|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01 Информатика и вычислительная техника**

БАКАЛАВРСКАЯ ПРОГРАММА **09.03.01\_03 Вычислительные машины, комплексы,**

**системы и сети**

**ОТЧЕТ ПО ПРОИЗВОДСТВЕННОЙ ПРАКТИКЕ**

|  |  |
| --- | --- |
| Тип практики | Технологическая практика |

|  |  |
| --- | --- |
| Название  предприятия | «название» **/** НУК ИУ МГТУ им. Н.Э.Баумана |

|  |  |  |  |
| --- | --- | --- | --- |
| Студент ИУ6-4ХБ |  |  | И.И. Иванов |
|  |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |
| Руководитель практики |  |  | С.С. Семенов |
|  |  | (Подпись, дата) | (И.О. Фамилия) |

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*2021 г.*

**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Московский государственный технический университет имени Н.Э. Баумана**

**(национальный исследовательский университет)»**

**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой \_\_\_ИУ6\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ А.В. Пролетарский

« \_01\_\_ » \_\_07\_\_\_\_\_2021 г.

**ЗАДАНИЕ**

**на производственную практику**

по теме \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Студент группы ИУ6-4ХБ

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Иванов Иван Иванович\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Фамилия, имя, отчество)

Направление подготовки 09.03.01 Информатика и вычислительная техника

Бакалаврская программа 09.03.01\_03 Вычислительные машины, комплексы, системы и сети

Тип практики Технологическая практика

Название предприятия «название» **/** НУК ИУ МГТУ им. Н.Э. Баумана

***Техническое задание*** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Оформление отчета по практике:***

Отчет на 15-25 листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_перечень, если есть, или слово «нет» при отсутствии\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания « ХХ » ХХХХ 2021 г.

|  |  |  |  |
| --- | --- | --- | --- |
| **Руководитель практики** |  |  | С.С. Семенов |
|  |  | (Подпись, дата) | (И.О. Фамилия) |
| **Студент** |  |  | И.И. Иванов |
|  |  | (Подпись, дата) | (И.О. Фамилия) |

Примечание: Задание оформляется в двух экземплярах.

СОДЕРЖАНИЕ

[Введение 4](#_Toc78918933)

[Основная часть 5](#_Toc78918934)

[1.1 Udev-правила 5](#_Toc78918935)

# Введение

Цель производственной практики: изучение механизмов работы с внешнеми устройствами в ОС семейства Linux, библиотеки алгоритмов компьютерного зрения OpenCV, овладение навыками работы с сетями типа “интернет вещей”.

Задачи производственной практики:

* Получить знания о работе драйверов и udev правил в ОС семейства Linux
* Получить базовые навыки работы с docker контейнерами
* Получить базовые знания об алгоритмах компьютерного зрения
* Получить базовые навыки работы с сообщениями в сетях типа “интернет вещей”

# Основная часть

# Udev правила

Udev (userpace / dev) – подсистема Linux для динамического обнаружения и управления устройствами.

Udev запускается как демон и принимает события uevents от ядра, которые генерируются при инициализации или удалении устройства из системы. Задаваемые пользователем (системой) правила сверяются со свойствами события и соответствующего устройства, и совпавшее правило может назвать и создать соответствующий файл устройств, а также выполнить другие программы для инициализации и конфигурации устройства.

Задание: Написать udev правило, которое будет копировать на флеш-накопитель с заданным ID логи загрузки системы.

Udev правило, решающее задачу представлено в листинге 1

Листинг 1

ACTION=="add", ATTRS{serial}=="408D5CBEC94AB471998546ED", RUN+="/usr/bin/add\_trigger.sh"

Запускаемый правилом сценарий bash представлен в листинге 2

Листинг 2

#!/bin/bash

sudo mount /dev/sdf1 /media/usb

sudo cat /var/log/boot.log > /media/usb/logs/boot.log

sudo cat /var/log/dmesg > /media/usb/logs/dmesg.log

sudo umount /media/usb

Результаты работы программы (содержимое флеш-накопителя просматривалось с ОС Windows во момент составления отчета):

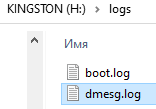


Рисунок 1 – просмотр файлов на флеш-накопителе

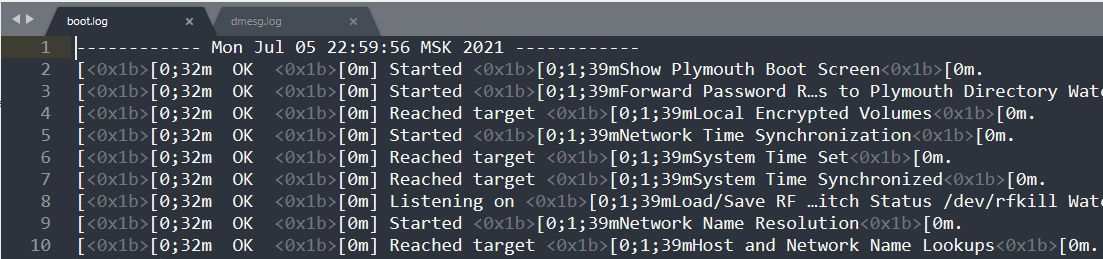


Рисунок 2 – фрагмент содержимого файла boot.log

# 1.2 Драйверы в Linux

Драйвер — программное обеспечение, с помощью которого другое программное обеспечение (операционная система) получает доступ к аппаратному обеспечению некоторого устройства.

Основные типы драйверов: символьные драйверы, блочные драйверы, сетевые драйверы.

Символьный драйвер обрабатывает поток байт.

Задание: написать драйвер, который инкрементирует внутреннюю переменную при каждом считывании.

Фрагмент исходного кода драйвера (на языке С), решающего поставленную задачу предствален в листинге 3

Листинг 3

#include <linux/kernel.h>

#include <linux/module.h>

// <…>подключение других библиотек

#define SUCCESS 0

#define DEVICE\_NAME "test"

static int device\_open( struct inode \*, struct file \* );

static int device\_release( struct inode \*, struct file \* );

static ssize\_t device\_read( struct file \*, char \*, size\_t, loff\_t \* );

static int major\_number;

static int is\_device\_open = 0;

static int counter = 0;

static struct file\_operations fops =

{

.read = device\_read,

.open = device\_open,

.release = device\_release

};

/\*

Реализация функций

static int \_\_init test\_init( void )

static void \_\_exit test\_exit( void )

static int device\_open( struct inode \*inode, struct file \*file )

static int device\_release( struct inode \*inode, struct file \*file )

\*/

static ssize\_t device\_read(struct file \*filp, char \*buffer, size\_t length,

loff\_t \*offset)

{

sprintf(buffer, "%d\n", counter++);

printk( "Read driver tick.\n" );

put\_user( counter , buffer);

return counter;

}

Для проверки работоспособности драйвера была написана программа, производящая циклическое чтение с заданного устройства. Ее исходный код представлен на листинге 4.

Листинг 4

#include <unistd.h>

#include <stdio.h>

#define O\_RDONLY 00

int main(){

int fd;

fd = open("/dev/test", O\_RDONLY);

while(1) {

sleep(1);

unsigned int t = 123;

unsigned int t2 = 456;

t2 = read(fd, &t, sizeof(t));

printf("%d\n", t2);

}

}

Результаты работы тестовой программы представлены на рисунке 3.

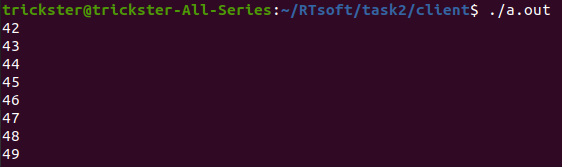


Рисунок 3 - результаты работы тестовой программы

# Контейнеризация

Контейнеризация (виртуализация на уровне операционной системы, контейнерная виртуализация, зонная виртуализация) — метод виртуализации, при котором ядро операционной системы поддерживает несколько изолированных экземпляров пространства пользователя вместо одного.

При этом при контейнеризации отсутствуют дополнительные ресурсные накладные расходы на эмуляцию виртуального оборудования и запуск полноценного экземпляра операционной системы, характерные при аппаратной виртуализации.

Docker — программное обеспечение для автоматизации развёртывания и управления приложениями в средах с поддержкой контейнеризации, контейнеризатор приложений.

Задание: создать docker-контейнер, содержащий браузер Mozilla firefox и запускающий его в графическом режиме.

Dockerfile, описывающий требуемый контейнер приведен в листинге 5.

Листинг 5

|  |
| --- |
| FROM centos:8 |
|  |

|  |
| --- |
| RUN yum install firefox -y |
|  |

|  |
| --- |
| RUN yum install -y libcanberra-gtk2 |
|  |

CMD firefox

Dockerfile устанавливает в контейнер минимальную версию Linux диструбитва Cent OS, затем устанавливает браузер, графические библиотеки, запускает браузер.

Перед запуском контейнера необходимо предоставить docker права доступа к оконной системе X Window System

$ sudo xhost +local:root

При запуске контейнера необходимо указать параметры сети и графического интерфейса

$ sudo docker run -it --env="DISPLAY" --net=host myfirefox

# Компьютерное зрение

Компьютерное зрение (иначе техническое зрение) — теория и технология создания машин, которые могут производить обнаружение, отслеживание и классификацию объектов.

OpenCV — библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом.

Оператор Кэнни (детектор границ Кэнни, алгоритм Кэнни) в дисциплине компьютерного зрения — оператор обнаружения границ изображения.

Задача: написать программу, производящую поиск и выделение границ контрастных объектов на видео.