Санкт-Петербургский Политехнический университет

Институт Компьютерных наук и технологий

Кафедра компьютерных систем и программных технологий

Лабораторная работа №0

Дисциплина: Проектирование ОС и компонентов

Тема: Разработка первичных загрузчиков выбранными средствами. Мультизагрузка.

Выполнил студент гр. 13541/4 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. И. Шайтан

(подпись)

Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е. В. Душутина

(подпись)

“\_\_\_\_” \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2017 г.

Санкт-Петербург

2017

**Выполнение работы:**

**Цели работы:**

Изучить и разработать первичный и вторичный загрузчики, с последующей загрузкой выбранного исполняемого файла.

Загрузчик операционной системы — системное программное обеспечение, обеспечивающее загрузку операционной системы непосредственно после включения и начальной загрузки. Первым делом, после запуска компьютера, запускается BIOS. BIOS проводит инициализацию компьютера, определяет загрузочное устройство и копирует первый сектор устройства (MBR 512 байт) в оперативную память по адресу 0x7C00.

Далее запускается первичный загрузчик. Из-за ограничения на его размеры (512 байт) и сложность процесса загрузки, основной задачей первичного загрузчика является запуск вторичного загрузчика. Первичный загрузчик находит логический раздел, копирует первый сектор логического раздела в память по адресу 0x7C00 и передает ему управление.

Вторичный загрузчик занимается загрузкой ОС. Он находит на носителе образ ядра, загружает его в память, инициализирует необходимые регистры и передает управление ядру ОС.

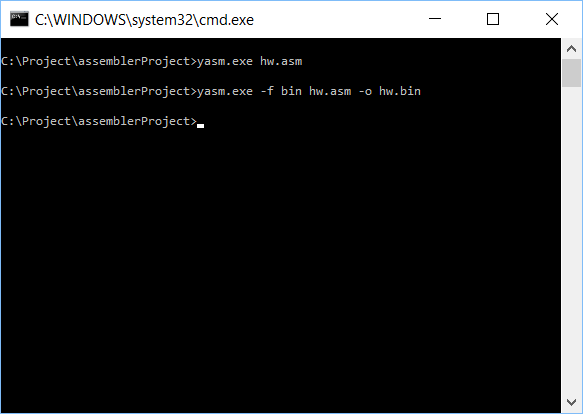
Для выполнения данной работы необходимо подготовить эмулятор Qemu. Виртуальная машина используется для того, чтобы не повредить систему на основной машине. Для создания виртуальных машин использовалась VMWare Workstation. Далее, связываем flash-накопитель с VMWare, для этого, при создании пустой гостевой машины, в качестве местоположения указываем наш flash-накопитель. После завершения создания гостевого хоста, среди прочих файлов, на flash-накопителе будет находиться Bootloaders.vmdk. Это и есть «Жесткий диск», с которого будет загружаться система.

Тестовый загрузчик

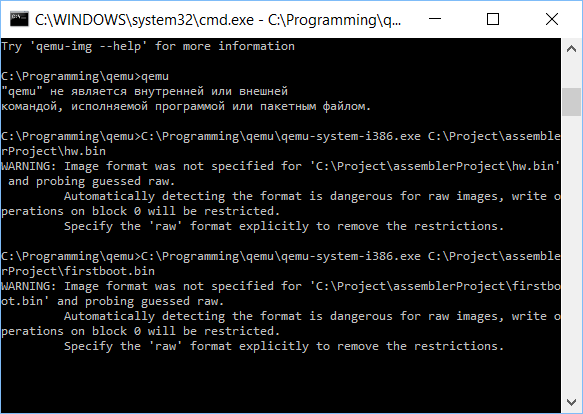
По традиции, для начала, создаем тестовый загрузчик «Hello world». Единственной задачей такого загрузчика будет вывод какого-то текста на экран. Код загрузчика написан на языке assembler. Листинг загрузчика приведен ниже:

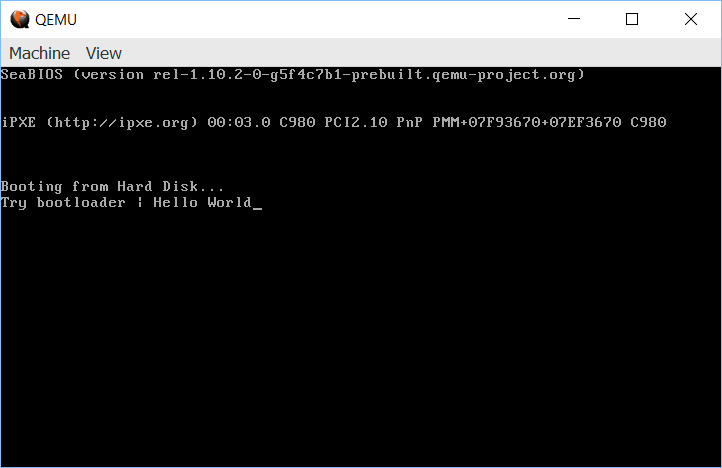
|  |
| --- |
| [BITS 16]  [ORG 0x7C00]  mov si, HelloString ; Записываем в стек указатель на строку  call PrintString ; Вызов процедуры печати строки  jmp $ ; Бесконечный цикл  PrintCharacter: ; Процедура печати символа  mov ah, 0x0E ; Флаг того, что нам нужно вывести на экран один символ  mov bh, 0x00 ; Номер страницы  mov bl, 0x07 ; Флаг того, что выводится светлый текст на черном фоне  int 0x10 ; Вызов прерывания видео  ret ; Возращение к вызванной процедуре  PrintString: ; Процедура печати строки  next\_character:  mov al, [si] ; Берем один байт из строки и записываем его в регистр AL  inc si ; Увеличиваем указатель SI  or al, al ; Проверка конца строки  jz exit\_function  call PrintCharacter ; Печатаем символ  jmp next\_character  exit\_function:  ret  ;Data  HelloString db 'Try bootloader | Hello World', 0  times 510 - ($ - $$) db 0 ;Заполняем оставшиеся байты нулями  dw 0xAA55 ;Добавляем сигнатуру в конец загрузчика |

С помощью компилятора yasm скомпилировали программу:



После запуска qemu, на экране отобразится следующее сообщение:





**Первичный загрузчик**

Работа первичного загрузчика будет производиться по следующему алгоритму:

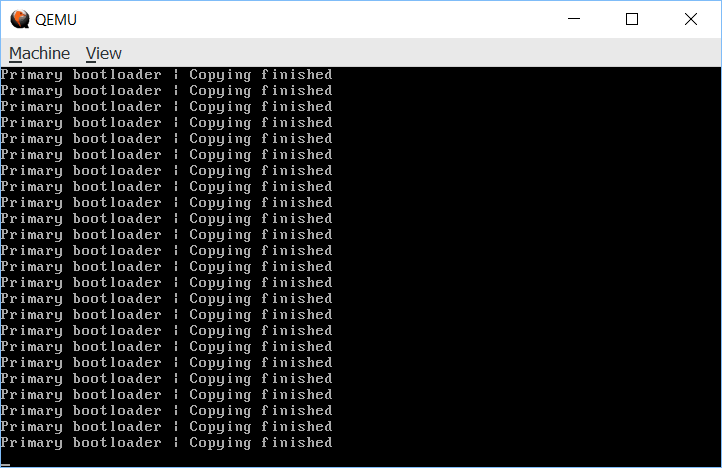
1. Инициализация регистров, стека

2. Копирование загрузочного сектора по новому адресу (0x1000) и передача ему управления

3. Вывод сообщения (Copying finished), сигнализирующего об успешной загрузке по новому адресу

|  |
| --- |
| [BITS 16]  [ORG 0x7C00]  ;Начало работы первичного загрузчика  cli ; запрещение аппаратных прерываний  mov ax,cs ; сегмент кода  mov ds,ax ; сегмент данных  mov ss,ax ; сегмент стека  mov es,ax ; дополнительный сегмент данных  mov sp,7c00h ; Инициализация стека  mov bx,7c00h ; Адрес загрузки  ;Копируем себя по адресу 0x1000  mov si,7c00h ; Копируем, начиная с адреса загрузки  mov di,1000h ; Адрес, куда производим копирование  mov cx,200h ; Сколько байт копируем (512 байт)  rep movsb  jmp 0000h:101fh ; Передаем управление по новому адресу  mov si, TestString ;Записываем в стек указатель на строку  call PrintString ;Вызов процедуры печати строки  PrintCharacter: ;Процедура печати символа  mov ah, 0x0E ;Флаг того, что нам нужно вывести на экран один символ  mov bh, 0x00 ;Номер страницы  mov bl, 0x07 ;Флаг того, что выводится светлый текст на черном фоне  int 0x10 ;Вызов прерывания видео  ret ;Возращение к вызванной процедуре  PrintString: ;Процедура печати строки  next\_character:  mov al, [si] ;Берем один байт из строки и записываем его в регистр AL  inc si ;Увеличиваем указатель SI  or al, al ;Проверка конца строки  jz exit\_function  call PrintCharacter ;Печатаем символ  jmp next\_character  exit\_function:  ; загрузка загрузочной записи активного раздела  mov si, DAP  mov ah, 0x42 ; функция  mov dl, 0x80 ; номер диска  int 0x13    jmp 0000h:7c00h ; переходим ко вторичному загрузчику      DAP:  dw 0x10, 0x02, 0x7c00, 0x00 ; Считываем 2 сектора и помещаем в оперативную память по адресу 0x7c00  dd 0x0800 ;абсолютный адрес первого сектора  dd 0x0000  TestString db 'Primary bootloader | Copying finished', 13, 10, 0  times 510 - ($ - $$) db 0 ;Заполняем оставшиеся байты нулями  dw 0xAA55 ;Добавляем сигнатуру в конец загрузчика |

Результат работы приведен ниже



**Список используемых источников:**

1. <http://www.qemu.org/> - QEMU the FAST! processor emulator
2. <http://yasm.tortall.net/> - The Yasm Modular Assembler Project