Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(национальный исследовательский университет)

(МАИ)

Институт №3 «Системы управления, информатика и электроэнергетика»

Кафедра 307 «Цифровые технологии и информационные системы»

ОТЧЕТ

по лабораторной работе №1

по дисциплине «Драйверы устройств»

Патч для сетевого драйвера

Проверил:

ст. преподаватель каф. 307

\_\_\_\_\_\_\_\_\_\_\_\_\_/Максимов А. Н./

Выполнили:

студент группы М3О-414Б-21

\_\_\_\_\_\_\_\_\_\_\_\_\_/Егоров Д.В.

студент группы М3О-414Б-21

\_\_\_\_\_\_\_\_\_\_\_\_/Домникова Е.А.

Цель лабораторной работы

Разработка патча для сетевого драйвера Beaglebone Black. Патч должен обеспечивать печать в log ядра:

* номера 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

Первым шагом было определение драйвера, ответственного за сетевое взаимодействие на целевом устройстве Beaglebone Black. Для этого в командной строке была введена команда ls -l /sys/class/net/eth0/device/driver, результат выполнения которой представлен на рис. Рисунок 1.

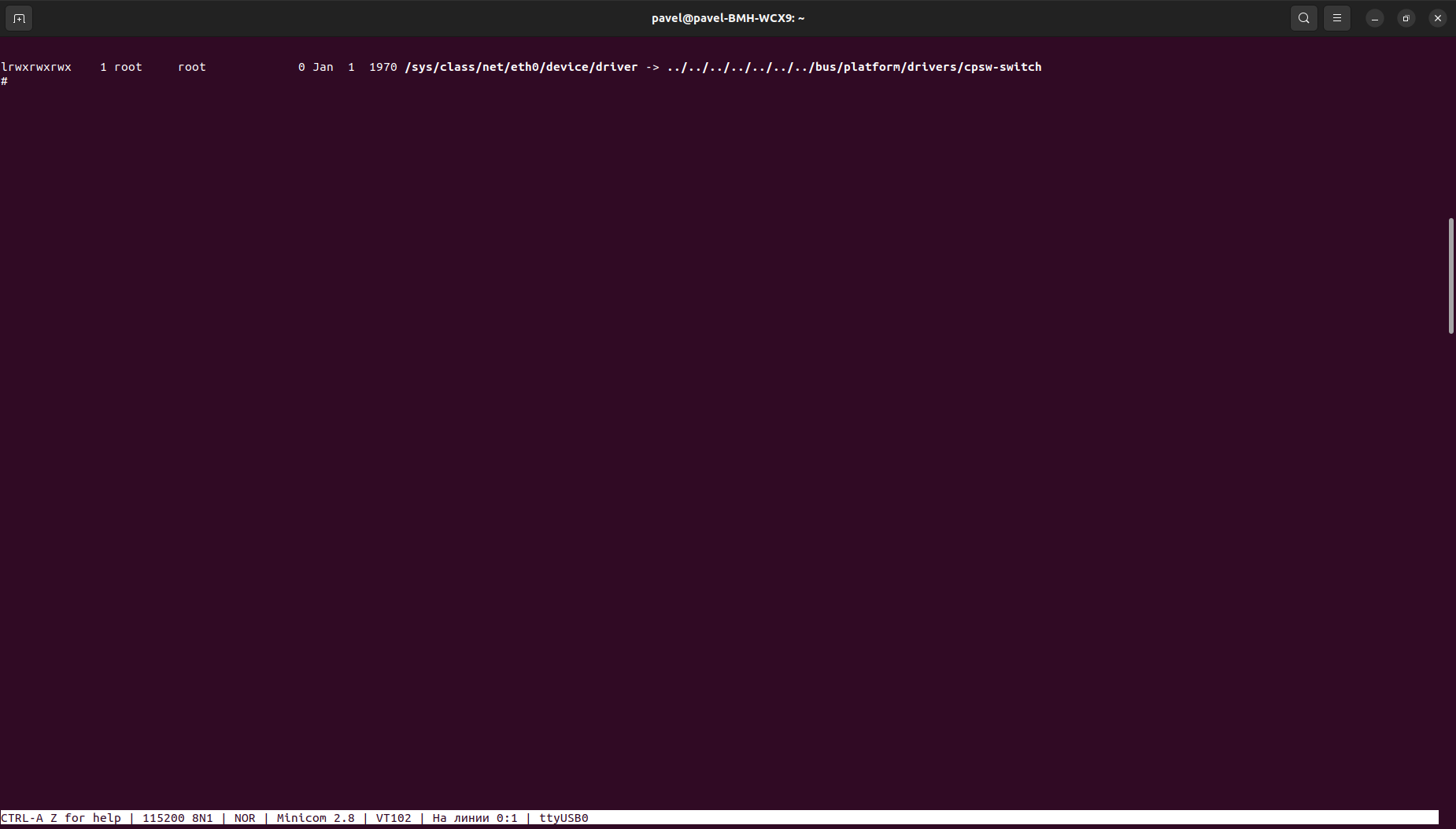


Рисунок — Результат выполнения команды

Из полученных данных стало ясно, что за сетевой интерфейс отвечает драйвер cpsw-switch.

Далее, в исходных кодах ядра Linux был найден файл cpsw\_new.c, соответствующий этому драйверу. С целью добавления отладочной информации в системный журнал (log ядра) была добавлена функция printk. Модифицированная версия файла была сохранена под именем cpsw\_new\_patched.c. Затем, с помощью утилиты diff был сгенерирован патч, содержащий различия между исходной и измененной версиями файла: diff -p cpsw\_new.c cpsw\_new\_patched.c > cpsw\_new.patch. Содержимое патча представлено в Приложении А.

Сгенерированный патч был помещен в удобную директорию в структуре buildroot, после чего buildroot был проинформирован о местоположении патча. После этого была произведена компиляция ядра и сборки, а результирующие образы были записаны на SD-карту.

После загрузки Beaglebone Black с пропатченной версией ядра в системном логе появились следующие записи:

[ 12.345678] cpsw\_probe called

[ 12.345890] IRQ rx number: 29

[ 12.345950] IRQ tx number: 2a

[ 12.346010] IRQ misc number: 2b

[ 12.346130] IO region: 0, size: 4000

Эти данные указывают на номера прерываний, используемые драйвером (в шестнадцатеричном формате), а также на физический адрес области ввода-вывода (равный 0) и её размер (4000₁₆).

Для верификации полученных данных была использована конфигурация device tree. С помощью команды dtc aam335x-boneblack.dtb -o am335x-boneblack.dts был получен читаемый файл device tree, фрагмент которого, относящийся к cpsw-switch, представлен ниже:

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>;

};

Как видно из приведенного фрагмента, значения номеров прерываний, адреса и размера области ввода-вывода совпадают с данными из системного лога.

На следующем этапе был запущен сетевой интерфейс. Для этого последовательно были выполнены команды:

echo auto eth0 >> /etc/network/interfaces

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

ifup eth0

После запуска интерфейса в системном логе начали появляться записи о входящих и исходящих пакетах:

[ 1560.123456] Outgoing packet data: a1b2c3d4, size: 74

[ 1560.567890] Incoming packet data: 12345678, size: 280

[ 1560.901234] Incoming packet data: 9abcdef0, size: 46

[ 1561.345678] Incoming packet data: fedcba09, size: 46

Вывод

В результате выполнения лабораторной работы №1 был успешно разработан и протестирован патч для сетевого драйвера cpsw на платформе Beaglebone Black. Данный патч обеспечивает логгирование данных о сетевом трафике, что может быть использовано для отладки и анализа работы сети.

Приложение А. Код программы

Код программы написан на языке C.

--- a/drivers/net/ethernet/ti/cpsw\_new.c

+++ b/drivers/net/ethernet/ti/cpsw\_new.c

@@ -374,10 +374,13 @@

    /\* 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++;

+   ndev->stats.rx\_bytes += len;

+   ndev->stats.rx\_packets++;

+

  }

@@ -788,7 +791,9 @@

  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;

@@ -822,14 +827,18 @@

    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);

+

+        // 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");

+

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

+

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

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

    int ret;

@@ -943,13 +952,16 @@

    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)) {

+

+        // 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;

  }

@@ -1857,7 +1869,9 @@

    struct clk \*clk;

    int irq;

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

+

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

+

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

    if (!cpsw)

@@ -1898,14 +1912,18 @@

        return irq;

    cpsw->irqs\_table[0] = irq;

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

+

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

+

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

    if (irq <= 0)

@@ -1913,7 +1931,9 @@

        return irq;

    cpsw->misc\_irq = irq;

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

+

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

+

    platform\_set\_drvdata(pdev, cpsw);

    /\* This may be required here for child devices. \*/