Драйверы флеш устройств

В этой секции описывается, как создать флеш драйвер и зарегистрировать флеш устройство.

Создание флеш драйвера

Флеш драйвер должен представлять собой элемент структуры alt_flash_dev, определённой в **sys/alt_flash_dev.h**. Следующий код представляет эту структуру:

```
struct alt_flash_dev
{
  alt_llist                 llist; // internal use only
  const char*               name;
  alt_flash_open            open;
  alt_flash_close           close;
  alt_flash_write           write;
  alt_flash_read            read;
  alt_flash_get_flash_info  get_info;
  alt_flash_erase_block     erase_block;
  alt_flash_write_block     write_block;
  void*                     base_addr;
  int                       length;
  int                       number_of_regions;
  flash_region    region_info[ALT_MAX_NUMBER_OF_FLASH_REGIONS];
};
```

Первый параметр llist для внутреннего использования, ему всегда должно быть установлено значение ALT_LLIST_ENTRY. Параметр name – это место устройства в файловой системе HAL, это имя устройства, определённое в **system.h**.

Семь полей с open до write_block являются указателями функций, которые реализуют функциональность в виде вызовов API приложений следующих функций:

- alt_flash_open_dev()
- alt_flash_close_dev()
- alt_write_flash()
- alt_read_flash()
- alt_get_flash_info()
- alt_erase_flash_block()
- alt_write_flash_block()

где:

- параметр base_addr – это базовый адрес флеш памяти
- length – это размер флеш памяти в байтах
- number_of_regions – это количество стираемых регионов во флеш памяти
- region_info – содержит информацию о месте и размере блоков в устройстве флеш памяти.

За дополнительной информацией о формате структуры flash_region обратитесь к секции "Использование флеш устройств" в главе "Разработка программ с использованием слоя аппаратной абстракции" и в настольной книге программиста под Nios II.

Некоторые флеш устройства, такие флеш с общим интерфейсом (CFI) – совместимые устройства, позволяющие вам считывать количество регионов (number_of_regions) и их конфигурацию (region_info) на стадии прогона. Для всех остальных флеш устройств, эти два поля должны быть заданы на стадии компиляции.

Регистрирование флеш устройств

После создания элемента структуры alt_flash_dev, вы должны сделать устройство доступным в системе HAL, вызвав следующую функцию:

```
int alt_flash_device_register( alt_flash_fd* fd)
```

Эта функция имеет один входной аргумент, который является регистрируемой структурой устройства. При успешной регистрации, возвращаемое значение – нуль. Отрицательное возвращаемое значение означает, что устройство не зарегистрировано.

Драйверы устройств DMA

HAL модели DMA транзакций, как принято, контролируются двумя оконечными устройствами: каналом приёма и каналом передачи. В этой секции отдельно описываются драйверы для каждого типа DMA канала.

За дополнительной информацией о формате структуры flash_region обратитесь к секции "Использование DMA устройств" в главе "Разработка программ с использованием слоя аппаратной абстракции" и в настольной книге программиста под Nios II.

Интерфейс драйвера устройства DMA задан в файле **sys/alt_dma_dev.h**.

Канал передачи DMA

Канал передачи DMA создаётся в виде элемента структуры alt_dma_txchan, см. пример 7-2.

Example 7–2. alt_dma_txchan Structure

```
typedef struct alt_dma_txchan_dev_s alt_dma_txchan_dev;
struct alt_dma_txchan_dev_s
{
  alt_llist   llist;
  const char* name;
  int         (*space) (alt_dma_txchan  dma);
  int         (*send) (alt_dma_txchan   dma,
                       const void*      from,
                       alt_u32          len,
                       alt_txchan_done* done,
                       void*            handle);
  int         (*ioctl) (alt_dma_txchan dma, int req, void* arg);
};
```

В табл. 7-2 показаны доступные поля и их функции.

Необходимо определить обе функции space и send. Если поле ioctl установлено в нуль, вызов alt_dma_txchan_ioctl() возвращает -ENOTTY для этого устройства.

После создания элемента структуры alt_dma_txchan, вы должны зарегистрировать устройства в системе HAL, чтобы сделать его доступным, вызвав следующую функцию:

int alt_dma_txchan_reg (alt_dma_txchan_dev* dev)

Входной аргумент dev служит для регистрации устройства. При успешной регистрации, возвращаемое значение – нуль. Отрицательное возвращаемое значение означает, что устройство не зарегистрировано.

Табл. 7-2. Поля структуры alt_dma_txchan

Поле	**Функция**
llist	Поле llist для внутреннего использования, ему всегда должно быть установлено значение ALT_LLIST_ENTRY.
name	Имя, связанное с этим каналом, при вызове alt_dma_txchan_open(). Имя устройства, определённое в файле **system.h**.
space	Указатель на функцию, которая возвращает количество дополнительных запросов передачи, находящихся в очереди запросов к устройству. Входной аргумент – это указатель на структуру alt_dma_txchan_dev.
send	Указатель на функцию, которая вызывается как результат вызова функции API приложения alt_dma_txchan_send(). Эта функция посылает запрос передачи в DMA устройство. Параметры, относящиеся к alt_txchan_send(), относятся непосредственно к send(). За описанием параметров и возвращаемых значений обратитесь к главе "Справка по HAL API" в настольной книге программиста под Nios II.
ioctl	Эта функция предлагает специальный I/O контроль. Обратитесь к файлу **sys/alt_dma_dev.h** за списком групповых опций, которые вы хотите, чтобы поддерживало ваше устройство.

Канал приёма DMA

Канал приёма DMA создаётся в виде элемента структуры alt_dma_rxchan, см. пример 7-3.

Example 7–3. alt_dma_rxchan Structure

```
typedef alt_dma_rxchan_dev_s alt_dma_rxchan;
struct alt_dma_rxchan_dev_s
{
  alt_llist   list;
  const char* name;
  alt_u32     depth;
  int         (*prepare) (alt_dma_rxchan   dma,
                          void*            data,
                          alt_u32          len,
                          alt_rxchan_done* done,
                          void*            handle);
  int         (*ioctl) (alt_dma_rxchan dma, int req, void* arg);
};
```

В табл. 7-3 показаны доступные поля и их функции.

Необходимо определить функцию prepare(). Если поле ioctl установлено в нуль, вызов alt_dma_rxchan_ioctl() возвращает -ENOTTY для этого устройства.

После создания элемента структуры alt_dma_rxchan, вы должны зарегистрировать устройства в системе HAL, чтобы сделать его доступным, вызвав следующую функцию:

int alt_dma_rxchan_reg (alt_dma_rxchan_dev* dev)

Входной аргумент dev служит для регистрации устройства. При успешной регистрации, возвращаемое значение – нуль. Отрицательное возвращаемое значение означает, что устройство не зарегистрировано.

Табл. 7-3. Поля структуры alt_dma_rxchan

Поле	**Функция**
llist	Поле llist для внутреннего использования, ему всегда должно быть установлено значение ALT_LLIST_ENTRY.
name	Имя, связанное с этим каналом, при вызове alt_dma_rxchan_open(). Имя устройства, определённое в файле **system.h**.
depth	Общее количество запросов на приём, которые ожидают выполнения в любое время.
prepare	Указатель на функцию, которая вызывается как результат вызова функции API приложения alt_dma_rxchan_prepare(). Эта функция посылает запрос приёма от DMA устройства. Параметры, относящиеся к alt_dma_rxchan_prepare(), относятся непосредственно к prepare(). За описанием параметров и возвращаемых значений обратитесь к главе "Справка по HAL API" в настольной книге программиста под Nios II.
ioctl	Эта функция предлагает специальный I/O контроль. Обратитесь к файлу **sys/alt_dma_dev.h** за списком групповых опций, которые вы хотите, чтобы поддерживало ваше устройство.