ЯДРО LINUX ДЛЯ 1892ВМ14Я. РУКОВОДСТВО ПРОГРАММИСТА

Версия v3.1

01.11.2019

ОГЛАВЛЕНИЕ

1. О ДОКУМЕНТЕ

Документ содержит описание основных подсистем и драйверов ядра Linux, реализованных для поддержки аппаратуры СнК 1892ВМ14Я и модулей на базе СнК.

Ядро Linux поддерживает модули следующих ревизий:

- Салют-ЭЛ24Д1 r1.3;
- Салют-ЭЛ24Д1 r1.4;
- Салют-ЭЛ24Д1 r1.5;
- Салют-ЭЛ24Д2 r1.1;
- Салют-ЭЛ24ОМ1 r1.1 с установленным Салют-ЭЛ24ПМ1 r1.1 или Салют-ЭЛ24ПМ1 r1.2;
- Салют-ЭЛ24ОМ1 r1.2 с установленным Салют-ЭЛ24ПМ1 r1.2, Салют-ЭЛ24ПМ2 r1.0 или Салют-ЭЛ24ПМ2 r1.1.

Файлы DTS `*.dtsi`, `*.dts` расположены в дереве исходных кодов U-Boot `arch/arm/dts/*.dts*`. Пути до прочих файлов приведены относительно корня дерева исходных кодов Linux.

2. ПОДСИСТЕМА УПРАВЛЕНИЯ ТАКТОВЫМИ СИГНАЛАМИ

Управление тактовыми сигналами и частотами в ядре Linux реализовано с использованием Common Clock Framework.[1] Тактовые сигналы микросхемы описаны в виде дерева в файле `mcom.dtsi`. Для управления тактовыми сигналами и частотами используются 4 драйвера, описанные в `drivers/clk/elvees/clk-mcom.c`:

- `mcom-clk-gate`;
- `mcom-clk-divider`;
- `mcom-clk-pll`;
- `mcom-cmctr`.

Для корректного управления тактовыми сигналами каждый драйвер устройства, входящий в состав ядра Linux, должен реализовывать:

1. При инициализации драйвера:
    1. Захват необходимого для устройства тактового сигнала, используя функцию `clk_get()`.
    2. Включение тактового сигнала, используя функцию `clk_enable()`.
2. При удалении драйвера:
    1. Выключение тактового сигнала, используя функцию `clk_disable()`.

При инициализации подсистемы управления тактовыми сигналами происходит начальная настройка всех PLL и делителей частот микросхемы. Устанавливаемые при инициализации значения множителей PLL и делителей частот описаны в файле `mcom.dtsi`.

[1] https://www.kernel.org/doc/Documentation/clk.txt

3. ДРАЙВЕР КОНТРОЛЛЕРА PWM *PWM-MCOM*

Драйвер *pwm-mcom* управляет контроллером PWM 1892ВМ14Я. Драйвер реализует стандартный интерфейс PWM[2]

Файл с исходным кодом драйвера — `drivers/pwm/pwm-mcom.c`. Описание DTS bindings представлено в файле `Documentation/devicetree/bindings/pwm/elvees,mcom-pwm.txt`.

Ограничения драйвера:

1. Не реализовано управление каналами OUTB.
2. Не поддерживается режим счёта PWM-контроллера (PWM API не поддерживает данный режим).
3. Не реализовано управление предделителем.

[2] https://www.kernel.org/doc/Documentation/pwm.txt

4. ДРАЙВЕР КОНТРОЛЛЕРА ДИСПЛЕЯ *VPOUT-DRM*

Данный раздел применим к драйверу контроллера дисплея VPOUT СнК 1892ВМ14Я для подсистемы DRM — *vpout-drm*.

Документация, описывающая текущую версию подсистемы DRM, доступна по ссылке Linux GPU Driver Developer's Guide[3].

Исходный код драйвера содержится в директории `drivers/gpu/drm/vpout`.

Драйвер обеспечивает следующие возможности:

1. Разрешение экрана до 1920x1080 пикселей;
2. Поддержка внешнего HDMI передатчика NXP TDA998x;
3. Поддержка внешних панелей с заданием параметров дисплея через DTS;
4. Чтение Extended Display Identification Data (EDID);
5. Эмуляция фреймбуфера через устройство `/dev/fb0`.

Ограничения драйвера:

1. Не поддерживаются чересстрочные видеорежимы (не поддерживаются контроллером дисплея VPOUT);
2. Не поддерживаются HDMI передатчики отличные от NXP TDA998x;
3. Не реализована поддержка абстракции плоскостей (plane abstraction);
4. Не поддерживается атомарная установка видеорежима.

При использовании в качестве устройства вывода HDMI монитора драйвер устанавливает оптимальный для подключенного монитора видеорежим, определяемый по EDID. С помощью параметров ядра (kernel parameters) возможно установить фиксированный видеорежим. Например, следующая строка задает разрешение экрана в 1280×720 пикселей:

```
video=HDMI-A-1:1280x720
```

Подробное описание параметров ядра, задающих видеорежим, содержится в документе `Documentation/fb/modedb.txt`.

[3] https://www.kernel.org/doc/html/latest/gpu/index.html

5. ДРАЙВЕР ФРЕЙМБУФЕРА *VPOUTFB*

Для вывода на экран графического окружения на СнК используется подсистема FBDev[4] и драйвер *vpoutfb*. Директория с исходным кодом драйвера — `drivers/video/fbdev/vpoutfb`. Драйвер управляет контроллером VPOUT и HDMI-адаптером ITE IT66121.

Алгоритм работы драйвера:

1. Если в DTS в узле output присутствует свойство compatible=”ite,it66121”, то выполнить настройку контроллера ITE IT66121, подключенного по I2C.
2. Считать из DTS видеорежим и настроить VPOUT для вывода в заданном видеорежиме.
3. Если в DTS отсутствует видеорежим или тайминги некорректны, или свойство output отсутствует, то настроить VPOUT для вывода в режиме 720p 60 FPS.

Вызов ioctl `FBIOPUT_VSCREENINFO` с заданием неподдерживаемого режима завершается с `-EINVAL`. (Следовательно, вызов `fbset` завершится с ненулевым кодом возврата).

Поддерживаются следующие ioctl:

- `FBIOGET_VSCREENINFO`;
- `FBIOPUT_VSCREENINFO`;
- `FBIOGET_FSCREENINFO`;
- `FBIOGETCMAP`;
- `FBIOPUTCMAP`;
- `FBIOBLANK`;
- `VPOUTFB_GET_MEMORY_ID`.

При появлении прерывания OUT_FIFO_INT блока VPOUT драйвер останавливает и переинициализирует VPOUT. При этом в dmesg печатается сообщение “Caught OUT_FIFO_INT, reinitializing VPOUT”.

В драйвере не реализовано:

1. Чтение EDID HDMI-монитора и ограничение возможных разрешений согласно данным из EDID.
2. Остановка/запуск VPOUT при отключении/подключении HDMI-монитора.

Примечание: Т.к. автоматическое определение подключения HDMI-монитора отсутствует, драйвер может быть выключен по умолчанию. Загрузка драйвера выполняется командой `modprobe vpoutfb`.

---

[4] https://www.kernel.org/doc/Documentation/fb/api.txt

Примечание: Для управления видеорежимами может использоваться утилита `fbset` и файл `fb.modes`.

Драйвер считывает видеорежим из DTS в соответствии с описанием в `Documentation/devicetree/bindings/video/display-timing.txt`. В DTS-файле `mcom.dtsi` описан формат цветовых компонентов изображения, устанавливаемые при инициализации драйвера. Подробное описание полей узла устройства VPOUT представлено в файле `Documentation/devicetree/bindings/fb/vpoutfb.txt`.

Примечание: Модуль *vpoutfb* используется консолью – перед выгрузкой модуля необходимо отключить консоль от драйвера:

```
echo 0 > /sys/class/vtconsole/vtcon1/bind
modprobe -r vpoutfb
```

6. ДРАЙВЕР VPU *AVICO*

Драйвер *avico* управляет VPU VELcore-01 и реализует аппаратное сжатие видео по стандарту H.264. Драйвер реализован с использованием подсистемы V4L2[5] и предоставляет стандартный программный интерфейс для сжатия и управления.

Возможности драйвера:

1. Поддерживаются входные кадры в формате M420[6].
2. Максимальная ширина кадра — 1920 пикселей.
3. Максимальная высота кадра — 4096 пикселей.
4. Поддержка ширины и высоты кадра, кратных 2.
5. Возможность установки FPS видеопотока.
6. Возможность установки параметра QP с помощью контролов[7]:
   - V4L2_CID_MPEG_VIDEO_H264_I_FRAME_QP;
   - V4L2_CID_MPEG_VIDEO_H264_P_FRAME_QP;
   - V4L2_CID_MPEG_VIDEO_H264_CHROMA_QP_INDEX_OFFSET.
7. Возможность установки IDR-кадра с помощью контрола V4L2_CID_MPEG_VIDEO_FORCE_KEY_FRAME.
8. Возможность установки размера GOP с помощью контрола V4L2_CID_MPEG_VIDEO_GOP_SIZE. Новый размер GOP применяется со следующего IDR-кадра после завершения текущего GOP. Чтобы применить новый размер GOP на следующем кадре, нужно запросить IDR-кадр с помощью контрола V4L2_CID_MPEG_VIDEO_FORCE_KEY_FRAME.
9. Поддержка более одного потока кодирования видео. Максимальное число поддерживаемых потоков кодирования зависит от ширины и высоты кадра, количества запрошенных V4L2-буферов и объема памяти, которая может быть выделена с помощью Contiguous Memory Allocator (CMA). Теоретическое максимальное число поддерживаемых потоков кодирования можно вычислить с помощью формулы:

$$n = M/(W * (128 + (2 * B_c + 3/2 * (B_o + 1)) * H))$$

   где *M* — размер памяти, которая может быть выделена с помощью CMA, *W* — ширина кадра, *H* — высота кадра, B_o — количество V4L2-буферов output-интерфейса[8], B_c — количество V4L2-буферов capture-интерфейса[9]. Дополнительно количество потоков *n* ограничивается фрагментацией CMA. Производительность кодирования

[5] https://linuxtv.org/downloads/v4l-dvb-apis/uapi/v4l/v4l2.html
[6] https://linuxtv.org/downloads/v4l-dvb-apis/uapi/v4l/pixfmt-m420.html
[7] https://linuxtv.org/downloads/v4l-dvb-apis/uapi/v4l/ext-ctrls-codec.html
[8] https://linuxtv.org/downloads/v4l-dvb-apis/uapi/v4l/dev-output.html
[9] https://linuxtv.org/downloads/v4l-dvb-apis/uapi/v4l/dev-capture.html

каждого видео понижается с увеличением числа потоков, т.к. используется один аппаратный поток кодирования. Пример реального максимального числа потоков и достигаемой при этом производительности кодирования при QP 23, четырех V4L2-буферов output-интерфейса, четырех V4L2-буферов capture-интерфейса, размере CMA-памяти 128 МБ и частоте VPU 312 МГц:

- 1920x1072 — 4 потока, ~15 FPS;
- 1280x720 — 6 потоков, ~22 FPS;
- 640x480 — 19 потоков, ~18 FPS.

Пример ограничения числа потоков кодирования для обеспечения производительности кодирования ~30 FPS при QP 23, четырех V4L2-буферов output-интерфейса, четырех V4L2-буферов capture-интерфейса, размере CMA-памяти 128 МБ и частоте VPU 312 МГц:

- 1920x1072 — 2 потока, ~30 FPS;
- 1280x720 — 4 потока, ~30 FPS;
- 640x480 — 12 потоков, ~30 FPS.

Ограничения драйвера:

1. Поддерживается только сжатие видео.
2. Поддерживается только один аппаратный поток кодирования.
3. Требуется нестандартный формат пикселей на входе (M420).
4. Шаг между яркостными и/или цветовыми строками должен быть кратен 16 байтам.
5. Требуется 180 КиБ памяти XYRAM.
6. Использование с драйвером *delcore30m* невозможно, т.к. драйвер *avico* использует SDMA через API DMA-engine, драйвер *delcore30m* — непосредственное управление SDMA.
7. Нет возможности сжатия с постоянным битрейтом.
8. Нет возможности менять FPS в процессе кодирования.

Для обхода проблемы rf#1382 драйвер использует промежуточные буферы в XYRAM для восстановленных и сжатых данных. Всего используется 4 буфера по 45 КиБ (строка макроблоков для кадра шириной 1920 пикселей в формате M420) — 2 буфера для восстановленных данных и 2 для сжатых. В результате реализации обхода проблемы, максимальная ширина кадров ограничилась 1920 пикселями.

После каждой строки макроблоков VPU останавливается и драйвер выполняет следующие действия:

1. Настраивает VPU на другой промежуточный буфер.
2. Запускает SDMA для копирования данных из промежуточного буфера в DDR.
3. Запускает VPU на обработку следующей строки макроблоков.

Для обхода проблемы rf#2003 в обработчике прерывания используется задержка, состоящая из следующих действий:

1. Ожидание завершения чтения очередных данных исходного и референсного кадров.
2. Ожидание завершения 80-кратного чтения регистра EVENTS.
3. Ожидание снятия всех флагов регистра EVENTS, указывающих на текущую работу VDMA.

7. ДРАЙВЕР КОНТРОЛЛЕРА ETHERNET *ARASAN-GEMAC*

Драйвер *arasan-gemac* управляет контроллером Ethernet Arasan GEMAC. Драйвер реализует стандартный интерфейс network devices, описанный в `Documentation/networking/netdevices.txt`. Обработка RX-прерываний реализована с использованием интерфейса NAPI[10].

Директория с исходным кодом драйвера — `drivers/net/ethernet/arasan`.

Драйвер поддерживает выполнение следующих операций из пространства пользователя:

1. Установка скорости (10/100/1000 МБит/с);
2. Установка дуплекса (full/half);
3. Установка уровня сообщений драйвера;
4. Установка MAC-адреса;
5. Перезапуск автосогласования;
6. Проверка физического подключения;
7. Установка MTU кадра в диапазоне 68 – 3500 байт.

Драйвер не поддерживает:

1. Управление паузой;
2. Чтение и запись EEPROM;
3. Wake-on-Lan;
4. Управление объединением прерываний.

[10] https://wiki.linuxfoundation.org/networking/napi

8. ПОДСИСТЕМА УПРАВЛЕНИЯ ЭНЕРГОПОТРЕБЛЕНИЕМ

Подсистема управления энергопотреблением Linux определяет модели управления энергопотреблением (подробнее см. Device Power Management Basics[11]):

- System Sleep;
- Runtime Power Management.

8.1 Модель System Sleep

В модели System Sleep определены состояния сна (подробнее см. System Power Management Sleep States[12]):

- Suspend-To-Idle (s2idle, freeze);
- Standby, Power-On Suspend (shallow, standby);
- Suspend-to-RAM (deep);
- Suspend-to-disk (disk).

Поддерживаемые состояния сна:

- Suspend-To-Idle;
- Power-On Suspend.

Для энергосбережения в состоянии Power-On Suspend используются свойства драйверов:

- поддержка приостановки (suspend) контроллера СнК в драйвере;
- поддержка приостановки (suspend) контроллера внешнего интерфейса (приёмопередатчик CAN, Ethernet PHY, и т.д.) в драйвере;
- поддержка CPU Hotplug (подробнее см. главу *CPU Hotplug*).

Поддержка приостановки реализована в драйверах контроллеров СнК:

- *avico* (невозможен переход в Power-On Suspend во время сжатия);
- *delcore-30m*;
- *designware-i2c*;
- *designware-i2s*;
- *dw-apb-gpio*;
- *dw-apb-uart*;

[11] https://www.kernel.org/doc/html/latest/driver-api/pm/devices.html
[12] https://www.kernel.org/doc/Documentation/power/states.txt

- *dw-wdt*;
- *dwc2*;
- *sdhci-mcom02*.

Поддержка приостановки реализована в драйверах контроллеров внешних интерфейсов модулей на базе СнК:

- *bcm4329-fmac*;
- *mcp2515*;
- *ft313h*.

Поддержка приостановки не реализована в драйверах контроллеров СнК:

- *arasan-gemac*;
- *dw-apb-ssi*;
- *dw-apb-timer*;
- *mcom-pwm*;
- *mfbsp-i2s*;
- *nfc-v2p99*;
- *pl330*;
- *vinc*;
- *vpout-drm*;
- *vpoutfb*.

Поддержка пробуждения (wakeup) реализована в драйверах:

- *dw-apb-uart* (подробнее см. *Пример пробуждения по событию от UART*);
- *dw-apb-gpio*;
- *rtc-ds1307* (подробнее см. *Пример пробуждения по событию от RTC*).

8.1.1 Пример пробуждения по событию от UART

1. Установить UART0 в качестве источника пробуждения:

```
echo enabled > /sys/devices/platform/38028000.serial/tty/ttyS0/power/wakeup
```

2. Перевести ОС в состояние сна:

```
echo freeze > /sys/power/state # enter Suspend-To-Idle state
```

или:

```
echo standby > /sys/power/state # enter Power-On Suspend state
```

3. В терминале на ПЭВМ отправить любой символ в приёмник контроллера UART0.

8.1.2 Пример пробуждения по событию от RTC

1. Перевести ОС в состояние сна до указанного времени пробуждения:

```
rtcwake -s 3 -m freeze # enter Suspend-To-Idle state
```

или:

```
rtcwake -s 3 -m standby # enter Power-On Suspend state
```

8.2 Модель Runtime Power Management

Для поддержки динамического управления энергопотреблением реализованы:

- механизм CPU hotplug;
- драйвер управления частотой ядер CPU *cpufreq-dt*.

8.2.1 Механизм CPU hotplug

Механизм CPU hotplug[13] позволяет включать и выключать процессорные ядра, не перезагружая систему, что может использоваться:

- для отключения CPU1;
- для перехода системы в состояния сна.

Для выключения и включения процессорных ядер используются функции `cpu_down()` и `cpu_up()`, описанные в файле `kernel/cpu.c`.

Использование через sysfs:

1. Для отключения питания ядра CPU1 необходимо выполнить:

```
echo 0 > /sys/devices/system/cpu/cpu1/online
```

2. Для включения питания ядра CPU1 необходимо выполнить:

```
echo 1 > /sys/devices/system/cpu/cpu1/online
```

8.2.2 Драйвер управления частотой ядер CPU *cpufreq-dt*

Штатный драйвер *cpufreq-dt*, позволяет управлять тактовой частотой ядер CPU0 и CPU1 через подсистему CPUfreq[14].

Директория с исходным кодом драйвера — `drivers/cpufreq`. Список частот ядер CPU описан в DTS-файле `mcom.dtsi`. Описание DTS bindings представлено в файле `Documentation/devicetree/bindings/cpufreq/cpufreq-dt.txt`.

Возможности драйвера:

[13] https://www.kernel.org/doc/Documentation/core-api/cpu_hotplug.rst
[14] https://www.kernel.org/doc/Documentation/cpu-freq/user-guide.txt

1. Регуляторы масштабирования тактовой частоты ядер CPU (CPUfreq governors):
    - `ondemand` (по-умолчанию) — устанавливает тактовую частоту в зависимости от нагрузки на ядрах CPU;
    - `conservative` — похож на `ondemand`, но более экономный (предпочтение отдаётся меньшим тактовым частотам);
    - `performance` — устанавливает тактовую частоту в максимальное значение;
    - `userspace` — позволяет устанавливать частоту из пространства пользователя.
2. Управление регуляторами и частотами через `sysfs`.

Ограничения драйвера:

1. Не поддерживается управление напряжением питания ядер CPU, т.к отсутствует поддержка в СнК.
2. Не поддерживается независимое управление частотой ядер CPU, т.к отсутствует поддержка в СнК.

Для установки тактовой частоты ядер из пространства пользователя необходимо:

1. Выбрать регулятор `userspace`:

```
echo userspace > /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor
```

2. Выбрать частоту из поддерживаемых. Список частот доступен в файле `/sys/devices/system/cpu/cpu0/cpufreq/scaling_available_frequencies`.
3. Установить частоту. Значение частоты передаётся в кГц, например:

```
echo 312000 > /sys/devices/system/cpu/cpu0/cpufreq/scaling_setspeed
```

9. ПОДСИСТЕМА UART В РЕЖИМЕ RS-485

Для управления полудуплексными приёмопередатчиками RS-485 используются ioctl TIOCMBIS/TIOCMBIC:

```
int rts_flag = TIOCM_RTS;
ioctl(fd, TIOCMBIS, &rts_flag); // set send mode
ioctl(fd, TIOCMBIC, &rts_flag); // set receive mode
```

Примечание: модули Салют-ЭЛ24ОМ1 имеют полудуплексный приёмопередатчик RS-485.