|  |  |
| --- | --- |
| ÐÐ¾Ð³Ð¾ÑÐ¸Ð¿ ÐÐÐ | Федеральное государственное бюджетное образовательное учреждение высшего образования  **«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ**  **(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»** |

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра 307 «Цифровые технологии и информационные системы»

ОТЧЕТ

о выполнении лабораторной работы №2 по дисциплине

«Драйверы устройств»

по теме:

«Патч для сетевого драйвера BBB»

Выполнили студенты  
группы: М3О-421Б-21  
Шульц Богдан   
Сунгатуллин Саид  
Олейник София  
Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил преподаватель:  
Максимов Алексей Николаевич

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Цель лабораторной работы

Разработка патча для сетевого драйвера Beaglebone Black. Патч должен обеспечивать вывод следующей информации в log ядра:

* номер прерываний драйвера;
* адрес (физический) области ввода-вывода;
* размер области;
* содержимое и размер данных входящих и исходящих пакетов.

Добавить патч в buildroot и протестировать его.

Ход работы

На устройстве Beaglebone Black была выполнена команда ls -l /sys/class/net/eth0/device/driver. Результат её выполнения показан на рис. Рисунок 1.

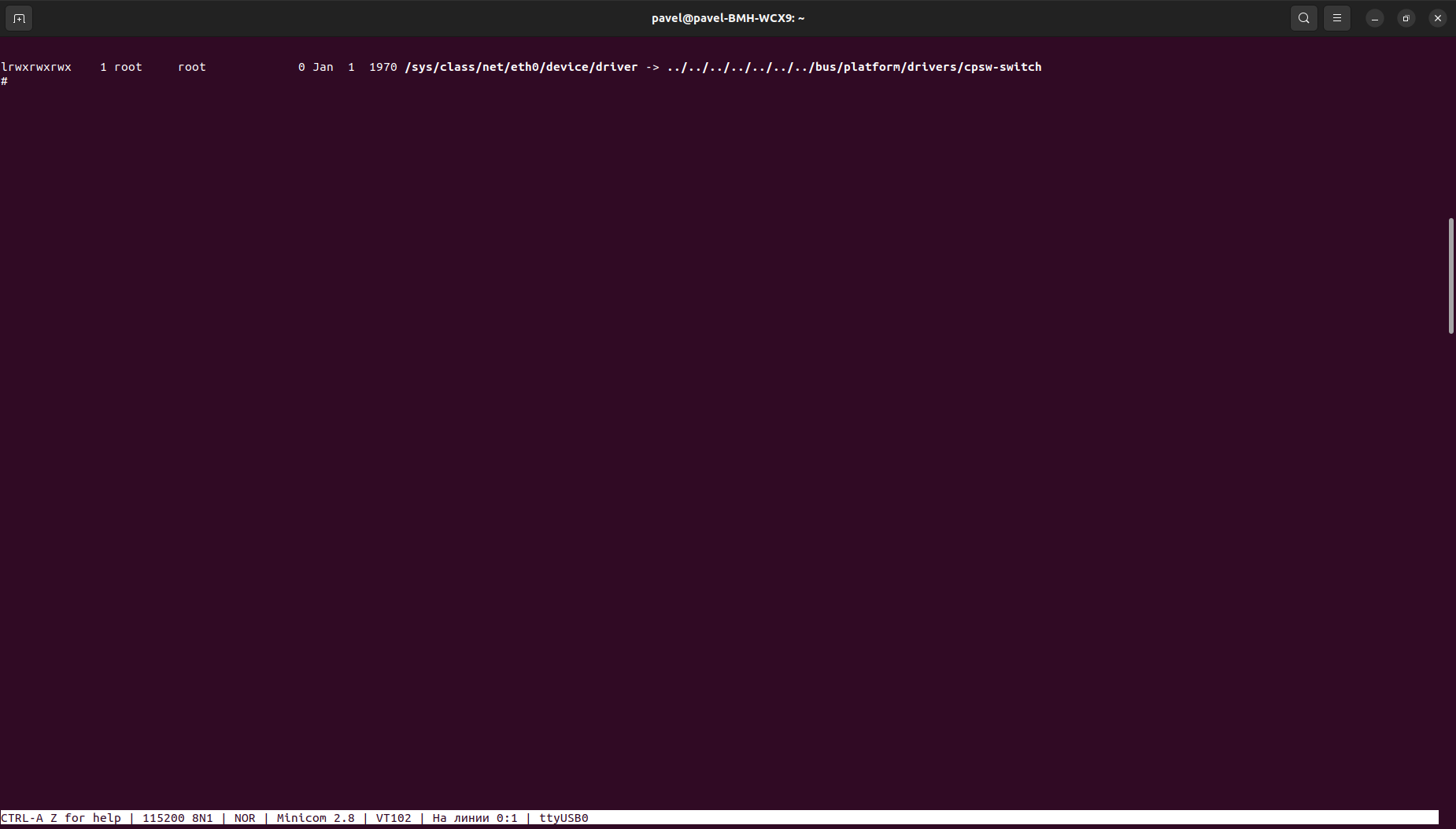


Рисунок 1 — Результат выполнения команды

Таким образом мы узнали, что за сеть отвечает драйвер cpsw-switch.

1. Модификация исходного кода ядра.

В исходном коде ядра Linux был найден файл cpsw\_new.c, являющийся частью драйвера Ethernet.

В исходный код файла cpsw\_new.c добавлены отладочные сообщения с использованием команды printk. Эти сообщения позволяют отслеживать работу драйвера в процессе загрузки и эксплуатации.

Измененная версия файла сохранена под именем cpsw\_new\_patched.c.

2. Создание патча.

С помощью команды diff (см. рис. 2) был создан файл патча cpsw\_new.patch, содержащий различия между оригинальным и модифицированным файлами.

diff -p cpsw\_new.c cpsw\_new\_patched.c > cpsw\_new.patch

При запуске beaglebone black log ядро выводит следующую информацию:

[ 0.504884] cpsw\_probe called

[ 0.505096] IRQ rx number: 29

[ 0.505167] IRQ tx number: 2a

[ 0.505225] IRQ misc number: 2b

[ 0.505334] IO region: 0, size: 4000

Эта информация указывает на номера прерываний для данного драйвера в шестнадцатеричной системе, а также информацию о физическом адресе ввода-вывода и размере этой области 4000.

3. Проверка информации с помощью конфигурации device tree.

Для получения читаемого файла в папке buildroot/output/images выполним следующие команду:

dtc aam335x-boneblack.dtb -o am335x-boneblack.dts

switch@0 {

compatible = "ti,am335x-cpsw-switch\0ti,cpsw-switch";

reg = <0x00 0x4000>;

ranges = <0x00 0x00 0x4000>;

clocks = <0x4f>;

clock-names = "fck";

#address-cells = <0x01>;

#size-cells = <0x01>;

syscon = <0x06>;

status = "okay";

interrupts = <0x28 0x29 0x2a 0x2b>;

interrupt-names = "rx\_thresh\0rx\0tx\0misc";

pinctrl-names = "default\0sleep";

pinctrl-0 = <0x52>;

pinctrl-1 = <0x53>;

Из этого фрагмента файла device tree видно, что номер прерывания, адрес и размер области совпадают с выведенными значениями.

4. Проверка логирования.

Для проверки логирования выполним следующие команды:

echo auto eth0 >> /etc/network/interfaces

echo iface eth0 inet dhcp >> /etc/network/interfaces

ifup eth0

Интерфейс начнёт логирование данных:

[ 1033.843836] Outgoing packet data: cbf11fa0, size: 70

[ 1034.078227] Incoming packet data: 43e59e1f, size: 46

[ 1034.083323] Incoming packet data: a811e238, size: 46

Вывод

В лабораторной работе №1 разработан патч для сетевого драйвера Beaglebone black, который осуществляет логирование данных, а именно содержимое и размер данных входящих и исходящих пакетов.

Приложение А. Код программы

Код программы написан на языке C.

\*\*\* /drivers/net/ethernet/ti/cpsw\_new.c

--- cpsw\_new\_patched.c

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* static void cpsw\_rx\_handler(void \*token,

\*\*\* 377,382 \*\*\*\*

--- 377,385 ----

/\* mark skb for recycling \*/

skb\_mark\_for\_recycle(skb);

netif\_receive\_skb(skb);

+

+ /\* Print incoming packet data \*/

+ printk(KERN\_INFO "Incoming packet data: %p, size: %d\n", skb->data, skb->len);

ndev->stats.rx\_bytes += len;

ndev->stats.rx\_packets++;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* static void cpsw\_slave\_open(struct cpsw\_

\*\*\* 788,793 \*\*\*\*

--- 791,797 ----

static int cpsw\_ndo\_stop(struct net\_device \*ndev)

{

+ printk(KERN\_INFO "cpsw\_ndo\_stop called");

struct cpsw\_priv \*priv = netdev\_priv(ndev);

struct cpsw\_common \*cpsw = priv->cpsw;

struct cpsw\_slave \*slave;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* static int cpsw\_ndo\_stop(struct net\_devi

\*\*\* 821,831 \*\*\*\*

--- 825,838 ----

cpsw->usage\_count--;

pm\_runtime\_put\_sync(cpsw->dev);

+ /\* Print IO region information \*/

+ printk(KERN\_INFO "IO region: %lx, size: %d\n", ndev->base\_addr, ndev->mem\_end - ndev->base\_addr);

return 0;

}

static int cpsw\_ndo\_open(struct net\_device \*ndev)

{

+ printk(KERN\_INFO "cpsw\_ndo\_open called");

struct cpsw\_priv \*priv = netdev\_priv(ndev);

struct cpsw\_common \*cpsw = priv->cpsw;

int ret;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* static netdev\_tx\_t cpsw\_ndo\_start\_xmit(s

\*\*\* 941,946 \*\*\*\*

--- 948,959 ----

skb\_tx\_timestamp(skb);

ret = cpdma\_chan\_submit(txch, skb, skb->data, skb->len,

priv->emac\_port);

+

+ /\* Print IO region information \*/

+ printk(KERN\_INFO "IO region: %lx, size: %d\n", ndev->base\_addr, ndev->mem\_end - ndev->base\_addr);

+

+ /\* Print outgoing packet data \*/

+ printk(KERN\_INFO "Outgoing packet data: %p, size: %d\n", skb->data, skb->len);

if (unlikely(ret != 0)) {

cpsw\_err(priv, tx\_err, "desc submit failed\n");

goto fail;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* static int cpsw\_probe(struct platform\_de

\*\*\* 1855,1860 \*\*\*\*

--- 1868,1875 ----

struct clk \*clk;

int irq;

+ printk(KERN\_INFO "cpsw\_probe called");

+

cpsw = devm\_kzalloc(dev, sizeof(struct cpsw\_common), GFP\_KERNEL);

if (!cpsw)

return -ENOMEM;

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* static int cpsw\_probe(struct platform\_de

\*\*\* 1896,1911 \*\*\*\*

--- 1911,1932 ----

if (irq < 0)

return irq;

cpsw->irqs\_table[0] = irq;

+

+ printk(KERN\_INFO "IRQ rx number: %d\n", irq);

irq = platform\_get\_irq\_byname(pdev, "tx");

if (irq < 0)

return irq;

cpsw->irqs\_table[1] = irq;

+ printk(KERN\_INFO "IRQ tx number: %d\n", irq);

+

irq = platform\_get\_irq\_byname(pdev, "misc");

if (irq <= 0)

return irq;

cpsw->misc\_irq = irq;

+

+ printk(KERN\_INFO "IRQ misc number: %d\n", irq);

platform\_set\_drvdata(pdev, cpsw);