Руководство (v3.1)

По работе с драйвером модулей “mPCIe – CAN”, “PCIe – CAN” “mPCIe – TTCAN”, “PCIe – TTCAN”

Интерфейс ISO-11898 CAN Bus
(ISO-11898-4 для TT CAN)

Для драйверов версии 3.x

ОС WINDOWS

26.01.2021

ООО “НОВОМАР”

Оглавление

1. Введение 4
2. Установка драйвера 4
3. Взаимодействие с драйвером 4
3.1 Режим FIFO 5
4. Список доступных запросов ioctl 5
4.1.1 IOCTL_READ_BAR 7
4.1.2 IOCTL_WRITE_BAR 8
4.1.3 IOCTL_READ_CAN 9
4.1.4 IOCTL_WRITE_CAN 10
4.1.5 IOCTL_MODIFY_CAN 11
4.1.6 IOCTL_DEVICE_VERSION 12
4.1.7 IOCTL_DRIVER_VERSION 13
4.2.1 IOCTL_READ_DMA_BUF 14
4.3.1 IOCTL_WRITE_TXBUF 15
4.3.2 IOCTL_WRITE_TXBUF_AND_SEND 16
4.3.3 IOCTL_SEND_FROM_TXBUF 16
4.3.4 IOCTL_CHECK_TX 18
4.3.5 IOCTL_WAIT_FOR_TX 19
4.3.6 IOCTL_REMOVE_TXREQ 20
4.3.7 IOCTL_ABAT 21
4.3.8 IOCTL_SEND_BY_TRIGGER 22
4.3.9 IOCTL_SEND_BY_TRIGGER_LOOP 22
4.3.10 IOCTL_CHECK_TRIGGER 24
4.4.1 IOCTL_RESET_TTCAN 25
4.4.2 IOCTL_RESET_CANn_TTCAN 26
4.4.3 IOCTL_SETSPEED 27
4.4.4 IOCTL_GETSPEED 28
4.4.5 IOCTL_SETSPEED_PARAMS 29
4.4.6 IOCTL_SETMODE 30
4.4.7 IOCTL_GETMODE 31
4.4.8 IOCTL_DMA_ENABLE 32
4.4.9 IOCTL_DMA_DISABLE 33
4.4.10 IOCTL_SET_ONESHOT 34

4.4.11 IOCTL_GET_CAN_ERRORS 35
4.4.12 IOCTL_SET_CAN_MASKS 36
4.4.13 IOCTL_SET_CAN_TIMER_TRSH 37
4.4.14 IOCTL_SET_CAN_TIMER_CEED 38
4.4.15 IOCTL_SET_CAN_TIMER_FREE 39
4.4.16 IOCTL_SET_CAN_TIMER_RST_RXB 40
4.4.17 IOCTL_STOP_CAN_TIMER 41
4.4.18 IOCTL_GET_CAN_TIMER 42
4.4.19 IOCTL_START_TIMER_INT 43
4.4.20 IOCTL_STOP_TIMER_INT 44
4.4.21 IOCTL_WAIT_TIMER_INT 45
4.4.22 IOCTL_SET_CAN_TIMEOUT 46
4.5 Работа в режиме FIFO 47
4.5.1 IOCTL_SET_DEVICEMODE 47
4.5.2 IOCTL_WRITE_DATA_FIFO 48
4.5.3 IOCTL_GET_FIFOINFO 49
4.5.4 IOCTL_SET_TXPAUSE 50
4.5.5 IOCTL_WRITE_TG_FIFO 51
5.0 Обновление руководства 52

1. Введение

Универсальный драйвер версии 3.x предназначен для работы с модулями: xPCIe-CAN (PCIe-CAN и mPCIe-CAN) и xPCIe-TTCAN (PCIe-TTCAN и mPCIe-TTCAN) в ОС WINDOWS версий 7 /10.

Поддерживаются произвольное количество модулей, произвольное количество каналов (1 или 2), режим DMA, параллельные запросы.

Взаимодействие с драйвером осуществляется с помощью запросов ioctl.

Универсальный драйвер по запросам ioctl совместим с драйверами xPCIe-CAN и xPCIe-TTCAN версий 2.x.

2. Установка драйвера

Драйвер можно установить вручную через диспетчер оборудования Windows.

3. Взаимодействие с драйвером

После установки драйвер создаёт файлы *\Device\nmCANx* и *\DosDevices\nmCANx* , или *\Device\nmTTCANx* и *\DosDevices\nmTTCANx* для каждого обнаруженного в системе устройства xPCIe-CAN или xPCIe-TTCAN. Каждое устройство реализует интерфейс CAN/TTCAN с уникальным идентификатором {3bd2b180-d211-4d88-8f46-b73845cf38fa}.

Для начала работы с устройством необходимо открыть соответствующий ему файл.

Используемые драйвером коды ioctl и структуры данных описаны в заголовочных файлах can.h и ttcan.h.

Рекомендуемый вариант – использование подключаемых библиотек libnmcan и libnmttcan, в которых реализованы все необходимые функции для работы с драйвером.

Все ioctl, которые взаимодействуют непосредственно с памятью устройства (в данной версии - вообще все методы ioctl кроме IOCTL_BLANK) являются блокирующими. Пользовательская программа продолжит выполнение только после полной обработки запроса устройством.

Коды IOCTL являются кодами функций, если не указано иное, для преобразования в число следует использовать макрос из заголовка <winioctl.h>
CTL_CODE(FILE_DEVICE_CONTROLLER, ioctl_function_code, METHOD_BUFFERED, FILE_ANY_ACCESS)

Нумерация однотипных объектов, например буферов или каналов, всегда начинается с нуля.

3.1 Режим FIFO

В устройствах Firmware v.03 – от 23.07.2020 и новее (PCI_REV_ID: 0x21-xPCIe-CAN, 0x22-xPCIe-TTCAN и выше) доступна функция отправки данных в режиме FIFO. В этом режиме доступ непосредственно к контроллерам CAN заблокирован, выполнение соответствующих запросов IOCTL завершится ошибкой STATUS_INVALID_DEVICE_REQUEST. Включение режима FIFO на устройствах без поддержки FIFO так же завершится ошибкой.

Для обновления Firmware модулей до v.03 обратитесь к производителю.

4. Список доступных запросов ioctl

	Базовые функции	
0x100	IOCTL_READ_BAR	Чтение из адресного пространства устройства
0x101	IOCTL_WRITE_BAR	Запись в адресное пространство устройства
0x102	IOCTL_READ_CAN	Чтение из регистров CAN контроллера
0x103	IOCTL_WRITE_CAN	Запись в регистры CAN контроллера
0x104	IOCTL_MODIFY_CAN	Изменение регистра CAN контроллера
0x105	IOCTL_DEBUG	Отладочный запрос
0x106	IOCTL_BLANK	Пустой запрос ioctl
0x107	IOCTL_DEVICE_VERSION	Запрос информации об устройстве
0x108	IOCTL_DRIVER_VERSION	Запрос информации о драйвере
	Чтение данных	
0x200	IOCTL_READ_DMA_BUF	Пакетное чтение данных из буфера DMA
	Передача данных	
0x300	IOCTL_WRITE_TXBUF	Запись сообщения в буфер передачи CAN контроллера
0x301	IOCTL_WRITE_TXBUF_AND_SEND	Запись и запрос передачи сообщения
0x302	IOCTL_SEND_FROM_TXBUF	Запрос передачи сообщения из буфера
0x303	IOCTL_CHECK_TX	Проверка состояния передачи
0x304	IOCTL_WAIT_FOR_TX	Ожидание прерывания передачи
0x305	IOCTL_REMOVE_TXREQ	Сброс бита TXREQ в 0
0x306	IOCTL_ABAT	Управление отменой передачи (CAN_CTRLx.ABAT)
0x307	IOCTL_SEND_BY_TRIGGER	Запуск однократной передачи из буфера по триггеру
0x308	IOCTL_SEND_BY_TRIGGER_LOOP	Запуск многократной передачи из буфера по триггеру
0x309	IOCTL_CHECK_TRIGGER	Проверка состояния триггера
0x310	IOCTL_WRITE_DATA_FIFO	Запись данных в буфер FIFO

0x311	IOCTL_WRITE_TG_FIFO	Программирование триггера FIFO
	Конфигурация контроллера	
0x500	IOCTL_RESET_TTCAN	Полный сброс устройства
0x501	IOCTL_RESET_CANn_TTCAN	Сброс одного из CAN контроллеров
0x502	IOCTL_SETSPEED	Установка скорости работы CAN контроллера
0x503	IOCTL_GETSPEED	Чтение скорости работы CAN контроллера
0x504	IOCTL_SETSPEED_PARAMS	Установка произвольной скорости работы CAN контроллера
0x505	IOCTL_SETMODE	Установка режима работы CAN контроллера
0x506	IOCTL_GETMODE	Чтение режима работы CAN контроллера
0x507	IOCTL_DMA_ENABLE	Включение DMA для всего устройства
0x508	IOCTL_DMA_DISABLE	Выключение DMA для всего устройства
0x509	IOCTL_SET_ONESHOT	Установка режима однократной передачи
0x50A	IOCTL_GET_CAN_ERRORS	Чтение регистров ошибок CAN контроллера
0x50B	IOCTL_SET_CAN_MASKS	Установка масок и фильтров приёма сообщений
0x50D	IOCTL_SET_CAN_TIMER_TRSH	Запуск таймера с указанным периодом
0x50E	IOCTL_SET_CAN_TIMER_CEED	Установка режима и значений корректировки начального значения таймера
0x50F	IOCTL_SET_CAN_TIMER_FREE	Запуск таймера в режиме свободного счёта
0x510	IOCTL_SET_CAN_TIMER_RST_RXB	Установка/снятие режима сброса таймера по приёму сообщения в буфер CAN контроллера
0x511	IOCTL_STOP_CAN_TIMER	Остановка таймера
0x512	IOCTL_GET_CAN_TIMER	Чтение текущего значения таймера
0x513	IOCTL_START_TIMER_INT	Запуск триггера прерывания по значению таймера
0x514	IOCTL_STOP_TIMER_INT	Остановка триггера прерывания по значению таймера
0x515	IOCTL_WAIT_TIMER_INT	Ожидание прерывания таймера
0x516	IOCTL_SET_CAN_TIEMOUT	Установка значений абсолютного и интервального таймера
0x517	IOCTL_SET_DEVICEMODE	Установка режима работы контроллера (Native/FIFO)
0x518	IOCTL_GET_FIFOINFO	Получение состояния буфера FIFO
0x519	IOCTL_SET_TXPAUSE	Управление регистром TXPAUSE

4.1 Базовые Функции

4.1.1 IOCTL_READ_BAR

Назначение:
Чтение одного регистра из адресного пространства устройства (BAR)

Действие:
Функция считывает значение одного регистра в адресном пространстве.

Примечание:
Адрес регистра должен быть выровнен по границе 4 байт.

Вход - структура BAR_IOCTL_REG_REQ:

Смещение	Размер	Назначение	Примечание
0x00	4	Адрес регистра	

Выход - структура BAR_IOCTL_REG_REQ:

Смещение	Размер	Назначение	Примечание
0x00	4	Адрес регистра	Равен запрошенному адресу
0x04	4	Значение регистра	

Пример вызова:

```
	DWORD size;
	BAR_IOCTL_REG_REQ barreq;

	barreq.reg_addr = bar_addr;
	DeviceIoControl(*ttcan, IOCTL_READ_BAR, &barreq, sizeof(BAR_IOCTL_REG_REQ), &barreq,
sizeof(BAR_IOCTL_REG_REQ), &size, NULL);
	*pbuf = barreq.buf;
```

4.1.2 IOCTL_WRITE_BAR

Назначение:
Запись одного регистра в адресное пространство устройства

Действие:
Функция записывает значение одного регистра в адресном пространстве

Примечание:
Адрес регистра должен быть выровнен по границе 4 байт.

Вход - структура BAR_IOCTL_REG_REQ:

Смещение	Размер	Назначение	Примечание
0x00	4	Адрес регистра	
0x04	4	Значение регистра	

Выход - нет

Пример вызова:

```
	DWORD size;
	BAR_IOCTL_REG_REQ barreq;

	barreq.reg_addr = bar_addr;
	barreq.buf = *pbuf;
	DeviceIoControl(*ttcan, IOCTL_WRITE_BAR, &barreq, sizeof(BAR_IOCTL_REG_REQ), NULL, 0,
&size, NULL);
```

4.1.3 IOCTL_READ_CAN

Назначение:
Чтение области памяти контроллера CAN.

Действие:
Функция считывает до 16 регистров из адресного пространства контроллера CAN.

Примечание:
Отсчёт номера канала начинается с 0.

Вход - структура CAN_IOCTL_REG_REQ:

Смещение	Размер	Назначение	Примечание
0x00	4	Адрес первого регистра	
0x04	1	Номер канала	
0x15	1	Количество байт для чтения	

Выход - структура CAN_IOCTL_REG_REQ:

Смещение	Размер	Назначение	Примечание
0x00	4	Адрес первого регистра	Равен запрошенному
0x04	1	Номер канала	Равен запрошенному
0x05	16	Буфер чтения	
0x15	1	Количество считанных байт	

Пример вызова:

```
	DWORD size;
	CAN_IOCTL_REG_REQ canbuf;

	canbuf.reg_addr = can_addr;
	canbuf.channel = channel;
	memset(pbuf, 0, rsize);
	canbuf.size = rsize;

	DeviceIoControl(*ttcan, IOCTL_READ_CAN, &canbuf, sizeof(CAN_IOCTL_REG_REQ), &canbuf,
sizeof(CAN_IOCTL_REG_REQ), &size, NULL);
	memcpy_s(pbuf, rsize, &canbuf.buf, canbuf.size);
```

4.1.4 IOCTL_WRITE_CAN

Назначение:
Запись области памяти контроллера CAN

Действие:
Функция записывает до 16 регистров в адресное пространство контроллера CAN.

Примечание:
Отсчёт номера канала начинается с 0.

Вход - структура CAN_IOCTL_REG_REQ:

Смещение	Размер	Назначение	Примечание
0x00	4	Адрес первого регистра	
0x04	1	Номер канала	
0x05	16	Буфер записи	
0x15	1	Количество байт для записи	

Выход - нет

Пример вызова:

```
        DWORD size;
        CAN_IOCTL_REG_REQ canbuf;

        canbuf.reg_addr = can_addr;
        canbuf.channel = channel;
        canbuf.size = rsize;
        memcpy_s(&canbuf.buf, 8, pbuf, canbuf.size);

      DeviceIoControl(*ttcan, IOCTL_READ_CAN, &canbuf, sizeof(CAN_IOCTL_REG_REQ), NULL, 0, &size,
NULL);
```

4.1.5 IOCTL_MODIFY_CAN

Назначение:
Изменение битов одного регистра контроллера CAN

Действие:
Изменяет биты в регистре контроллера CAN

Примечание:
Отсчёт номера канала начинается с 0.

Вход - структура CAN_IOCTL_REG_REQ:

Смещение	Размер	Назначение	Примечание
0x00	4	Адрес регистра	
0x04	1	Номер канала	
0x05	1	Маска	
0x06	1	Данные	

Выход - нет

Пример вызова:

```
        DWORD size;
        CAN_IOCTL_REG_REQ canbuf;

        canbuf.reg_addr = can_addr;
        canbuf.channel = channel;
        memcpy_s(&canbuf.buf, 1, &data, 1);
        memcpy_s(&canbuf.buf + 1, 1, &mask, 1);

      DeviceIoControl(*ttcan, IOCTL_READ_CAN, &canbuf, sizeof(CAN_IOCTL_REG_REQ), NULL, 0, &size,
NULL);
```

4.1.6 IOCTL_DEVICE_VERSION

Назначение:
Чтение информации об устройстве

Действие:
Возвращает информацию о версии устройства

Примечание:
нет

Вход - нет:

Выход - структура TTCAN_DEVINFO:

Смещение	Размер	Назначение	Примечание
0x00	4	ID устройства	PCI
0x04	4	ID производителя	PCI
0x08	4	ID ревизии	PCI
0x0C	510	Имя файла устройства	WCHAR[255];

Пример вызова:

```
TTCAN_DEVINFO info;
DeviceIoControl(hTTCAN, IOCTL_DEVICE_VERSION, NULL, 0, &info, sizeof(info), &size, NULL);
```

4.1.7 IOCTL_DRIVER_VERSION

Назначение:
Чтение информации о драйвере устройства

Действие:
Возвращает информацию о версии драйвера устройства

Примечание:
Версия - в формате Major.Minor.Build.Build,
Дата сборки в формате YearH.YearL.Month.Date

Вход - нет:

Выход - структура TTCAN_DRVINFO:

Смещение	Размер	Назначение	Примечание
0x00	4	Версия	
0x04	4	Дата сборки	

Пример вызова:

```
TTCAN_DRVINFO info;
DeviceIoControl(hTTCAN, IOCTL_DRIVER_VERSION, NULL, 0, &info, sizeof(info), &size, NULL);
UINT16 build  = (UINT16)info.version;
UINT8  minor  = (UINT8)(info.version >> 16);
UINT8  major  = (UINT8)(info.version >> 24);

UINT16 year   = (UINT16)(info.build_date >> 16);
UINT8  month  = (UINT8)(info.build_date >> 8);
UINT8  day    = (UINT8)(info.build_date);
```

4.2 Чтение данных

4.2.1 IOCTL_READ_DMA_BUF

Назначение:
Чтение данных из буфера DMA

Действие:
Вычитывает данные из буфера DMA.

Примечание:
В этой версии драйвера поле таймаут игнорируется, метод возвращает максимальное доступное на момент вызова количество сообщений. В этой версии используется общий буфер DMA, соответственно поле номер канала игнорируется.

Вход - структура DMA_STR_TTCAN:

Смещение	Размер	Назначение	Примечание
0x00	4	Количество блоков для чтения	Не более 1024

Выход - структура DMA_STR_TTCAN:

Смещение	Размер	Назначение	Примечание
0x00	4	Количество считанных блоков	
0x0C	22528	Блоки данных	До 1024 блоков DMA_SLOT_TTCAN

Пример вызова:

```
	DWORD size;
	DMA_STR_TTCAN dmareq;

	dmareq.number_channel = channel;
	dmareq.number_block = count;
	dmareq.timeout = timeout;

	DeviceIoControl(*ttcan, IOCTL_READ_DMA_BUF, &dmareq, sizeof(DMA_STR_TTCAN), &dmareq,
sizeof(DMA_STR_TTCAN), &size, NULL);

	memcpy_s(dmabuf, count, &dmareq.buf, sizeof(DMA_SLOT_TTCAN) * dmareq.number_block);
```

4.3 Передача данных

4.3.1 IOCTL_WRITE_TXBUF

Назначение:
Запись сообщения в буфер передачи контроллера CAN.

Действие:
Записывает сообщение в буфер передачи контроллера CAN без запроса передачи.

Примечание:
нет

Вход - структура SEND_DATA:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера передачи	
0x02	1	Приоритет	
0x03	1	TXBnCTRL	
0x04	4	Таймаут передачи	
0x08	4	Стандартный идентификатор	
0x0C	4	Расширенный идентификатор	
0x10	8	Буфер данных сообщения	
0x18	4	Количество байт в буфере данных	

Выход - нет:

Пример вызова:

```
	DWORD size;
	TTCAN_SEND_DATA sd;

	sd.nChannel = channel;
	sd.nBufNumber = nbuf;
	sd.nPriority = prio;
	sd.timeout = timeout;

	sd.SID = sid;
	sd.EID = eid;
	memcpy_s(&sd.nData, 8, pdata, datasize);

	DeviceIoControl(*ttcan, IOCTL_WRITE_TXBUF, &sd, sizeof(TTCAN_SEND_DATA), NULL, 0, &size,
NULL);
```

4.3.2 IOCTL_WRITE_TXBUF_AND_SEND

Назначение:
Запись сообщения в буфер передачи контроллера CAN и последующая отправка.

Действие:
Записывает сообщение в буфер передачи контроллера CAN с запросом передачи.

Примечание:
Вызов блокируется до момента подтверждения отправки

Вход - структура SEND_DATA:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера передачи	
0x02	1	Приоритет	
0x03	1	TXBnCTRL	
0x04	4	Таймаут передачи	
0x08	4	Стандартный идентификатор	
0x0C	4	Расширенный идентификатор	
0x10	8	Буфер данных сообщения	
0x18	4	Количество байт в буфере данных	

Выход - нет:

Пример вызова:

```
DWORD size;
	TTCAN_SEND_DATA sd;

	sd.nChannel = channel;
	sd.nBufNumber = nbuf;
	sd.nPriority = prio;
	sd.timeout = timeout;

	sd.SID = sid;
	sd.EID = eid;
	memcpy_s(&sd.nData, 8, pdata, datasize);

	if(autorts)
		DeviceIoControl(*ttcan, IOCTL_WRITE_TXBUF_AND_SEND, &sd, sizeof(TTCAN_SEND_DATA),
NULL, 0, &size, NULL);
```

4.3.3 IOCTL_SEND_FROM_TXBUF

Назначение:
Отправка сообщения из буфера.

Действие:
Устанавливает флаг запроса отправки.

Примечание:
Вызов блокируется до момента подтверждения отправки

Вход - структура SEND_DATA:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера передачи	

Выход - нет:

Пример вызова:

```
	DWORD size;
	TTCAN_SEND_DATA_NOW sd;

	sd.channel = channel;
	sd.nBuf = nbuf;

	DeviceIoControl(*ttcan, IOCTL_SEND_FROM_TXBUF, &sd, sizeof(TTCAN_SEND_DATA_NOW), NULL, 0,
&size, NULL);
```

4.3.4 IOCTL_CHECK_TX

Назначение:
Проверка состояния отправки сообщения из буфера.

Действие:
Возвращает содержимое TXBnCTRL для выбранного буфера выбранного канала

Примечание:
нет

Вход - структура SEND_DATA:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера передачи	

Выход - структура SEND_DATA:

Смещение	Размер	Назначение	Примечание
0x03	1	TXBnCTRL	

Пример вызова:

```
	DWORD size;
	TTCAN_SEND_DATA_NOW sd;

	sd.channel = channel;
	sd.nBuf = nbuf;

	DeviceIoControl(*ttcan, IOCTL_CHECK_TX, &sd, sizeof(TTCAN_SEND_DATA_NOW), &sd,
sizeof(TTCAN_SEND_DATA_NOW), &size, NULL);

	memcpy(txbnctrl, &sd.txb_ctrl, 1);
```

4.3.5 IOCTL_WAIT_FOR_TX

Назначение:
Ожидание прерывания по отправке сообщения или таймаута.

Действие:
Блокирует выполнение потока до момента отправки сообщения.

Примечание:
нет

Вход - структура SEND_DATA:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера передачи	

Выход - нет:

Пример вызова:

```
	DWORD size;
	TTCAN_SEND_DATA_NOW sd;

	sd.channel = channel;
	sd.nBuf = nbuf;
	sd.timeout = timeout;

	DeviceIoControl(*ttcan, IOCTL_WAIT_FOR_TX, &sd, sizeof(TTCAN_SEND_DATA_NOW), NULL, 0,
&size, NULL);
```

4.3.6 IOCTL_REMOVE_TXREQ

Назначение:
Снятие запроса на отправку.

Действие:
Снимает бит TXREQ в регистре TXBnCTRL.

Примечание:
нет

Вход - структура SEND_DATA:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера передачи	

Выход - нет:

Пример вызова:

```
	DWORD size;
	TTCAN_SEND_DATA_NOW sd;

	sd.channel = channel;
	sd.nBuf = nbuf;

	DeviceIoControl(*ttcan, IOCTL_REMOVE_TXREQ, &sd, sizeof(TTCAN_SEND_DATA_NOW), NULL, 0,
&size, NULL);
```

4.3.7 IOCTL_ABAT

Назначение:
Устанавливает режим "отмена всех активных передач".

Действие:
Устанавливает режим "отмена всех активных передач".

Примечание:
нет

Вход - структура CAN_IOCTL_SETMODE_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Состояние режима отмены всех активных передач	0 - отмена всех активных передач выключена, все остальные значения - отмена всех активных передач включена

Выход - нет:

Пример вызова:

```
	DWORD size;
	CAN_IOCTL_SETMODE_REQ sd;

	sd.channel = channel;

	DeviceIoControl(*ttcan, IOCTL_ABAT, &sd, sizeof(CAN_IOCTL_SETMODE_REQ), NULL, 0, &size,
NULL);
```

4.3.8 IOCTL_SEND_BY_TRIGGER

Назначение:
Запуск однократной передачи сообщения из буфера передачи по срабатыванию триггера.

Действие:
Запускает однократную передачу по триггеру.

Примечание:
Для обоих вызовов IOCTL_SEND_BY_TRIGGER* значение триггера, находящееся в поле nTrigger, записывается в регистр CANn_TXm_TRIG (значение записывается целиком с нулевого бита).

Если поле bEpoch не равно нулю, то значение из поля nEpoch записывается в регистр CANn_TXm_TRIG_EPOCH (значение записывается целиком с нулевого бита), а бит TX_TRIGm_EPOCH_EN регистра CANn_TRIG_CTRL устанавливается в единицу.

Бит TX_TRIGm_RPT регистра CANn_TRIG_CTRL устанавливается в единицу, а биты TX_TRIGm_EN устанавливаются в “01”.
Номер буфера (поле nBuf) в этом вызове должен совпадать с номером буфера, в который были записаны данные.

Вход - структура SEND_DATA_TG:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера	
0x02	1	Флаг использования счётчика циклов	
0x03	4	Счётчик циклов	
0x07	4	Значение триггера	

Выход - нет

Пример вызова:

```
DWORD size;
	SEND_DATA_TG sd;

	sd.nChannel = channel;
	sd.nBuf = buffer;
	sd.bEpoch = bEpoch;
	sd.nEpoch = nEpoch;
	sd.nTrigger = nTrigger;
	DeviceIoControl(*ttcan, IOCTL_SEND_BY_TRIGGER, &sd, sizeof(SEND_DATA_TG), NULL, 0, &size,
NULL);
```

4.3.9 IOCTL_SEND_BY_TRIGGER_LOOP

Назначение:
Запуск многократной передачи сообщения из буфера передачи по срабатыванию триггера.

Действие:
Запускает передачу в цикле.

Примечание:
Для обоих вызовов IOCTL_SEND_BY_TRIGGER* значение триггера, находящееся в поле nTrigger, записывается в регистр CANn_TXm_TRIG (значение записывается целиком с нулевого бита).

Если поле bEpoch не равно нулю, то значение из поля nEpoch записывается в регистр CANn_TXm_TRIG_EPOCH (значение записывается целиком с нулевого бита), а бит TX_TRIGm_EPOCH_EN регистра CANn_TRIG_CTRL устанавливается в единицу.

Бит TX_TRIGm_RPT регистра CANn_TRIG_CTRL устанавливается в единицу, а биты TX_TRIGm_EN устанавливаются в “01”.
Номер буфера (поле nBuf) в этом вызове должен совпадать с номером буфера, в который были записаны данные.

Вход - структура SEND_DATA_TG:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера	
0x02	1	Флаг использования счётчика циклов	
0x03	4	Счётчик циклов	
0x07	4	Значение триггера	

Выход - нет

Пример вызова:

```
DWORD size;
	SEND_DATA_TG sd;

	sd.nChannel = channel;
	sd.nBuf = buffer;
	sd.bEpoch = bEpoch;
	sd.nEpoch = nEpoch;
	sd.nTrigger = nTrigger;

	DeviceIoControl(*ttcan, IOCTL_SEND_BY_TRIGGER_LOOP, &sd, sizeof(SEND_DATA_TG), NULL, 0,
&size, NULL);
```

4.3.10 IOCTL_CHECK_TRIGGER

Назначение:
Проверка состояния триггера

Действие:
Функция читает регистр CANn_TRIG_CTRL и проверяет биты TX_TRIGm_EN.
Если биты равны 01, то триггер занят, функция вернёт значение STATUS_WAIT_1.
Если биты равны 00, то триггер свободен и функция вернёт значение 0.

Примечание:
Для получения данных использовать GetLastError()

Вход - структура TTCAN_SEND_DATA_NOW:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Номер буфера	

Выход - нет:

Пример вызова:

```
	DWORD size;
	TTCAN_SEND_DATA_NOW sd;

	sd.channel = channel;
	sd.nBuf = buffer;

	DeviceIoControl(*ttcan, IOCTL_CHECK_TRIGGER, &sd, sizeof(TTCAN_SEND_DATA_NOW), NULL, 0,
&size, NULL);
```

4.4 Конфигурация контроллера

4.4.1 IOCTL_RESET_TTCAN

Назначение:
Полный сброс устройства

Действие:
Выполняет полный сброс устрйства

Примечание:
Полный сброс включает в себя сброс всех доступных CAN-контроллеров.

Вход - нет

Выход - нет

Пример вызова:

```
DWORD size;
DeviceIoControl(*ttcan, IOCTL_RESET_TTCAN, NULL, 0, NULL, 0, &size, NULL);
```

4.4.2 IOCTL_RESET_CANn_TTCAN

Назначение:
Сброс одного из CAN-контроллеров

Действие:
Выполняет сброс одного из CAN-контроллеров

Примечание:
нет

Вход - структура CAN_IOCTL_SETMODE_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	

Выход - нет

Пример вызова:

```
DWORD size;
CAN_IOCTL_SETMODE_REQ smr;
smr.channel = channel;
DeviceIoControl(*ttcan, IOCTL_RESET_CANn_TTCAN, &smr, sizeof(smr), NULL, 0, &size, NULL);
```

4.4.3 IOCTL_SETSPEED

Назначение:
Установка скорости работы контроллера

Действие:
Устанавливает одну из предопределённых стандартных скоростей работы контроллера

Примечание:
Доступно только когда контроллер находится в режиме конфигурации

Вход - структура CAN_IOCTL_SETSPEED_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	4	Константа скорости	125, 250, 500 или 1000

Выход - нет

Пример вызова:

```
DWORD size;
CAN_IOCTL_SETSPEED_REQ spdreq;

spdreq.channel = channel;
spdreq.can_speed = speed;

DeviceIoControl(*ttcan, IOCTL_SETSPEED, &spdreq, sizeof(spdreq), NULL, 0, &size, NULL);
```

4.4.4 IOCTL_GETSPEED

Назначение:
Получение скорости работы контроллера

Действие:
Получает данные о текущей скорости работы контроллера

Примечание:
Вернёт содержимое регистров конфигурации и при возможности - соответствующую данным настройкам стандартную скорость. Описание содержания регистров CAN_CNFx доступно в документе "Руководство по программированию модуля (m)PCIe-TTCAN в разделе "6.5 Конфигурация скорости шины CAN".

Вход - структура CAN_IOCTL_SETSPEED_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	

Выход - структура CAN_IOCTL_SETSPEED_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	4	Константа скорости	0, 125, 250, 500 или 1000
0x05	3	CAN_CNFx	Содержимое регистров конфигурации

Пример вызова:

```
	DWORD size;
	CAN_IOCTL_SETSPEED_REQ spdreq;

	spdreq.channel = channel;

	DeviceIoControl(*ttcan, IOCTL_GETSPEED, &spdreq, sizeof(spdreq), &spdreq, sizeof(spdreq),
&size, NULL);

	*speed = spdreq.can_speed;
	memcpy(params, &spdreq.params, 3);
```

4.4.5 IOCTL_SETSPEED_PARAMS

Назначение:
Установка произвольной скорости работы контроллера

Действие:
Устанавливает произвольную скорость работы контроллера

Примечание:
Доступно только в режиме конфигурации. Описание содержания регистров CAN_CNFx доступно в документе "Руководство по программированию модуля (m)PCIe-TTCAN в разделе "6.5 Конфигурация скорости шины CAN".

Вход - структура CAN_IOCTL_SETSPEED_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x05	3	CAN_CNFx	Содержимое регистров конфигурации

Выход - нет

Пример вызова:

```
	DWORD size;
	CAN_IOCTL_SETSPEED_REQ spdreq;

	spdreq.channel = channel;
	memcpy(&spdreq.params, params, 3);

	DeviceIoControl(*ttcan, IOCTL_SETSPEED_PARAMS, &spdreq, sizeof(spdreq), NULL, 0, &size,
NULL);
```

4.4.6 IOCTL_SETMODE

Назначение:
Установка режима работы контроллера CAN

Действие:
Переключает режимы работы контроллера.

Примечание:
Режим конфигурации доступен всегда. Переключение в любой другой режим осуществляется только из режима конфигурации.

Вход - структура CAN_IOCTL_SETMODE_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Режим работы	CAN_WORK = 0 CAN_SLEEP = 1 CAN_LOOP = 2 CAN_MON = 3 CAN_CONF = 4

Выход - нет

Пример вызова:

```
DWORD size;
CAN_IOCTL_SETMODE_REQ modreq;

modreq.can_mode = mode;
modreq.channel = channel;

DeviceIoControl(*ttcan, IOCTL_SETMODE, &modreq, sizeof(modreq), NULL, 0, &size, NULL);
```

4.4.7 IOCTL_GETMODE

Назначение:
Чтение режима работы контроллера CAN

Действие:
Считывает режим работы контроллера

Примечание:
нет

Вход - структура CAN_IOCTL_SETMODE_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	

Выход - структура CAN_IOCTL_SETMODE_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Режим работы	CAN_WORK = 0 CAN_SLEEP = 1 CAN_LOOP = 2 CAN_MON = 3 CAN_CONF = 4

Пример вызова:

```
	DWORD size;
	CAN_IOCTL_SETMODE_REQ modreq;

	modreq.channel = channel;

	DeviceIoControl(*ttcan, IOCTL_SETMODE, &modreq, sizeof(modreq), &modreq, sizeof(modreq),
&size, NULL);

	*mode = (CAN_MODE)modreq.can_mode;
```

4.4.8 IOCTL_DMA_ENABLE

Назначение:
Включение режима DMA

Действие:
Включает DMA

Примечание:
нет

Вход - нет

Выход - нет

Пример вызова:

```
DWORD size;
DeviceIoControl(*ttcan, IOCTL_DMA_ENABLE, NULL, 0, NULL, 0, &size, NULL);
```

4.4.9 IOCTL_DMA_DISABLE

Назначение:
Выключение режима DMA

Действие:
Выключает DMA

Примечание:
нет

Вход - нет

Выход - нет

Пример вызова:

```
DWORD size;
DeviceIoControl(*ttcan, IOCTL_DMA_DISABLE, NULL, 0, NULL, 0, &size, NULL);
```

4.4.10 IOCTL_SET_ONESHOT

Назначение:
Переключение режима однократной передачи

Действие:
Включает или выключает режим однократной передачи.

Примечание:
В этом режиме CAN контроллер не будет пытаться повторно передать сообщение в случае возникновения ошибки. Значение can_mode > 0 включает режим однократной передачи, can_mode = 0 - выключает.

Вход - структура CAN_IOCTL_SETMODE_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Желаемое состояние режима однократной передачи	

Выход - нет

Пример вызова:

```
	DWORD size;
	CAN_IOCTL_SETMODE_REQ modreq;

	modreq.channel = channel;
	modreq.can_mode = oneshot;

	DeviceIoControl(*ttcan, IOCTL_SET_ONESHOT, &modreq, sizeof(modreq), &modreq,
sizeof(modreq), &size, NULL);
```

4.4.11 IOCTL_GET_CAN_ERRORS

Назначение:
Чтение ошибок CAN-контроллера

Действие:
Считывает состояние регистров EFLG, TEC, REC с выбранного контроллера

Примечание:
Описание содержания регистров доступно в документе "Руководство по программированию модуля (m)PCIe-TTCAN в разделе "6.6 Ошибки CAN-шины".

Вход - структура CAN_ERROR_INFO:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	

Выход - структура CAN_ERROR_INFO:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	EFLG	
0x02	1	TEC	
0x03	1	REC	

Пример вызова:

```
	DWORD size;

	errorinfo->channel = channel;

    DeviceIoControl(*ttcan, IOCTL_GET_CAN_ERRORS, errorinfo, sizeof(CAN_ERROR_INFO), errorinfo, sizeof(CAN_ERROR_INFO), &size, NULL);
```

4.4.12 IOCTL_SET_CAN_MASKS

Назначение:
Установка масок и фильтров приёма CAN контроллера.

Действие:
В зависимости от выбранного канала (**channel**) и номера фильтра (**filter**) функция записывает значение **rxb_mode** в биты RXM регистра **RXBn*CTRL****.

* n – номер буфера.

** См. раздел 6.8.1 и 6.8.2 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

Далее в зависимости от выбранного идентификатора (**ident**) и номера фильтра (**filter**) функция записывает значения масок и фильтров в соответствующие регистры*.

* См. раздел 6.9 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

Примечание:
нет

Вход - структура TTCAN_MASKS:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Режим работы буфера	
0x02	1	Номер фильтра	
0x03	1	Идентификатор	
0x04	4	Маска идентификатора	
0x08	4	Маска фильтра	

Выход - нет

Пример вызова:

```
	DWORD size;
	TTCAN_MASKS masks;

	masks.channel = channel;
	masks.filter = filter;
	masks.ident = ident;
	masks.id_filter = id_filter;
	masks.id_mask = id_mask;
	masks.rxb_mode = rxb;

	DeviceIoControl(*ttcan, IOCTL_SET_CAN_MASKS, &masks, sizeof(TTCAN_MASKS), NULL, 0, &size,
NULL);
```

4.4.13 IOCTL_SET_CAN_TIMER_TRSH

Назначение:
Запуск таймера с указанным периодом.

Действие:
Запуск таймера с указанным периодом.
Значение поля **nValue** записывается в регистр **CANn*_TIMER_TRSH**** (значение записывается целиком с нулевого бита).

Биты RST, ENABLE и NTU_MODE регистра **CANn*_TIMER_CTRL***** устанавливаются в единицу, а битовая маска EPOCH_MASK устанавливается в соответствии с параметром **nEpochBits**. Значения остальных битов данного регистра не изменяются.

* n – номер канала.

** См. раздел 5.2.2 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

*** См. раздел 5.2.4 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

Примечание:
нет

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x02	1	Битовая маска	nEpochBits
0x07	4	Значение	nValue

Выход - нет

Пример вызова:

```
        DWORD size;
        TTCAN_TIMER_TRSH trsh;

        trsh.channel = channel;
        trsh.nValue = nValue;
        trsh.bEpoch = bEpoch;
        trsh.nEpoch = nEpoch;

      DeviceIoControl(*ttcan, IOCTL_SET_CAN_TIMER_TRSH, &trsh, sizeof(TTCAN_TIMER_TRSH), NULL, 0,
&size, NULL);
```

4.4.14 IOCTL_SET_CAN_TIMER_CEED

Назначение:
Установка режима и значений корректировки начального значения таймера.

Действие:
Значение поля **nValue** записывается в регистр **CANn*_TIMER_TRSH**** (значение записывается целиком с нулевого бита).

Биты RST, ENABLE и NTU_MODE регистра **CANn*_TIMER_CTRL***** устанавливаются в единицу, а битовая маска EPOCH_MASK устанавливается в соответствии с параметром **nEpochBits**. Значения остальных битов данного регистра не изменяются.

* n – номер канала.

** См. раздел 5.2.2 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

*** См. раздел 5.2.4 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

Примечание:
нет

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Флаг использования битовой маски	bEpoch
0x02	1	Битовая маска	nEpochBits
0x03	4	nEpoch	nEpoch
0x07	4	Значение	nValue

Выход - нет

Пример вызова:

```
        DWORD size;
        TTCAN_TIMER_TRSH trsh;

        trsh.channel = channel;
        trsh.nValue = nValue;
        trsh.bEpoch = bEpoch;
        trsh.nEpoch = nEpoch;

      DeviceIoControl(*ttcan, IOCTL_SET_CAN_TIMER_CEED, &trsh, sizeof(TTCAN_TIMER_TRSH), NULL, 0,
&size, NULL);
```

4.4.15 IOCTL_SET_CAN_TIMER_FREE

Назначение:
Запуск таймера в режиме свободного счёта.

Действие:
В регистре **CANn*_TIMER_CTRL**** биты EPOCH_CEED_EN, RST_ON_RXB0, RST_ON_RXB1, NTU_MODE и CEED_EN устанавливаются в ноль, биты RST и ENABLE устанавливаются в единицу, а битовая маска EPOCH_MASK устанавливается в соответствии с параметром **nEpochBits**. Значения остальных битов данного регистра не изменяются.

* n – номер канала.

** См. раздел 5.2.4 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

Примечание:
нет

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x02	1	Битовая маска	nEpochBits

Выход - нет

Пример вызова:

```
        DWORD size;
        TTCAN_TIMER_TRSH trsh;

        trsh.channel = channel;
        trsh.nEpochBits = epochBits;

      DeviceIoControl(*ttcan, IOCTL_SET_CAN_TIMER_FREE, &trsh, sizeof(TTCAN_TIMER_TRSH), NULL, 0,
&size, NULL);
```

4.4.16 IOCTL_SET_CAN_TIMER_RST_RXB

Назначение:
Установка/снятие режима сброса таймера по приёму сообщения в буфер CAN контроллера.

Действие:
В регистре **CANn*_TIMER_CTRL**** бит RST_ON_RXB0 устанавливается в соответствии со значением бита 0 параметра **nValue**, а бит RST_ON_RXB1 устанавливается в соответствии со значением бита 1 параметра **nValue**. Значения остальных битов данного регистра не изменяются.

* n – номер канала.

** См. раздел 5.2.4 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

Примечание:
нет

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x07	4	Значение	nValue

Выход - нет

Пример вызова:

```
	DWORD size;
	TTCAN_TIMER_TRSH trsh;

	trsh.channel = channel;
	trsh.nValue = nValue;

    DeviceIoControl(*ttcan, IOCTL_SET_CAN_TIMER_RST_RXB, &trsh, sizeof(TTCAN_TIMER_TRSH), NULL,
0, &size, NULL);
```

4.4.17 IOCTL_STOP_CAN_TIMER

Назначение:
Остановка таймера.

Действие:
В регистре **CANn*_TIMER_CTRL**** бит ENABLE устанавливается в ноль. Значения остальных битов данного регистра не изменяются.

* n – номер канала.

** См. раздел 5.2.4 документа "*Руководство по программированию модуля "mPCIe-TTCAN*".

Примечание:
нет

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	

Выход - нет

Пример вызова:

```
	DWORD size;
	TTCAN_TIMER_TRSH trsh;

	trsh.channel = channel;

	DeviceIoControl(*ttcan, IOCTL_STOP_CAN_TIMER, &trsh, sizeof(TTCAN_TIMER_TRSH), NULL, 0,
&size, NULL);
```

4.4.18 IOCTL_GET_CAN_TIMER

Назначение:
Чтение текущего значения таймера.

Действие:
Регистр **CANn*_TIMER**** читается в поле **nValue**, а регистр **CANn*_TIMER_EPOCH***** читается в поле **nEpoch**.

Примечание:
нет

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	

Выход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x03	4	nEpoch	nEpoch
0x07	4	Значение	nValue

Пример вызова:

```
	DWORD size;
	TTCAN_TIMER_TRSH trsh;

	trsh.channel = channel;

	DeviceIoControl(*ttcan, IOCTL_GET_CAN_TIMER, &trsh, sizeof(TTCAN_TIMER_TRSH), &trsh,
sizeof(TTCAN_TIMER_TRSH), &size, NULL);

	*value = trsh.nValue;
	*epoch = trsh.nEpoch;
```

4.4.19 IOCTL_START_TIMER_INT

Назначение:
Запуск триггера прерывания по значению таймера.

Действие:
Значение поля **nValue** записывается в регистр **CANn*_INT_TRIG**** (значение записывается целиком с нулевого бита).

Если значение **bEpoch** отлично от нуля, то значение поля **nEpoch** записывается в регистр **CANn*_INT_TRIG_EPOCH***** (значение записывается целиком с нулевого бита), а бит INT_TRIG_EPOCH_EN регистра **CANn*_TRIG_CTRL****** устанавливается в единицу. Иначе бит INT_TRIG_EPOCH_EN регистра **CANn*_TRIG_CTRL****** устанавливается в ноль.

В регистре **CANn*_TRIG_CTRL****** бит INT_TRIG_RPT устанавливается в единицу, а биты INT_TRIG_EN регистра устанавливаются в “01”. Значения остальных битов данного регистра не изменяются.

* n – номер канала.

** См. раздел 5.3.5 документа “*Руководство по программированию модуля “mPCIe-TTCAN*”.

*** См. раздел 5.3.8 документа “*Руководство по программированию модуля “mPCIe-TTCAN*”.

**** См. раздел 5.3.3 или 5.3.4 документа “*Руководство по программированию модуля “mPCIe-TTCAN*”.

Примечание:
нет

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	1	Флаг использования битовой маски	bEpoch
0x07	4	Значение	nValue

Выход - нет

Пример вызова:

```
	DWORD size;
	TTCAN_TIMER_TRSH trsh;

	trsh.channel = channel;
	trsh.bEpoch = bEpoch;
	trsh.nValue = nValue;

	DeviceIoControl(*ttcan, IOCTL_START_TIMER_INT, &trsh, sizeof(TTCAN_TIMER_TRSH), NULL, 0,
&size, NULL);
```

4.4.20 IOCTL_STOP_TIMER_INT

Назначение:
Остановка триггера прерывания по значению таймера.

Действие:
В регистре **CANn*_TRIG_CTRL**** биты INT_TRIG_EPOCH_EN и INT_TRIG_RPT устанавливаются в ноль, а биты INT_TRIG_EN устанавливаются в “10”. Значения остальных битов данного регистра не изменяются.

Примечание:
нет

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	

Выход - нет

Пример вызова:

```
	DWORD size;
	TTCAN_TIMER_TRSH trsh;

	trsh.channel = channel;

	DeviceIoControl(*ttcan, IOCTL_STOP_TIMER_INT, &trsh, sizeof(TTCAN_TIMER_TRSH), NULL, 0, &size, NULL);
```

4.4.21 IOCTL_WAIT_TIMER_INT

Назначение:
Ожидание прерывания таймера

Действие:
Ожидание прерывания таймера

Примечание:
Выполнение блокируется до срабатывания прерывания или достижения таймаута

Вход - структура TTCAN_TIMER_TRSH:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x03	4	Таймаут прерывания, в микросекундах	nEpoch

Выход - нет

Пример вызова:

```
	DWORD size;
	TTCAN_TIMER_TRSH trsh;

	trsh.channel = channel;

	DeviceIoControl(*ttcan, IOCTL_WAIT_TIMER_INT, &trsh, sizeof(TTCAN_TIMER_TRSH), NULL, 0,
&size, NULL);
```

4.4.22 IOCTL_SET_CAN_TIMEOUT

Назначение:
Установка таймаута таймера.

Действие:
Поле **absolute** пишется в регистр **CANn*_TIMEOUT_ABSOLUTE****, а поле **interval** пишется в регистр **CANn*_TIMEOUT_INTERVAL*****.
Значение обоих полей в микросекундах, допустимые значения - 0x0 .. 0x3FFFFFF

Примечание:
нет

Вход - структура TTCAN_TIMEOUTS:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	4	Интервальный таймаут	
0x07	4	Абсолютный таймаут	

Выход - нет

Пример вызова:

```
DWORD size;
TTCAN_TIMEOUTS timeouts;

timeouts.channel = channel;
timeouts.absolute = absolute;
timeouts.interval = interval;
```

DeviceIoControl(*ttcan, IOCTL_SET_CAN_TIMEOUT, &timeouts, sizeof(TTCAN_TIMEOUTS), NULL, 0, &size, NULL);

4.5 Работа в режиме FIFO

4.5.1 IOCTL_SET_DEVICEMODE

Назначение:
Установка режима работы контроллера CAN.

Действие:
Устанавливает один из режимов работы контроллера CAN. По умолчанию контроллер работает в режиме NATIVE, возможен прямой доступ к памяти контроллера. Доступны режимы - NATIVE, FIFO.
Примечание:
см. структуру CTRL_MODE

Вход - структура CAN_IOCTL_SETMODE_REQ:

Смещение	Размер	Назначение	Примечание
0x00	1	Режим работы	
0x01	1	Номер канала	

Выход - нет

4.5.2 IOCTL_WRITE_DATA_FIFO

Назначение:
Запись данных в буфер FIFO

Действие:
Записывает данные в буфер FIFO. Перед записью проводится проверка свободного места, если места недостаточно, попытка записи произведена не будет.
Примечание:
нет

Вход - структура TTCAN_FIFO_REQUEST:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	
0x01	4	SID	
0x05	4	EID	
0x09	1	dlc	Не используется
0xA	8	data	8 байт
0x12	4	Размер буфера данных	
0x16	1	ID сообщения	
0x17	1	Флаг HPFIFO	1 - использовать hpfifo, иначе - fifo

Выход - нет

4.5.3 IOCTL_GET_FIFOINFO

Назначение:
Получение данных о буфере FIFO

Действие:
Получение данных о буфере FIFO
Примечание:
нет

Вход - структура TTCAN_FIFO_REQUEST:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	

Выход - структура TTCAN_FIFO_REQUEST:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	TTCAN_FIFO_REQUEST.channel
0x01	4	Регистр reg_fifo_buf	TTCAN_FIFO_REQUEST.sid
0x05	4		
0x09	1		
0xA	8		
0x12	4	Количество сообщений в буфере	TTCAN_FIFO_REQUEST.size
0x16	1	ID последнего сообщения	TTCAN_FIFO_REQUEST.msgid
0x17	1	Флаг HPFIFO	TTCAN_FIFO_REQUEST.isHPFIFO

4.5.4 IOCTL_SET_TXPAUSE

Назначение:
Управление регистром TXPAUSE

Действие:
Управление регистром TXPAUSE
Примечание:
нет

Вход - структура TTCAN_FIFO_REQUEST:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	TTCAN_FIFO_REQUEST.channel
0x01	4	Значение 1 включает, 0 - выключает txpause	TTCAN_FIFO_REQUEST.sid

Выход - нет

4.5.5 IOCTL_WRITE_TG_FIFO

Назначение:
Управление триггерами FIFO

Действие:
Управление триггерами FIFO
Примечание:
нет

Вход - структура TTCAN_FIFO_REQUEST:

Смещение	Размер	Назначение	Примечание
0x00	1	Номер канала	TTCAN_FIFO_REQUEST.channel
0x01	4	Значение делителя единицы сетевого времени	TTCAN_FIFO_REQUEST.sid
0x05	4	Значение таймера в единицах сетевого времени	TTCAN_FIFO_REQUEST.eid
0x09	1	Значение счётчика циклов	TTCAN_FIFO_REQUEST.dlc
0xA	8		
0x12	4	Не используется	
0x16	1	Не используется	
0x17	1	Не используется	

Выход - нет

5.0 Обновление руководства

Версия	Дата	Примечание
3.0	22.01.2021	Руководство универсального драйвера создано
3.1	26.01.2021	Исправлены неточности во вводной части документа.