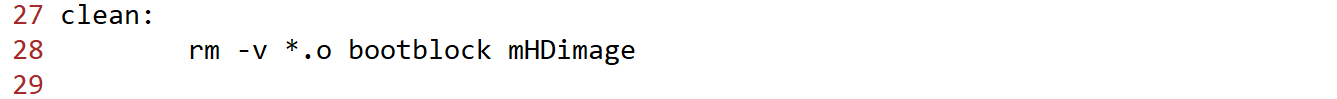


Archivo x86.h

A continuación se incluye un archivo makefile para construir el cargador de arranque, en este caso, se llamará bootblock.







Archivo makefile para construir bootblock