

Процессорное ядро .....	9
Memory Scrubber .....	9
Основные характеристики .....	9
Карта регистров .....	9
Регистры .....	10
CTRL [0x0]: Регистр управления .....	10
BAD_ADDR [0x4]: Регистр адреса обнаруженной ошибки .....	11
CUR_ADDR [0x8]: Регистр текущего тестируемого адреса .....	11
TIMER [0xC]: Регистр управления интервалами .....	11
Ax_INIT [0x10...]: Начальный адрес диапазона x .....	11
Ax_FIN [0x14...]: Конечный адрес диапазона x .....	12
SCTRL [0x50]: Управление сигнализацией ошибок .....	12
ECNT0 [0x54]: Регистры счетчиков ошибок 0-3 .....	13
Применение .....	13
Memory Controller (MEMC) .....	14
Управление частотой ROM .....	14
Управление EXTMEM .....	14
Карта регистров Memory Controller .....	14
Регистры .....	15
ROM Key Register (MEMC_ROM_KEY) [0x00] .....	15
ROM Control Register (MEMC_ROM_CTRL) [0x04] .....	15
ROM Clock Divider Register (MEMC_ROM_CLKDIV) [0x08] .....	15
ROM Status Register (MEMC_ROM_ST) [0x0C] .....	15
External Memory Control Register (MEMC_EXTMEM_CTRL) [0x10] .....	16
<b>Алгоритм загрузки программы в ROM</b> .....	16
EXTMEM .....	17
CMM – Блок управления тактовыми сигналами .....	19
Источники тактовых сигналов в системе .....	19
Частоты .....	22
Сброс .....	22
Всегда работающие тактовые сигналы .....	22
Диаграмма переключения .....	22
Карта регистров CMM .....	23
Регистры .....	24
Status Register (ST) [0x00] .....	24
Interrupt Mask Register (MSK) [0x04] .....	24
Clock Source Enable (CLKSRCEN) [0x08] .....	24
System Clock Select Register (SYSCLKSEL) [0x0C] .....	25
AHB Clock Enable Register (AHBCLKEN) [0x14] .....	25
APB0 Clock Divider Register (APB0CLKDIV) [0x18] .....	26
APB0 Clock Enable Register (APB0CLKEN) [0x1C] .....	26

APB1 Clock Divider Register (APB1CLKDIV) [0x20] .....	28
APB1 Clock Enable Register (APB1CLKEN) [0x24]. Управление тактированием периферийных устройств на шине APB1.28	
MILSTD PLL Control Register (PLLCTRL_MILSTD) [0x28]. .....	28
SPW PLL Control Register (PLLCTRL_SPW) [0x2C]. .....	29
System PLL Control Register (PLLCTRL_SYS) [0x30]. .....	29
Порты ввода/вывода GPIO.....	30
Основные возможности GPIO.....	30
Структурная схема .....	30
Прерывания .....	31
Список альтернативных функций GPIO .....	33
Карта регистров GPIO .....	35
Регистры .....	36
GPIO_DIR_SET / GPIO_DIR_CLR [0x0/0x4]: парные регистры управления режимом работы выходных буферов порта. ...	36
GPIO_DATA_SET/GPIO_DATA_CLR [0x8 / 0xC]: парные регистры данных GPIO. ....	36
GPIO_INTEN_SET / GPIO_INTEN_CLR [0x10 / 0x14]: парные регистры разрешения генерации прерывания по событиям на входах GPIO.....	37
GPIO_INTTYPE_SET / GPIO_INTTYPE_CLR [0x18 / 0x1C]: парные регистры установки типа прерывания (по фронту/уровню) генерируемого GPIO. ....	37
GPIO_INTPOL_SET / GPIO_INTPOL_CLR [0x20 / 0x24]: парные регистры установки полярности события GPIO, по которому генерируется прерывание. ....	38
GPIO_INT [0x28]: регистр статуса прерываний GPIO. ....	38
GPIO_ALTF0, GPIO_ALTF1, GPIO_ALTF2 [0x2C / 0x30 / 0x34]: регистры GPIO_ALTF 0 и 1 управляют GPIO_MUX. ....	39
Контроллер CAN интерфейса .....	40
Регистры режима Basic-CAN .....	42
Регистры режима Pelican .....	45
Аппаратные фильтры.....	50
Общие регистры контроллера CAN.....	52
Контроллер интерфейса MILSTD1553B (ГОСТ Р 52070-2003).....	54
Перечень выводов.....	54
Параметры конфигурации.....	55
Функционирование .....	56
Перечень регистров .....	56
Прерывания .....	56
Контроллер Шины .....	57
Оконечное Устройство .....	57
Монитор Шины.....	57
Регистры .....	57
Config [0x0]: Регистр настройки .....	57
BC_ADDR [0x4]: Регистр начального адреса очереди заданий КШ (по записи).....	58
Control [0x8]: Регистр управления.....	58
Timer [0xC]: Регистр настройки таймера позволяет сконфигурировать временные параметры работы шины, отличные от стандартных значений. ....	59
LSTAT [0x10]: Регистр статуса последнего завершенного сообщения содержит копию статусного слова дескриптора....	59
BM_ADDR_INIT [0x14]: Начальный адрес буфера МШ. ....	59
BM_ADDR_LAST [0x18]: Конечный адрес буфера МШ.....	60
BM_ADDR_IRQ [0x1C]: Адрес генерации прерывания МШ. ....	60
BM_ADDR_CUR [0x20]: Адрес следующей ячейки, записываемой МШ. ....	60

BM_TIMER [0x24]: Регистр таймера МШ .....	60
RT_MCC [0x28]: Регистр маскирования команд удаленному терминалу .....	60
RT_CFG [0x2C]: Регистр настройки удаленного терминала .....	61
RT_ADDR [0x30]: Регистр настройки адреса дескрипторов ОУ .....	62
IRQ_ENA [0x34]: Регистр разрешения прерываний .....	62
IRQ_FLAG [0x38]: Регистр флагов прерываний .....	63
REVISION [0x3C]: Регистр версии контроллера .....	64
Режим контроллера шины .....	65
Дескрипторы .....	65
Режим оконечного устройства .....	68
Отключение передатчика .....	68
Дескрипторы .....	69
Режим Data Wrap-around .....	70
Монитор шины .....	70
I2C .....	72
Основные характеристики модуля .....	72
Структурная схема .....	72
Регистры I2C .....	73
Карта регистров интерфейса I2C .....	73
Configuration Register [0x00]: .....	73
Control Register [0x04]: .....	74
Status Register [0x08]: .....	74
Address Register [0x0C]: .....	76
Prescaler Register [0x10]: .....	76
Mask Register [0x14]: .....	77
Transmit FIFO [0x18]: .....	77
Receive FIFO [0x1C]: .....	77
Transmit FIFO Unread Words [0x20]: .....	78
Receive FIFO Threshold [0x24]: .....	78
Программирование I2C .....	78
Работа I2C в режиме мастер: .....	78
OWI .....	79
Основные характеристики модуля .....	79
Структурная схема .....	79
Перечень блоков OWI .....	79
Регистры OWI .....	80
OWI_CFG [0x00]: .....	80
OWI_BUF [0x04]: .....	81
OWI_ST [0x08]: .....	81
OWI_MSK [0x0C]: .....	82
OWI_PRCR [0x10]: .....	83
OWI_CTRL [0x14]: .....	83
SPI .....	85
Основные особенности .....	85
Структурная схема .....	85
Перечень блоков SPI .....	86

Режим «ведущий» (master) .....	86
Процедура работы с «ведущим» .....	86
Особенности работы в режиме «ведущий» .....	87
Режим «ведомый» (slave) .....	87
Дуплексный и симплексный режимы работы .....	88
Две однонаправленные линии данных .....	88
Одна двунаправленная линия данных .....	88
Передача данных .....	88
Приём данных .....	89
Режим работы с входом выбора микросхемы .....	89
Статусы и прерывания модуля SPI .....	89
Карта регистров модуля SPI .....	90
Регистры .....	90
SPI_CTRL [0x00] .....	90
SPI_CFG [0x04] .....	91
SPI_MSK [0x08] .....	92
SPI_ST [0x0C] .....	92
SPI_BUFTX / SPI_BUFRX [0x10] .....	93
UART .....	94
Основные возможности .....	94
Формат посылки .....	94
Структурная схема .....	95
Делитель частоты .....	96
Высокоскоростной режим .....	97
Приёмник .....	97
Буфер приёмника .....	98
Передатчик .....	98
Буфер передатчика .....	99
Флаги и события .....	99
Прерывания .....	100
Дополнительные функции .....	100
Таймер тайм-аута .....	100
Генерация и распознавание сигнала break .....	100
Особые режимы работы .....	101
Эхо-режим .....	101
Режим внутренней петли .....	101
Режим внешней петли .....	101
9-битный режим .....	102
Аппаратный контроль обмена .....	103
Карта регистров .....	104
Configuration Register (CFG) [0x00] .....	104
Baud Rate Register (BDR) [0x04] .....	106
Tx FIFO Level Register (TXFIFOLVL) [0x08] .....	106
Rx FIFO Level Register (RXFIFOLVL) [0x0C] .....	107
Nine Bit Mode Mask Register (NBMSK) [0x10] .....	107
Nine Bit Mode Address Register (NBADDR) [0x14] .....	107

Interrupt Mask Register (MSK) [0x18].....	107
Control Register (CTRL) [0x1C].....	108
Transmitter Data Buffer Register (TX) [0x20].....	108
Receiver Data Buffer Register (RX) [0x24] .....	109
Status Register (ST) [0x28] .....	109
SPW (SpaceWire) .....	112
Основные особенности .....	112
Структурная схема .....	112
Описание основных блоков .....	115
Установка скорости передачи данных.....	116
Маркеры времени.....	116
Работа с пакетами прерываний и подтверждений .....	116
Работа с пакетами прерываний.....	117
Игнорирование пакетов прерываний.....	117
Работа с пакетами подтверждений .....	117
Игнорирование пакетов подтверждений прерываний .....	117
Прямой доступ к памяти (DMA).....	118
Порядок байтов в памяти.....	118
Работа DMA каналов.....	119
Дескрипторы .....	119
Первое 32-битное слово дескриптора.....	119
Второе 32-битное слово дескриптора .....	120
Третье 32-битное слово дескриптора .....	120
Переполнение буфера данных при приеме .....	121
Ошибки на системной шине.....	121
RMAP .....	122
Пакеты RMAP .....	122
Доступ к основной памяти.....	122
Ограничение на адреса, указанные в командах RMAP .....	122
CRC принятого пакета.....	122
Ошибки в пакетах RMAP .....	122
Порядок работы со SPW .....	123
Формат пакета RMAP для передачи.....	123
Тестовый режим работы .....	124
Частотные ограничения .....	125
Карта регистров.....	125
SPW_CTRL [0x00].....	126
SPW_CFG [0x04].....	127
SPW_MSK [0x08] .....	128
SPW_ST [0x0C].....	130
SPW_DIV [0x10] .....	132
SPW_PAUSE [0x14] .....	132
SPW_MSK_INT [0x18].....	133
SPW_MSK_ACK [0x1C].....	133
SPW_WAITACK_CNT [0x20].....	133
SPW_INT_ACK [0x24] .....	133

SPW_INTPACK [0x28].....	134
SPW_ACK_NOT [0x2C].....	134
SPW_TIME_CNT [0x30]. ....	134
SPW_TIME [0x34].....	134
SPW_TX_DESCR_ADDR [0x38]. ....	134
SPW_RX_DESCR_ADDR [0x3C]. ....	134
SPW_LADDR [0x40]. ....	134
SPW_KEY [0x44]. ....	135
SPW_ST_RMAP [0x48]. ....	135
SPW_MSK_INT_RMAP [0x4C]. ....	136
SPW_RMAP_REGION_BASE_ADDR_x [0x50 / 0x54 / 0x58 / 0x5C]. ....	137
SPW_RMAP_REGION_EXTENDED_ADDR [0x60]. ....	137
SPW_RMAP_REGION_SIZE [0x64 / 0x68 / 0x6C / 0x70]. ....	137
Таймер сна (SLEEP TIMER) .....	138
Общее описание.....	138
Регистры модуля STIMER.....	138
Sleep Timer Control (STIMER_CTRL) [0x00]. ....	138
Sleep Timer Period (STIMER_PERIOD) [0x04]. ....	139
Таймеры.....	140
Структурная схема .....	141
Простой таймер .....	141
Принцип работы .....	141
Статусы и прерывания.....	141
Алгоритм работы.....	142
Таймер с внешней остановкой.....	142
Принцип работы .....	142
Статусы и прерывания.....	142
Алгоритм работы.....	142
Межсобытийный таймер .....	142
Принцип работы .....	142
Статусы и прерывания.....	143
Алгоритм работы.....	143
Таймер-счётчик.....	143
Принцип работы .....	143
Статусы и прерывания.....	144
Алгоритм работы .....	144
Для модуля в режиме таймера: .....	144
Для модуля в режиме счетчика: .....	144
Карта регистров.....	145
TMR_CTRL [0x0C].....	145
TMR_CFG [0x04]. ....	145
TMR_PERIOD [0x08]. ....	146
TMR_VALUE [0x0C]. ....	146
TMR_MSK [0x10]. ....	146
TMR_ST [0x14]. ....	146
Сторожевой таймер (WDT).....	148

Общее описание.....	148
Регистры сторожевого таймера.....	148
WDT_LOAD [0x00]. .....	148
WDT_VAL [0x04] .....	149
WDT_CTRL [0x08].....	149
WDT_CLR [0x0C] .....	149
WDT_INTRAW [0x10].....	149
WDT_INT [0x14] .....	149
WDT_LOCK [0x18] .....	150
WDT_TCR [0x1C] .....	150
WDT_TOP [0x20].....	150
Система управления сбросом и питанием (PMM) .....	151
Основные функции.....	151
Структурная схема .....	151
Управление режимами сна .....	151
Формирование сигналов сброса .....	151
Регистры PMM .....	152
PMM_CTRL [0x00].....	152
PMM_ST [0x04]. Регистр очищается при чтении.....	152
Модуль PLIC .....	154
ID прерываний .....	155
Прерывания .....	155
Разрешение прерываний .....	155
Приоритет и порог приоритета .....	156
Входной блок(Gateway).....	156
Подтверждение начала/окончания обработки прерывания .....	156
Подтверждение начала обработки прерывания .....	156
С-функции для работы с PLIC .....	157
Карта регистров.....	157
Interrupt Pending (IP) .....	158
Interrupt Source Type Level / Edge (LE). .....	159
Interrupt Priority (PRIO).....	160
Interrupt Enable (CPU_IE, FPGA_IE).....	160
Threshold (CPU_THRESHOLD, FPGA_THRESHOLD). .....	161
Claim / Complete (CPU_CC, FPGA_CC). .....	161
Machine-Mode Software Interrupt Pending (CPU_MSIP, FPGA_MSIP): .....	161
FPGA .....	162
Функционирование .....	163
Режим загрузки из внутренней памяти.....	163
Режим загрузки из внешней памяти .....	163
Регистры FPGA.....	164
CSR0 [0x50197FF0] / CSR1 [0x50197FF4].....	164
DMA [0x50197FF8] .....	164
CFG [0x50197FFC].....	165
Пользовательские выводы FPGA .....	165
Использование внешних выводов .....	171





## Процессорное ядро

В микросхеме установлено 32-битное процессорное ядро (CPU) с открытым кодом RISC-V RV32IMC Ibex. Гарантированная максимальная частота работы вычислительного ядра – 30 МГц.

### Рисунок 8. Структура цифрового ядра

Вычислительное ядро поддерживает ряд интерфейсов: SPI (размер слов 8, 16); UART, I2C, GPIO, 1-Wire, MILSTD1553B, SpaceWire, CAN. Каждый интерфейс имеет по 2 независимых контроллера.

Размер ОЗУ варьируется от 32кБайт до 64кБайт в зависимости от конфигурации. 64кБайт ОЗУ доступно в конфигурации, когда в качестве памяти команд целиком используется ОППЗУ.

Размер ОППЗУ варьируется от 32кБайт до 128кБайт в зависимости от конфигурации. 128кБайт ОППЗУ доступно в случае, если не используется программируемый логический блок (FPGA).

Доступна загрузка программ из внешней памяти.

## Memory Scrubber

Memory Scrubber предназначен для работы с модулями памяти, снабженных механизмом проверки целостности бит. Memory Scrubber выполняет периодическое чтение из заданных диапазонов адресов. В случае обнаружения корректируемой ошибки модули памяти исправляют ее без сигнализации.

В случае обнаружения некорректируемой ошибки происходит сигнализация через шину АНВ.

Memory Scrubber, в свою очередь, сигнализирует некорректируемую ошибку прерыванием или сбросом.

### Основные характеристики

- Master на шине АНВ (32 бит адрес и данные) для доступа к памяти. Только чтение и только 32-битное.
- Slave на шине APB для конфигурирования. Общий синхросигнал и сброс с АНВ.
- 8 программируемых диапазонов - начальный адрес, конечный адрес, разрешение скрабирования.
- Программно задаваемое включение / выключение блока, режим работы: циклический / однократный.
- Маскируемое прерывание / сброс при обнаружении некорректируемой ошибки.
- Сохранение в программно доступный регистр адреса последней ошибки.
- Текущий тестируемый адрес доступен по чтению.
- Программирование интервала запуска заданий и интервала между командами чтения.

## Карта регистров

**Замечание.** Не допускается модифицировать регистры при установленном CTRL.ACT (за исключением CTRL.CE, CTRL.STOP, CTRL.RUN).

Регистр	Адрес	Описание
CTRL	0x00	Регистр управления
BAD_ADDR	0x04	Регистр адреса последней обнаруженной ошибки
CUR_ADDR	0x08	Регистр текущего тестируемого адреса
TIMER	0x0C	Регистр управления интервалами
A0_INIT	0x10	Начальный адрес диапазона 0

Регистр	Адрес	Описание
A0_FIN	0x14	Конечный адрес диапазона 0
A1_INIT	0x18	Начальный адрес диапазона 1
A1_FIN	0x1C	Конечный адрес диапазона 1
A2_INIT	0x20	Начальный адрес диапазона 2
A2_FIN	0x24	Конечный адрес диапазона 2
A3_INIT	0x28	Начальный адрес диапазона 3
A3_FIN	0x2C	Конечный адрес диапазона 3
A4_INIT	0x30	Начальный адрес диапазона 4
A4_FIN	0x34	Конечный адрес диапазона 4
A5_INIT	0x38	Начальный адрес диапазона 5
A5_FIN	0x3C	Конечный адрес диапазона 5
A6_INIT	0x40	Начальный адрес диапазона 6
A6_FIN	0x44	Конечный адрес диапазона 6
A7_INIT	0x48	Начальный адрес диапазона 7
A7_FIN	0x4C	Конечный адрес диапазона 7
SCTRL	0x50	Управление сигнализацией ошибок
ECNT0	0x54	Счетчики ошибок 0-3
ECNT1	0x58	Счетчики ошибок 4-7

## Регистры

## CTRL [0x0]: Регистр управления

Биты	Название	Описание	Доступ	Сброс
31:18	-	Резерв	R	0
17:16	EE	ECC Enable Управляет отключением проверки и генерации проверочного кода для основного и отладочного ОЗУ.  0 – аппаратная генерация проверочного кода отключена; 1 – аппаратная генерация проверочного кода включена;  Генерация ECC может быть выключена, если не требуется в выбранном применении микросхемы с целью снижения времени доступа к памяти. Байтовая запись в память в таком случае не требует последовательности Read-Modify-Write и выполняется за 1 такт.	R/W	0
15	-	Резерв	R	0
14	ACT	Active Признак активности блока: 0 - выключен; 1 - активен. При записи 1 в STOP бит сбрасывается после проверки очередного диапазона.	R	0
13	CE	Clear Error Очистка флагов прерываний и счетчиков ошибок. Сбрасывается аппаратно. Бит сбрасывается автоматически после записи. По чтению всегда равен 0.	R/W/SC	0

Биты	Название	Описание	Доступ	Сброс
12	EV	Error Valid Признак валидности данных в регистре BAD_ADDR. При одновременном сбросе и обнаружении ошибки устанавливается в 1.	R	0
11	-	Резерв	R	0
10	MODE	Режим работы: 0 - однократный; 1 - циклический. В однократном режиме контроллер проведет скрабирование разрешенных регионов (CTRL.RE) и остановится. В циклическом режиме после проверки всех разрешенных регионов, проверка повторится по истечении паузы TIMER.TI.	R/W	0
9	STOP	Запись 1 останавливает контроллер после проверки очередного диапазона. Бит сбрасывается автоматически после записи. По чтению всегда равен 0.	R/W/SC	0
8	RUN	Запись 1 запускает контроллер. Запись 1 не имеет действия, если контроллер уже активен (CTRL.ACT) или одновременно происходит запись 1 в CTRL.STOP. Бит сбрасывается автоматически после записи. По чтению всегда равен 0.	R/W/SC	0
7:0	RE	Range Enable. Разрешение проверки соответствующего диапазона адресов.	R/W	0

**BAD\_ADDR [0x4]: Регистр адреса обнаруженной ошибки**

Биты	Название	Описание	Доступ	Сброс
31:0	-	Адрес, по которому была обнаружена ошибка.	R	x

**CUR\_ADDR [0x8]: Регистр текущего тестируемого адреса**

Биты	Название	Описание	Доступ	Сброс
31:0	-	Текущий тестируемый адрес (при CTRL.ACT = 1), или последний протестированный адрес (при CTRL.ACT = 0).	R	x

**TIMER [0xC]: Регистр управления интервалами**

Биты	Название	Описание	Доступ	Сброс
31:24	RI	Read Interval Интервал ожидания между последовательными командами чтения. Служит для балансирования нагрузки на систему.	R/W	0
23:0	TI	Task Interval Интервал между последовательными запусками скрабирования. Каждый запуск состоит из последовательной проверки всех интервалов.	R/W	0

**Ax\_INIT [0x10...]: Начальный адрес диапазона x**

Биты	Название	Описание	Доступ	Сброс
31:2	-	Начальный адрес диапазона тестируемой памяти.	R/W	x
1:0	-	Резерв	R	x

**Ax\_FIN [0x14...]: Конечный адрес диапазона x**

Биты	Название	Описание	Доступ	Сброс
31:2	-	Следующий за тестируемым диапазоном адрес. Для диапазона, завершающегося адресом 0xFFFFFFFF следует задать равным 0x0. Должен быть больше, чем соответствующий ему начальный адрес, если диапазон разрешен для проверки.	R/W	x
1:0	-	Резерв.	R	x

**SCTRL [0x50]: Управление сигнализацией ошибок**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв.	R	0
7:0	SM	Signal Mode. Выбор режима сигнализации ошибки для каждого диапазона. 0 - прерывание; 1 - сброс. Для диапазонов, содержащих код программ, может быть правильнее проводить сброс системы при обнаружении некорректируемой ошибки. Для диапазонов данных может быть достаточно прерывания.	R/W	0

**ECNT0 [0x54]: Регистры счетчиков ошибок 0-3.**

Регистр ECNT1 аналогичен для ошибок диапазонов 4-7.

Биты	Название	Описание	Доступ	Сброс
31:24	ECNT3	Аналогично ECNT0	R	0
23:16	ECNT2	Аналогично ECNT0	R	0
15:8	ECNT1	Аналогично ECNT0	R	0
7:0	ECNT0	Error Counter Счетчик количества обнаруженных ошибок. Инкрементируется при обнаружении ошибки до достижения максимального значения 255. Сбрасывается установкой CTRL.CE. При одновременном сбросе и обнаружении ошибки записывается значением 1.	R	0

**Применение**

При использовании ОЗУ с проверочным кодом следует убедиться, что все данные, записанные в ОЗУ записаны после активации режима (CTRL.EE=1). Это может быть сделано несколькими способами:

Активировать CTRL.EE перед запуском основной программы (см. пример в src/cpp/crt0.S).

Перезаписать все используемые программой ячейки памяти после активации CTRL.EE с помощью DWORD/int обращений. Обращения по записи char/short могут привести к ошибкам кода Хэмминга. Не следует читать неинициализированные ячейки памяти до их инициализации кодом Хэмминга.

## Memory Controller (MEMC)

Контроллер памяти позволяет настраивать работу блоков памяти в системе. ROM (ОППЗУ) память в системе является однократно программируемой. По умолчанию (после сброса системы) запись в память заблокирована. Через блок Memory Controller DM или SPW RMAP можно разблокировать запись в ROM память и загрузить программу.

Максимальная частота, на которой может работать память ROM – 12,5 МГц. Если частота АНВ превышает это значение, то необходимо заблаговременно поделить частоту работы ROM, задав коэффициент деления в блоке MEMC.

В Memory Controller задаются параметры диаграммы чтения внешней памяти EXTMEM.

**Внимание:** Memory Controller предназначен для того, чтобы позволить DM или SPW RMAP прошивать память. При попытке процессора в рабочем режиме (BOOT\_MODE == 0) перевести ROM-память в режим программирования, он не сможет более выполнять программу, т.к. чтение ROM-памяти будет заблокировано.

### Управление частотой ROM

Максимальная частота, на которой может работать память ROM - 12.5 МГц. Если частота АНВ превышает это значение, то необходимо заблаговременно поделить частоту работы ROM, задав коэффициент деления в блоке MEMC (регистр **MEMC\_ROM\_CLKDIV**). Частота АНВ будет автоматически поделена в блоке ROM INTERFACE и полученный тактовый сигнал будет выдан на ROM память.

Примечание: **MEMC\_ROM\_CLKDIV** влияет только на ROM в режиме чтения, в режиме программирования деления частоты не происходит.

### Управление EXTMEM

В Memory Controller задаются параметры диаграммы чтения EXTMEM.

### Карта регистров Memory Controller

Смещение от базового адреса блока	Аббревиатура	Название регистра	Описание
Регистры управления ROM			
0x0	MEMC_ROM_KEY	ROM Key Register	В этот регистр нужно записать ключ разблокировки, чтобы перевести ROM в режим программирования
0x4	MEMC_ROM_CTRL	ROM Control Register	Регистр управления ROM
0x8	MEMC_ROM_CLKDIV	ROM Clock Divider Register	Коэффициент деления частоты для ROM памяти
0xC	MEMC_ROM_ST	ROM Status Register	Регистр статуса ROM
Регистры управления EXTMEM			
0x10	MEMC_EXTMEM_CTRL	EXTMEM Control Register	Регистр управления EXTMEM

## Регистры

## ROM Key Register (MEMC\_ROM\_KEY) [0x00].

Биты	Название	Описание	Доступ	Сброс
31:0	KEY	Для перевода ROM в режим программирования необходимо записать в этот регистр значение ключа (0x1234ABCD). После окончания прошивки ROM, запишите в этот регистр любое другое значение, чтобы вывести ROM из режима программирования.	R/W	0

## ROM Control Register (MEMC\_ROM\_CTRL) [0x04].

Биты	Название	Описание	Доступ	Сброс
31:0	PGM_CYCLES	Длительность программирования одной ячейки ROM в циклах частоты АНВ. Именно такое количество тактов будет удерживаться сигнал o_se_n ROM-памяти при программировании одного слова. Все это время будет выставлен флаг BUSY в регистре MEMC_ROM_ST. Значение по-умолчанию - 128 (0x80) тактов.	R/W	0x80

## ROM Clock Divider Register (MEMC\_ROM\_CLKDIV) [0x08].

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв	R	0
2:0	ROM_DIV	Коэффициент деления ROM делителя частоты. Коэффициент деления частоты равен ROM_DIV + 1. Таким образом можно задать коэффициенты деления от 1 (нет деления) до 8.	R/W	0

## ROM Status Register (MEMC\_ROM\_ST) [0x0C].

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3	RD_ERR	Ошибка чтения ROM: • 0 - не зафиксировано ошибки чтения из ROM; • 1 - зафиксирована ошибка чтения из ROM: произошла попытка чтения из ячейки ROM при PGM == 1. Этот бит сбрасывается в 0 при чтении.	R/W	0
2	WR_ERR	Ошибка записи в ROM: • 0 - не зафиксировано ошибки записи в ROM; • 1 - зафиксирована ошибка записи в ROM: произошла попытка записи в ячейку ROM при ROM_BUSY == 1 Этот бит сбрасывается в 0 при чтении.	R/W	0
1	ROM_BUSY	Признак занятости ROM: • 0 - в данный момент не происходит запись в ячейки ROM, можно записывать в ROM (разумеется, запись возможна только если бит PGM выставлен в 1); • 1 - в данный момент происходит запись в одну из ячеек ROM, записывать в ROM запрещено, пока этот бит не вернется в значение 0.	R/W	0

Биты	Название	Описание	Доступ	Сброс
0	PGM	Режим программирования ROM: <ul style="list-style-type: none"> <li>0 - ROM в рабочем режиме, не введен верный ключ в регистр MEMC_ROM_KEY. Запись в ROM заблокирована, попытка записи вызывает ошибку на системной шине.</li> <li>1 - ROM в режиме программирования, введен верный ключ в регистре MEMC_ROM_KEY. Запись в IMEM (ROM) разблокирована. При этом запрещено чтение из ROM, при попытке чтения возникает ошибка на системной шине и выставляется в 1 бит RD_ERR.</li> </ul>	R/W	0

#### External Memory Control Register (MEMC\_EXTMEM\_CTRL) [0x10].

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв	R	0
2	T1	Количество тактов, которое продлится время T1 на диаграмме чтения EXTMEM (см. документацию на EXTMEM): <ul style="list-style-type: none"> <li>0 - 0 тактов;</li> <li>1 - 1 такт.</li> </ul>	R/W	0
1:0	T0	Количество тактов, которое продлится время T0 на диаграмме чтения EXTMEM (см. документацию на EXTMEM): <ul style="list-style-type: none"> <li>00 - 1 такт;</li> <li>01 - 2 такта;</li> <li>10 - 3 такта;</li> <li>11 - 4 такта.</li> </ul>	R/W	0

#### Алгоритм загрузки программы в ROM

Для того, чтобы загрузить программу в ROM, необходимо:

1. Перевести микросхему в рабочий режим (BOOT\_MODE == 0)
2. Подать необходимое для программирования напряжение 9V на соответствующий вывод микросхемы
3. Записать ключ разблокировки (0x1234ABCD) в регистр MEMC\_ROM\_KEY. После этого ROM перейдет в режим программирования и выставляется бит MEMC\_ROM\_ST.PGM.
4. (Опционально) изменить длительность программирования одного слова в регистре MEMC\_ROM\_CTRL.
5. Записать программу в адреса IMEM. Перед каждой записью 32-битного слова в адрес IMEM необходимо проверить, что бит **MEMC\_ROM\_ST.ROM\_BUSY** имеет значение 0.

Если попытаться записать в адрес ROM когда бит **MEMC\_ROM\_ST.PGM** не выставлен в 1, произойдет ошибка доступа на системной шине. Аналогично, если попытаться прочитать адрес ROM, когда бит MEMC\_ROM\_ST.PGM выставлен в 1, произойдет та же самая ошибка.

В режиме отладки (BOOT\_MODE == 1) возможно писать в регистры MEMC\_ROM\_KEY и MEMC\_ROM\_CTRL, но это не будет иметь эффекта. В этом режиме в качестве памяти программ (IMEM) используется DBG\_RAM, запись в DBG\_RAM возможна и без разблокировки в блоке Memory Controller, а также без подачи напряжения 9 В.

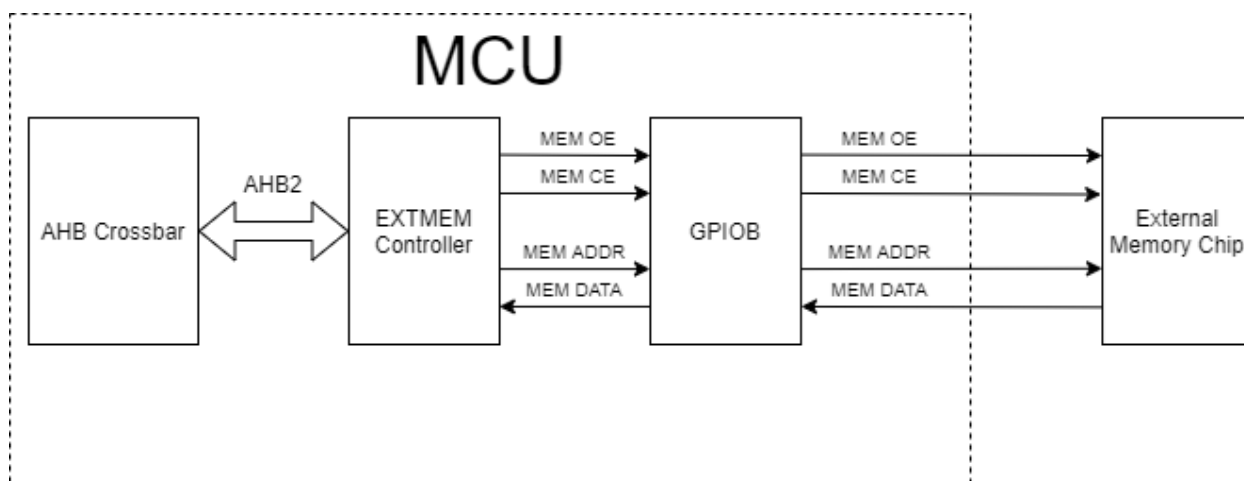


**EXTMEM**

Блок EXTMEM предназначен для чтения данных из внешней памяти 1636PP3У (512 Кбайт) / 1636PP4У (2 мегабайта) или подобной.

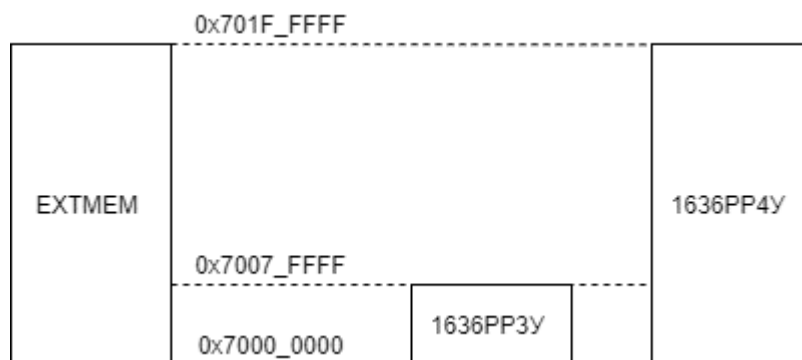
Шина данных внешней памяти восьмибитная, но из EXTMEM можно считывать данные как по 8, так и по 16 или 32 бита. Блок EXTMEM автоматически произведет несколько обращений к внешней памяти за каждое 16/32-битное чтение по АНВ.

Процессор может исполнять программу из EXTMEM, если из IMEM сделать jump на адрес EXTMEM, но скорость выполнения программы может быть значительно ниже, чем при исполнении программы из внутренней памяти. EXTMEM не поддерживает запись во внешнюю память. При попытке записать в любой из адресов EXTMEM возникнет ошибка на системной шине.

**Предупреждение**

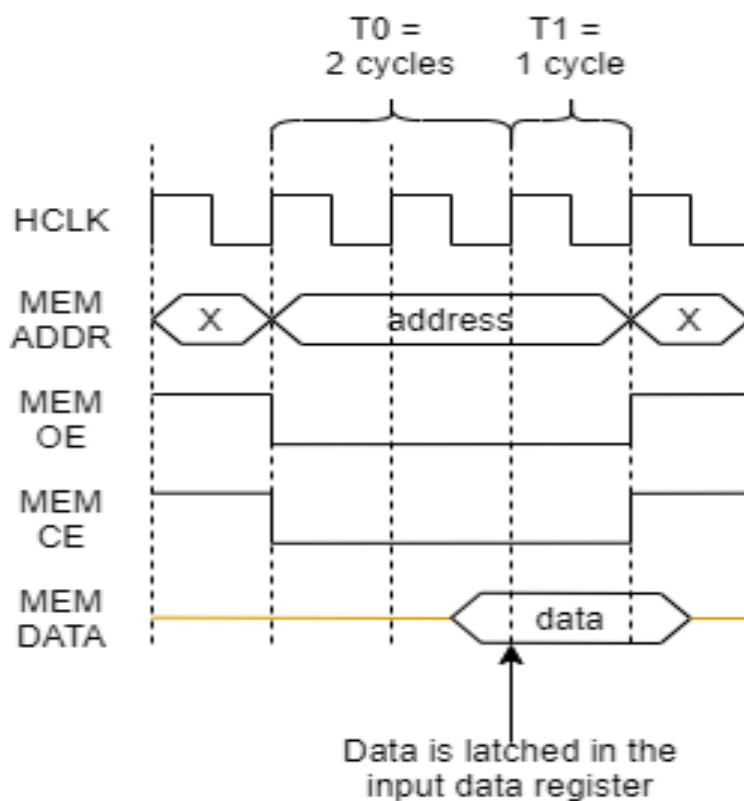
Рекомендуется подключить вывод микросхемы памяти MRST к соответствующему сигналу сброса схемы, в которой применяется данная микросхема памяти, или к одному из свободных выводов GPIO и перед началом работы с памятью подать на него сигнал сброса длительностью 150 мкс (активный уровень сброса - 0). Без предварительной подачи сброса правильная работа микросхемы памяти не гарантируется.

Когда микросхема памяти подключена к соответствующим выводам GPIO и настроены альтернативные функции GPIO, любой из мастеров АНВ в системе может читать из внешней памяти. Адреса внешней памяти маппированы на адреса EXTMEM, как показано на рисунке ниже.



Шина данных внешней памяти восьмибитная, но из EXTMEM можно считывать данные как по 8, так и по 16 или 32 бита. Блок EXTMEM автоматически произведет несколько обращений к внешней памяти за каждое 16/32-битное чтение по АНВ.

При каждом чтении байта блок EXTMEM разворачивает следующую диаграмму на выводах MEM\_ADDR, MEM\_OE, MEM\_CE (T0 = 2 такта, T1 = 1 такт):



Время T0 (1-4 такта АНВ) - это время от момента выставления MEM\_OE и MEM\_CE в 0 до захвата MEM\_DATA во входной регистр EXTMEM.

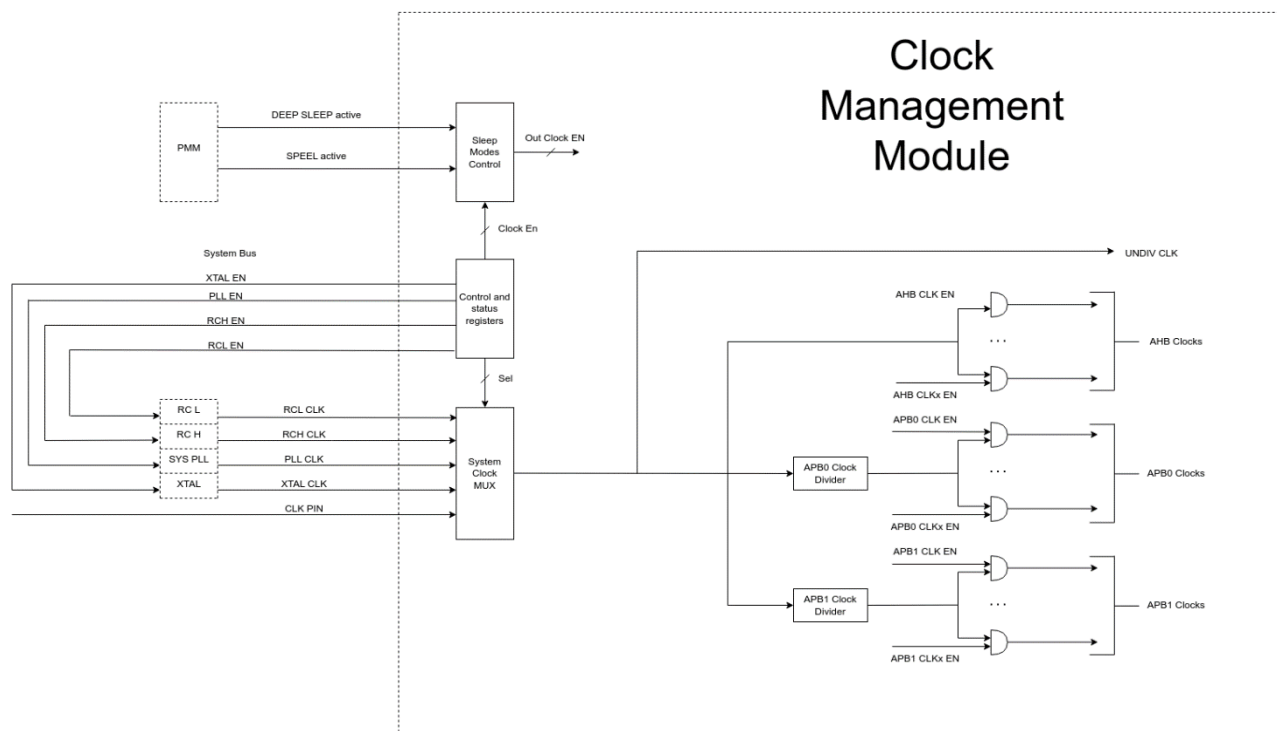
Время T1 (0-1 такта АНВ) - это время от момента захвата MEM\_DATA во входной регистр EXTMEM до момента возвращения MEM\_OE и MEM\_CE в 1.

Времена T0 и T1 следует выбирать исходя из системной частоты (частоты АНВ) и характеристик внешней памяти. Времена T0 и T1 задаются в регистре MEMC\_EXTMEM\_CTRL блока MEMC.

Процессор может исполнять программу из EXTMEM, если из IMEM сделать jump на адрес EXTMEM, но в зависимости от T0 и T1 скорость выполнения программы может быть значительно ниже, чем при исполнении программы из внутренней памяти.

EXTMEM не поддерживает запись во внешнюю память. При попытке записать в любой из адресов EXTMEM возникнет ошибка на системной шине.

## СММ – Блок управления тактовыми сигналами



Модуль СММ (Clock Management Module) осуществляет управление тактовыми сигналами в системе: выбор источника тактовой частоты, деление частоты, отключение тактирования отдельных компонентов системы.

В системе присутствуют пять источников тактовых сигналов – низкочастотный (100 кГц) RC-генератор (RC GEN L), высокочастотный RC-генератор (10 МГц) (RC GEN H), внешний кварцевый резонатор (XTAL), фазовая автоподстройка частоты (SYS PLL), а также внешняя тактовая частота (CLK).

Мультиплексор системной частоты выбирает один из входных тактовых сигналов в качестве системной частоты. Мультиплексор системной частоты является Glitch Free, то есть не создает ложных тактовых сигналов в процессе переключения.

### Источники тактовых сигналов в системе

- Высокочастотный RC генератор (RC H),
- Низкочастотный RC генератор (RC L),
- Внешний кварцевый кристалл (XTAL),
- Внешний КМОП синхросигнал (CLK PIN),
- Генератор с фазовой автоподстройкой частоты (PLL).

Мультиплексор системной частоты (System Clock MUX) выбирает один из входных тактовых сигналов в качестве системной частоты (System Clock). Мультиплексор системной частоты является Glitch Free, то есть не создает ложных тактовых сигналов в процессе переключения (см. «Диаграмма Переключения» ниже).

Выходные тактовые сигналы формируются из System Clock. Они делятся на три группы - тактовые сигналы APB0 и APB1. Частоты можно поделить делителями APB0 / APB1 Clock Divider на 2, 4, 8, 16 или 32.

Каждый тактовый сигнал покидает СММ через ячейку клокгейтинга. Большинство тактовых сигналов (кроме самых критичных для работы системы) можно отключить записью в управляющий регистр. Также некоторые выходные тактовые сигналы автоматически выключаются в режимах сна. Сигналы

разрешения работы (Out clock EN) для всех выходных тактовых сигналов формирует блок Sleep Mode Control. Этот блок получает информацию о активном на данный момент режиме сна от PMM (сигналы SLEEP active и DEEP SLEEP active) и автоматически включает и отключает соответствующие режиму сна тактовые сигналы.

Пользователь взаимодействует с CMM читая и записывая значения в управляющие и статусные регистры (Control and Status Registers) через системную шину (System Bus).

Номер	Название сигнала	Тактируемые блоки	После сброса системы	В режиме SLEEP	В режиме DEEP SLEEP	Может быть отключен записью в CLKEN
AHB						
0	ahb_main_clk	AHB Crossbar, IMEM, DMEM, AHB Bus	Работает	Работает		нет
1	ahb_cpu_clk	CPU		Не работает		
2	ahb_dm_clk	DM		Работает		
3	ahb_spw0_clk	AHB-часть SPW0	Не работает			
4	ahb_spw1_clk	AHB-часть SPW1				
5	ahb_gpioa_clk	GPIOA				
6	ahb_gpiob_clk	GPIOB				
7	ahb_extmem_clk	EXTMEM	Работает			
8	ahb_fpga_clk	FPGA				
9	ahb_milstd0_clk	AHB-часть MILSTD0	Не работает		Работает	Не работает
10	ahb_milstd1_clk	AHB-часть MILSTD1				
11	ahb_scrub_clk	AHB-часть SCRUBBER				
APB0						
0	apb0_main_clk	APB0 Bus	Работает		Не работает	Нет
1	apb0_lp_clk	Частично PMM и STIMER			Работает	
2	apb0_cmm_clk	CMM*			Не работает	
3	apb0_pmm_clk	PMM*	Не работает	Работает	Да	
4	apb0_plic_clk	PLIC				
5	apb0_romc_clk	ROMC				
6	apb0_wdt_clk	WDT				
7	apb0_stimer_clk	STIMER*				
8	apb0_timer0_clk	TIMER 0				
9	apb0_timer1_clk	TIMER 1				
10	apb0_timer2_clk	TIMER 2				

Номер	Название сигнала	Тактируемые блоки	После сброса системы	В режиме SLEEP	В режиме DEEP SLEEP	Может быть отключен записью в CLKEN
11	apb0_uart0_clk	UART 0				
12	apb0_uart1_clk	UART 1				
13	apb0_spi0_clk	SPI 0				
14	apb0_spi1_clk	SPI 1				
15	apb0_i2c0_clk	I2C 0				
16	apb0_i2c1_clk	I2C 1				
17	apb0_spw0_clk	APB-часть SPW0				
18	apb0_spw1_clk	APB-часть SPW1				
19	apb0_milstd0_clk	APB-часть MILSTD0				
20	apb0_milstd1_clk	APB-часть MILSTD1				
21	apb0_can0_clk	CAN0				
22	apb0_can1_clk	CAN1				
23	apb0_owi0_clk	OWI0				
24	apb0_owi1_clk	OWI1				
25	apb0_fpga_clk	APB-часть FPGA	Работает			
26	apb0_scrub_clk	APB-часть SCRUBBER	Не работает			
APB1						
0	apb1_main_clk	APB1 Bus	Работает	Работает	Не работает	Нет
1	apb1_adfe_ctrl_clk	ADFE CTRL	Не Работает	Работает	Не работает	Да
Специальные тактовые сигналы						
-	undiv_clk	Частично PMM	Работает	Работает	Работает	Нет

\* Только управляющие и статусные регистры этих блоков тактируются данной частотой.

Только управляющие и статусные регистры CMM тактируются `arb_cmm_clk`.

Только управляющие и статусные регистры STIMER тактируются `arb_stimer_clk`. Непосредственно таймер тактируется `arb0_lp_clk`, так как он должен работать в режиме DEEP SLEEP.

Только управляющие и статусные регистры PMM тактируются `arb_pmm_clk`. Блок управления режимом сна в PMM тактируется `arb_lp0_clk`, т.к. этот блок должен работать в режиме DEEP\_SLEEP и вывести систему из этого режима. Блок формирования сигналов сброса в PMM (фильтр и таймер) тактируется `undiv_clk`, т.к. во-первых, формирование сброса должно работать в любом режиме сна, а во-вторых формирование сброса не должно зависеть от коэффициента деления системной частоты.

После сброса часть тактовых сигналов отключена. Работает только необходимый минимум тактовых сигналов (Процессор, память, системные шины и CMM) - их достаточно, чтобы программно включить в регистре CMM CLKEN любые другие тактовые сигналы.

Блоки FPGA, SPW0, SPW1, MILSTD0, MILSTD1 тактируются двумя тактовыми сигналами: один из группы APB и один из группы AHB, т.к. эти блоки являются ведомыми на шине APB и ведущими на шине AHB. Для работы с этими блоками нужно включить обе тактовые частоты.

Частоты

Группа	Частота работы
Специальные тактовые сигналы	(RCL или RCH или XTAL или PLL или CLK)
AHB	(RCL или RCH или XTAL или PLL или CLK)
APB0	(RCL или RCH или XTAL или PLL или CLK) ÷ APB0 Clock Divider
APB1	(RCL или RCH или XTAL или PLL или CLK) ÷ APB1 Clock Divider

Сброс

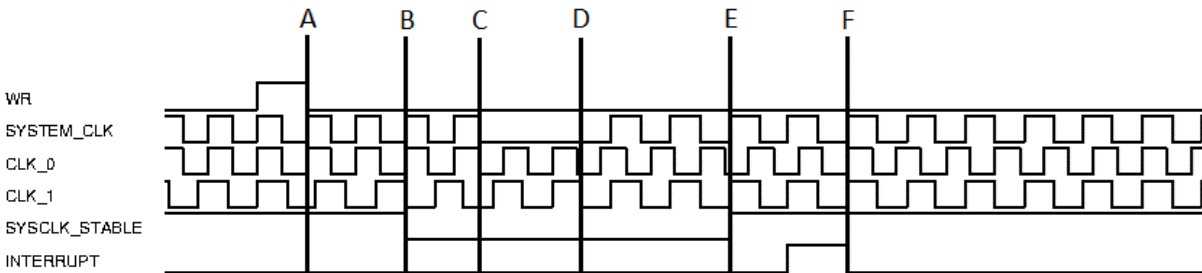
CMM не сбрасывается сбросом WDT и DM. При сбросе от WDT или DM конфигурация CMM не меняется.

Всегда работающие тактовые сигналы

Сигнал `arb_lp_clk` (Low Power Clock) активен даже в режиме DEEP SLEEP. На этой частоте работает STIMER т.к. он должен работать в режиме глубокого сна, а также часть PMM, отвечающая за выход из глубокого сна - блок управления режимом сна.

Сигнал `undiv_clk` (Undivided Clock) не зависит от выбранного коэффициента деления CMM. Им тактируется часть PMM, отвечающая за генерацию сигналов сброса - фильтр и таймер. Таким образом, система обрабатывает внешний сброс вне зависимости от выбранного коэффициента деления CMM.

Диаграмма переключения



В момент времени А происходит запись в регистр **SYSCLKSEL** (сигнал WR), который инициирует переключение системной частоты (SYSTEM\_CLK) с источника CLK\_0 на источник CLK\_1. Через два такта текущей системной частоты (A-B) новое значение SEL достигает мультиплексора системной частоты и бит **ST.AHBCLK\_STABLE** сбрасывается в 0. Еще 1,5 такта сигнал SEL проходит через синхронизатор на частоте CLK\_0 (B-C). В момент С системная частота останавливается. После этого сигнал разрешения переключения синхронизируется на частоту CLK\_1, что занимает от 1,5-2,5 тактов CLK\_1 (C-D) в зависимости от соотношения частот и фаз CLK\_0 и CLK\_1. В момент D переключение фактически завершено. Еще 2,5 такта CLK\_1 сигнал завершения переключения проходит через синхронизатор (D-E). В момент Е выставляется в 1 **ST.SYSCLK\_STABLE**. Еще через такт будет сформировано прерывание (сигнал INTERRUPT) и через еще один такт (в момент F) оно будет зафиксировано контроллером прерываний.

Итого процесс переключения частот занимает:

3.5 тактов CLK\_0

6.5 - 7.5 тактов CLK\_1

от момента записи в регистр до возникновения сигнала прерывания.

Причина того, что некоторые времена на диаграмме занимают не целое количество тактов в том, что в мультиплексоре используются как триггеры, работающие по фронту, так и триггеры, работающие по спаду тактовых частот.

#### Карта регистров CMM

Отн. адрес	Аббревиатура	Название	Доступ	Описание
0x00	ST	Status Register	RO	Информация о текущем состоянии модуля
0x04	MSK	Interrupt Mask Register	RW	Маска, регулирующая формирование прерываний
0x08	CLKSRCEN	Clock Source Enable	RW	Включение/выключение источников тактовых сигналов
0x0C	SYSCLKSEL	System Clock Select Register	RW	Выбор источника системной частоты
<b>AHB</b>				
—	—	—	—	—
0x14	AHBCLKEN	AHB Clock Enable Register	RW	Включение/выключение тактовых сигналов AHB
<b>APB0</b>				
0x18	APB0CLKDIV	APB0 Clock Divider Register	RW	Коэффициент деления частоты APB0
0x1C	APB0CLKEN	APB0 Clock Enable Register	RW	Включение/выключение тактовых сигналов APB0
<b>APB1</b>				
0x20	APB1CLKDIV	APB1 Clock Divider Register	RW	Коэффициент деления частоты APB1
0x24	APB1CLKEN	APB1 Clock Enable Register	RW	Включение/выключение тактовых сигналов APB1
<b>PLL</b>				

Отн. адрес	Аббревиатура	Название	Доступ	Описание
0x28	PLLCTRL_MILSTD	MILSTD PLL Control Register	RW	Управление PLL MILSTD
0x2C	PLLCTRL_SPW	SPW PLL Control Register	RW	Управление PLL SPW
0x30	PLLCTRL_SYS	System PLL Control Register	RW	Управление системной PLL

## Регистры

## Status Register (ST) [0x00].

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв.	R	0
2	APB1CLK_STABLE	частота APB1 стабильна: • 0 - идет процесс изменения коэффициента деления частоты для APB1, инициированный записью в APB1CLKDIV; • 1 - процесс переключения не идет, частота APB1 стабильна.	R	1
1	APB0CLK_STABLE	частота APB0 стабильна: • 0 - идет процесс изменения коэффициента деления частоты для APB0, инициированный записью в APB0CLKDIV; • 1 - процесс переключения не идет, частота APB0 стабильна.	R	1
0	-	-	-	-

## Interrupt Mask Register (MSK) [0x04].

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв.	R	0
2	APB1CLK_STABLE	разрешение прерывания при переходе ST.APB1CLK_STABLE из 0 в 1: • 0 - прерывание запрещено; • 1 - прерывание разрешено.	R/W	0
1	APB0CLK_STABLE	разрешение прерывания при переходе ST.APB0CLK_STABLE из 0 в 1: • 0 - прерывание запрещено; • 1 - прерывание разрешено.	R/W	0
0	-	-	-	-

## Clock Source Enable (CLKSRCEN) [0x08].

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв.	R	0
3	XTAL_EN	• 0 - низкочастотный RC-генератор выключен; • 1 - низкочастотный RC-генератор включен.	R/W	0
2	-	Резерв.	R	0
1	RCH_EN	• 0 - высокочастотный RC-генератор выключен; • 1 - высокочастотный RC-генератор включен.	R/W	0
0	RCL_EN	• 0 - низкочастотный RC-генератор выключен;	R/W	1



Биты	Название	Описание	Доступ	Сброс
		• 1 - низкочастотный RC-генератор включен.		

**System Clock Select Register (SYSCLKSEL) [0x0C].**

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв.	R	0
2:0	SRC_NUM	Source Number. номер источника тактового сигнала, который выбран в качестве системной частоты (после сброса выбран RCL): <ul style="list-style-type: none"> <li>• 0 - выбран RCL;</li> <li>• 1 - выбран RCH;</li> <li>• 2 - выбран CLK;</li> <li>• 3 - выбран XTAL;</li> <li>• 4 - выбран PLL;</li> <li>• 5...7 - запрещенные значения.</li> </ul>	R/W	0

Запись в этот регистр запускает процесс переключения источника тактового сигнала. При этом бит **ST.AHBCLK\_STABLE** принимает значение 0. Дождитесь выставления этого бита в 1 (окончания процесса переключения), прежде чем писать в этот регистр снова.

**AHB Clock Enable Register (AHBCLKEN) [0x14]**

Каждый бит в этом регистре соответствует выходному тактовому сигналу

Биты	Название	Описание	Доступ	Сброс
31:12	-	Резерв.	R	0
11	SCRUB	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	0
10	MILSTD1	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	0
9	MILSTD0	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	0
8	FPGA	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	0
7	EXTMEM	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	0
6	GPIOB	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	0
5	GPIOA	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	0
4	SPW1	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	1
3	SPW0	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	1
2	DM	• 0 - тактовый сигнал выключен; • 1 - тактовый сигнал включен.	R/W	1
1:0	-	Резерв.	R	3

Пользователю следует выключать тактовые сигналы не задействованных в данный момент блоков, чтобы уменьшить энергопотребление.

**APB0 Clock Divider Register (APB0CLKDIV) [0x18].**

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв.	R	0
2:0	APB0_DIV	значение определяет коэффициент деления APB0 делителя частоты: <ul style="list-style-type: none"> <li>• 000 - деления не происходит;</li> <li>• 001 - деление на 2;</li> <li>• 010 - деление на 4;</li> <li>• 011 - деление на 8;</li> <li>• 100 - деление на 16;</li> <li>• 101 - деление на 32;</li> <li>• 11x - деления не происходит.</li> </ul>	R/W	0

Запись в этот регистр запускает процесс переключения коэффициента деления. При этом бит **ST.APB0CLK\_STABLE** принимает значение 0. Дождитесь выставления этого бита в 1 (окончания процесса переключения), прежде чем писать в этот регистр снова.

**APB0 Clock Enable Register (APB0CLKEN) [0x1C]**

Каждый бит в этом регистре соответствует выходному тактовому сигналу:

0 - тактовый сигнал выключен; 1 - тактовый сигнал включен.

Биты	Название	Описание	Доступ	Сброс
31:27	-	Резерв.	R	0
26	SCRUB		R/W	0
25	FPGA		R/W	1
24	OWI1		R/W	0
23	OWI0		R/W	0
22	CAN1		R/W	0
21	CAN0		R/W	0
20	MILSTD1		R/W	0
19	MILSTD0		R/W	0
18	SPW1		R/W	0
17	SPW0		R/W	0
16	I2C1		R/W	0
15	I2C0		R/W	0
14	SPI1		R/W	0
13	SPI0		R/W	0
12	UART1		R/W	0
11	UART0		R/W	0
10	TIMER2		R/W	0
9	TIMER1		R/W	0
8	TIMER0		R/W	0
7	STIMER		R/W	0
6	WDT		R/W	0
5	ROMC		R/W	0
4	PLIC		R/W	0
3	PMM		R/W	0
2:0	-	Резерв.	R	7

Пользователю следует выключать тактовые сигналы не задействованных в данный момент блоков, чтобы уменьшить энергопотребление.



**APB1 Clock Divider Register (APB1CLKDIV) [0x20]**

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв.	R	0
2:0	APB1_DIV	значение определяет коэффициент деления APB1 делителя частоты: <ul style="list-style-type: none"> <li>• 000 - деления не происходит;</li> <li>• 001 - деление на 2;</li> <li>• 010 - деление на 4;</li> <li>• 011 - деление на 8;</li> <li>• 100 - деление на 16;</li> <li>• 101 - деление на 32;</li> <li>• 11x - деления не происходит.</li> </ul>	R/W	0

Запись в этот регистр запускает процесс переключения коэффициента деления. При этом бит **ST.APB1CLK\_STABLE** принимает значение 0. Дождитесь выставления этого бита в 1 (окончания процесса переключения), прежде чем писать в этот регистр снова.

**APB1 Clock Enable Register (APB1CLKEN) [0x24]. Управление тактированием периферийных устройств на шине APB1.**

Биты	Название	Описание	Доступ	Сброс
31:2	-	Резерв.	R	0
1	ADFE_CTRL	<ul style="list-style-type: none"> <li>• 0 - тактовый сигнал выключен;</li> <li>• 1 - тактовый сигнал включен.</li> </ul>	R/W	0
0	-	Резерв.	R	1

Пользователю следует выключать тактовые сигналы не задействованных в данный момент блоков, чтобы уменьшить энергопотребление.

**MILSTD PLL Control Register (PLLCTRL\_MILSTD) [0x28].**

Биты	Название	Описание	Доступ	Сброс
31:7	-	Резерв.	R	0
6:5	SRC_SEL	значение определяет опорную частоту PLL: <ul style="list-style-type: none"> <li>• 0 - вывод CLK;</li> <li>• 1 - вывод OSCI;</li> <li>• 2 - вывод высокочастотного генератора.</li> </ul>	R/W	0
4	EN	<ul style="list-style-type: none"> <li>• 1 - PLL включена;</li> <li>• 0 - PLL выключена.</li> </ul>	R/W	0
3:0	FREQ_SEL	коэффициент умножения опорной частоты PLL: <ul style="list-style-type: none"> <li>• 0 - без умножения;</li> <li>• 1 - умножить на 2;</li> <li>• 2 - умножить на 4;</li> <li>• 3 - умножить на 8;</li> <li>• 4 - умножить на 9;</li> <li>• 5 - умножить на 10;</li> <li>• 6 - умножить на 12;</li> <li>• 7 - умножить на 15;</li> <li>• 8 - умножить на 16;</li> <li>• 9 - умножить на 18;</li> <li>• 10 - умножить на 20.</li> </ul> Остальные значения зарезервированы.	R/W	0

## SPW PLL Control Register (PLLCTRL\_SPW) [0x2C].

Биты	Название	Описание	Доступ	Сброс
31:7	-	Резерв.	R	0
6:5	SRC_SEL	значение определяет опорную частоту PLL: • 0 - вывод CLK; • 1 - вывод OSCI; • 2 - вывод высокочастотного генератора.	R/W	0
4	EN	• 1 - PLL включена; • 0 - PLL выключена.	R/W	0
3:0	FREQ_SEL	коэффициент умножения опорной частоты PLL: • 0 - без умножения; • 1 - умножить на 2; • 2 - умножить на 4; • 3 - умножить на 8; • 4 - умножить на 9; • 5 - умножить на 10; • 6 - умножить на 12; • 7 - умножить на 15; • 8 - умножить на 16; • 9 - умножить на 18; • 10 - умножить на 20. Остальные значения зарезервированы.	R/W	0

## System PLL Control Register (PLLCTRL\_SYS) [0x30].

Биты	Название	Описание	Доступ	Сброс		
31:7	-	Резерв.	R	0		
6:5	SRC_SEL	Значение определяет опорную частоту PLL: <ul style="list-style-type: none"><li>0 - вывод CLK;</li><li>1 - вывод OSCI;</li><li>2 - вывод высокочастотного генератора.</li></ul>	R/W	0		
4	EN	<ul style="list-style-type: none"><li>1 - PLL включена;</li><li>0 - PLL выключена.</li></ul>	R/W	0		
3:0	FREQ_SEL	Коэффициент умножения опорной частоты PLL:			R/W	0
		FREQ_SEL	Коэффициент	Выходная частота (для опорной частоты 10 МГц)		
		0	1/4	2,5		
		1	1/2	5		
		2	3/4	7,5		
		3	1	10		
		4	5/4	12,5		
		5	3/2	15		
		6	9/5	18		
		7	2	20		
		8	9/4	22,5		
		9	5/2	25		
		10	8/3	26,7		
		11	3	30		
		12	10/3	33,3		
13	15/4	37,5				

Биты	Название	Описание			Доступ	Сброс
		14	4	40		
		15	9/2	45		

### Порты ввода/вывода GPIO

GPIO – это периферийное устройство, которое позволяет управлять выводами общего назначения микросхемы. Каждый вывод можно подключить к одной из альтернативных функций вывода (к другому периферийному устройству внутри микросхемы), либо управлять выводом напрямую, записывая в регистры блока GPIO.

Количество выводов общего назначения в системе зависит от вида корпуса.

**В 208-выводном корпусе 56 выводов общего назначения: 24 в GPIOA и 32 в GPIOB**

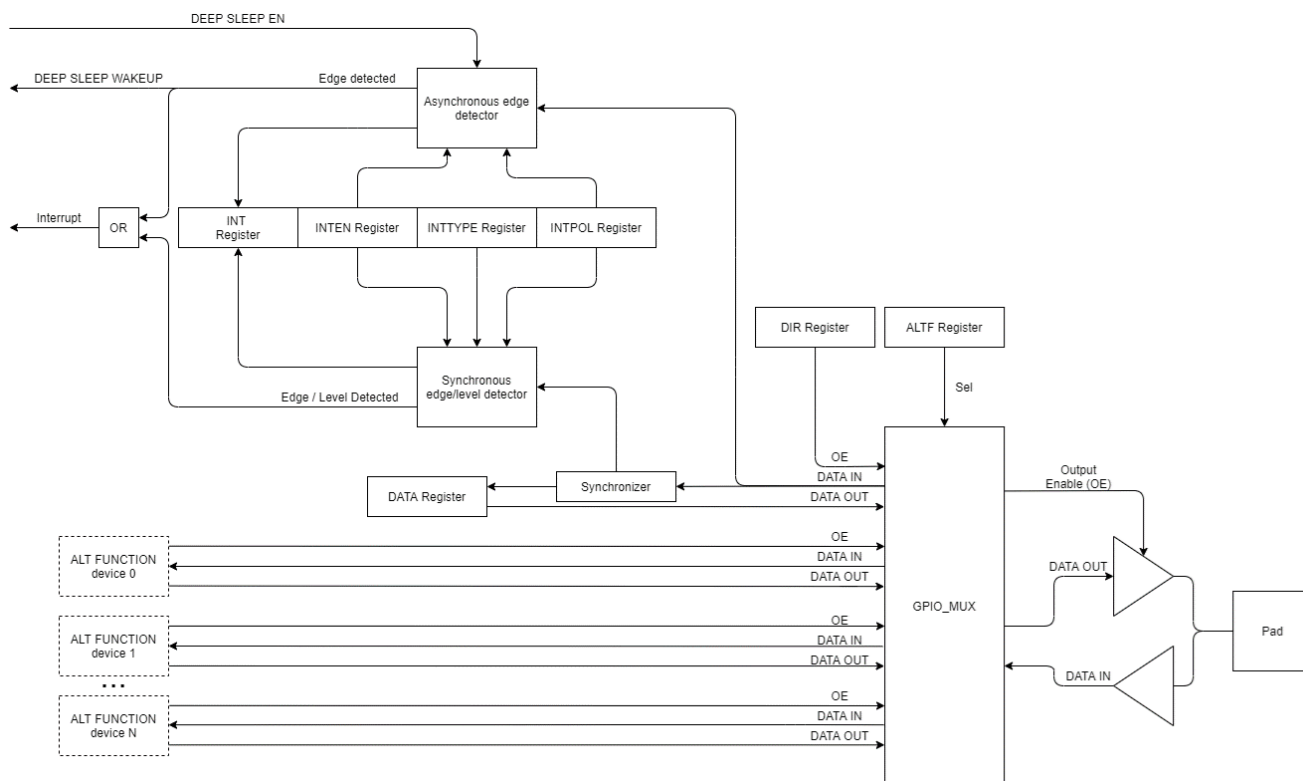
**В 256-выводном корпусе 64 выводов общего назначения: 32 в GPIOA и 32 в GPIOB.**

### Основные возможности GPIO

- Мультиплексор для каждого вывода микросхемы позволяет либо соединить его с одним из периферийных устройств (использовать альтернативную функцию порта), либо задавать и считывать значение вывода напрямую через регистр.
- Если вывод используется как вывод общего назначения, то блок GPIO позволяет настроить его на вход или на выход.
- Блок GPIO может сформировать прерывание при определенном уровне или изменении уровня на порту микроконтроллера.
- Блок GPIO может зафиксировать фронт сигнала на выводе микроконтроллера даже когда система находится в режиме глубокого сна (с помощью асинхронного детектора фронта) и вывести систему из режима сна.

### Структурная схема

На рисунке ниже показана структура GPIO для одного из выводов микроконтроллера.



GPIO\_MUX соединяет PAD (вывод микроконтроллера) либо с регистров GPIO\_DATA, либо с одной из альтернативных функций (ALT FUNCTION device) этого вывода. GPIO\_MUX управляется регистрами ALT\_F. Если вывод соединен с портом процессора, то направление (вход/выход) определяется регистром DIR. Если вывод соединен с альтернативной функцией, то направление (вход/выход) определяется этой альтернативной функцией (периферийным устройством).

В блоке GPIO присутствуют два детектора, способных формировать прерывания - синхронный (Synchronous edge/level detector) и асинхронный (Asynchronous edge detector). Работа детекторов управляется регистрами GPIO\_INTEN, GPIO\_INTPOL и GPIO\_INTTYPE. Статус прерываний сохраняется в регистре GPIO\_INT.

## Прерывания

GPIO поддерживает два режима регистрации событий - синхронный и асинхронный. Синхронный детектор работает в рабочем режиме и в режиме сна «Сон процессора», но не работает в режиме «Глубокий сон». Асинхронный детектор, наоборот, работает только в режиме «Глубокий сон» и предназначен для вывода системы из глубокого сна по внешнему сигналу.

Прерывание для каждого из выводов разрешается записью в регистры GPIO\_INTEN\_SET / GPIO\_INTEN\_CLR.

Синхронный детектор может сформировать прерывание как по фронту сигнала, так и по уровню, выбор типа прерывания осуществляется записью в регистр GPIO\_INTTYPE\_SET / GPIO\_INTTYPE\_CLR. Регистры GPIO\_INTPOL\_SET / GPIO\_INTPOL\_CLR определяют, какой уровень (низкий/высокий) или какой фронт (возрастающий/спадающий) вызовет прерывание.

Асинхронный детектор может сформировать прерывание только по фронту сигнала. При переходе в режим глубокого сна (где работает асинхронный детектор) значение в регистре GPIO\_INTTYPE игнорируется, прерывание срабатывает по фронту сигнала (возрастающему или спадающему, в зависимости от GPIO\_INTPOL).

Какой именно вывод вызвал прерывание можно выяснить, прочитав регистр статуса прерываний GPIO\_INT. Соответствующий бит в регистре GPIO\_INT выставляется в 1 только если прерывание по

этому выводу разрешено в регистре GPIO\_INTEN.

Асинхронный детектор не использует системную частоту, поэтому может работать в режиме глубокого сна микроконтроллера. Асинхронный детектор при срабатывании, помимо прерывания, формирует сигнал WAKEUP, который приводит к выводу системы из режима глубокого сна. Таким образом блок GPIO можно использовать, чтобы выйти из режима глубокого сна по внешнему событию.

Асинхронный детектор фронта работает с несинхронизированным на системную частоту входным сигналом, поэтому даже короткий глитч входного сигнала будет гарантированно зарегистрирован как фронт.



## Список альтернативных функций GPIO

В столбце GPIOA\_EXT содержатся функции, значение которых определяется не регистрами GPIO\_ALTF, а другими выводами микросхемы.

№	GPIOA_EXT	GPIOA_ALTF0	GPIOA_ALTF1	GPIOA_ALTF2	GPIOA_ALTF3	GPIOB_ALTF0	GPIOB_ALTF1	GPIOB_ALTF2
0	-	SPI0_MOSI	UART0_TX	Timer0 Ext Event	M0A_RXP	EXTMEM D0	M0A_RXP	UART0_TX
1	-	SPI0_MISO	UART0_RX	Timer1 Ext Event	M0A_RXM	EXTMEM D1	M0A_RXM	UART0_RX
2	-	SPI0_CS	UART0_RTS	Timer2 Ext Event	M0A_RXE	EXTMEM D2	M0A_RXE	UART0_RTS
3	-	SPI0_SCK	UART0_CTS	-	M0A_TXP	EXTMEM D3	M0A_TXP	UART0_CTS
4	-	SPI1_MOSI	UART1_TX	OWI0 Data	M0A_TXM	EXTMEM D4	M0A_TXM	UART1_TX
5	-	SPI1_MISO	UART1_RX	OWI0 STP	M0A_TXI	EXTMEM D5	M0A_TXI	UART1_RX
6	-	SPI1_CS	UART1_RTS	OWI1 Data	M0B_RXP	EXTMEM D6	M0B_RXP	UART1_RTS
7	-	SPI1_SCK	UART1_CTS	OWI1 STP	M0B_RXM	EXTMEM D7	M0B_RXM	UART1_CTS
8	-	UART0_TX	CAN0_TX	-	M0B_RXE	EXTMEM CS	M0B_RXE	SPI0_MOSI
9	-	UART0_RX	CAN0_RX	-	M0B_TXP	EXTMEM OE	M0B_TXP	SPI0_MISO
10	-	UART0_RTS	CAN1_TX	-	M0B_TXM	EXTMEM A0	M0B_TXM	SPI0_CS
11	-	UART0_CTS	CAN1_RX	-	M0B_TXI	EXTMEM A1	M0B_TXI	SPI0_SCK
12	-	UART1_TX	OWI0 Data	-	M1A_RXP	EXTMEM A2	M1A_RXP	SPI1_MOSI
13	-	UART1_RX	OWI0 STP	-	M1A_RXM	EXTMEM A3	M1A_RXM	SPI1_MISO
14	-	UART1_RTS	OWI1 Data	-	M1A_RXE	EXTMEM A4	M1A_RXE	SPI1_CS
15	BS_SRC при BS_ENA=1	UART1_CTS	OWI1 STP	-	M1A_TXP	EXTMEM A5	M1A_TXP	SPI1_SCK
16	-	I2C0_SCL	FPGA0	CAN0_TX	M1A_TXM	EXTMEM A6	M1A_TXM	I2C0_SCL
17	-	I2C0_SDA	FPGA1	CAN0_RX	M1A_TXI	EXTMEM A7	M1A_TXI	I2C0_SDA
18	-	I2C1_SCL	FPGA2	CAN1_TX	M1B_RXP	EXTMEM A8	M1B_RXP	I2C1_SCL
19	-	I2C1_SDA	FPGA3	CAN1_RX	M1B_RXM	EXTMEM A9	M1B_RXM	I2C1_SDA
20	TCK при JTAG_EN=1	FPGA0	FPGA4	I2C0_SCL	M1B_RXE	EXTMEM A10	M1B_RXE	CAN0_TX
21	TMS при JTAG_EN=1	FPGA1	FPGA5	I2C0_SDA	M1B_TXP	EXTMEM A11	M1B_TXP	CAN0_RX
22	TDI при JTAG_EN=1	FPGA2	FPGA6	I2C1_SCL	M1B_TXM	EXTMEM A12	M1B_TXM	CAN1_TX
23	TDO при JTAG_EN=1	FPGA3	FPGA7	I2C1_SDA	M1B_TXI	EXTMEM A13	M1B_TXI	CAN1_RX
24	-	Timer0 Ext Event	SPI0_MOSI	FPGA0	CAN0_TX	EXTMEM A14	FPGA8	Timer0 Ext Event
25	-	Timer1 Ext Event	SPI0_MISO	FPGA1	CAN0_RX	EXTMEM A15	FPGA9	Timer1 Ext Event
26	-	Timer2 Ext Event	SPI0_CS	FPGA2	CAN1_TX	EXTMEM A16	FPGA10	Timer2 Ext Event
27	-	OWI0 Data	SPI0_SCK	FPGA3	CAN1_RX	EXTMEM A17	FPGA11	OWI0 Data
28	-	OWI0 STP	SPI1_MOSI	FPGA4	UART0_TX	EXTMEM A18	FPGA12	OWI0 STP
29	-	OWI1 Data	SPI1_MISO	FPGA5	UART0_RX	EXTMEM A19	FPGA13	OWI1 Data
30	-	OWI1 STP	SPI1_CS	FPGA6	UART0_RTS	EXTMEM A20	FPGA14	OWI1 STP
31	-	-	SPI1_SCK	FPGA7	UART0_CTS	-	FPGA15	-

Разряды GPIOA [31:24] отсутствуют в 208-выводном корпусе.

Какая именно альтернативная функция выбрана для порта GPIO определяется значением регистров GPIO\_ALTF0, GPIO\_ALTF1, GPIO\_ALTF2 (регистр GPIO\_ALTF3 есть только у GPIOA т.к. у GPIOA больше альтернативных функций) следующим образом:

Бит X регистра ALTF2 (только для GPIOA)	Бит X регистра ALTF1	Бит X регистра ALTF0	Вывод GPIO X ...
0	0	0	Используется как вывод общего назначения (выходным буфером управляет регистр GPIO_DATA)
0	0	1	Соединен с альтернативной функцией №0 (АФ0)
0	1	0	Соединен с альтернативной функцией №1 (АФ1)
0	1	1	Соединен с альтернативной функцией №2 (АФ2)
1	0	0	Соединен с альтернативной функцией №3 (АФ3) (только для GPIOA)

## Карта регистров GPIO

Смещение от базового адреса блока	Аббревиатура	Название	Описание
0x00	GPIO_DIR_SET	GPIO Direction Set	Установка режима работы выходного буфера
0x04	GPIO_DIR_CLR	GPIO Direction Clear	
0x08	GPIO_DATA_SET	GPIO Data Set	Установка значений на выводах общего назначения
0x0C	GPIO_DATA_CLR	GPIO Data Clear	
0x10	GPIO_INTEN_SET	GPIO Interrupt Enable Set	Разрешение прерываний
0x14	GPIO_INTEN_CLR	GPIO Interrupt Enable Clear	
0x18	GPIO_INTTYPE_SET	GPIO Interrupt Type Set	Выбор типа прерывания (фронт/уровень)
0x1C	GPIO_INTTYPE_CLR	GPIO Interrupt Type Clear	
0x20	GPIO_INTPOL_SET	GPIO Interrupt Polarity Set	Выбор полярности входного сигнала, при которой формируются прерывания
0x24	GPIO_INTPOL_CLR	GPIO Interrupt Polarity Clear	
0x28	GPIO_INT	GPIO Interrupt Status	Статус прерываний
0x2C	GPIO_ALTF0	GPIO Alternative Function Select 0	Выбор альтернативной функции
0x30	GPIO_ALTF1	GPIO Alternative Function Select 1	
0x34	GPIO_ALTF2	GPIO Alternative Function Select 2	Выбор альтернативной функции (Только для GPIOA)
0x38-0x7F	Резерв	Запись в данные регистры не влияет на их содержимое. При чтении значение будет равно 0.	Зарезервированные регистры

После сброса все регистры имеют значение 0x00000000.

## Регистры

**GPIO\_DIR\_SET / GPIO\_DIR\_CLR [0x0/0x4]: парные регистры управления режимом работы выходных буферов порта.**

Каждый бит регистра управляет соответствующим выводом микросхемы.

Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_DIR_SET: <ul style="list-style-type: none"> <li>1 - включить выходной буфер на передачу;</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_DIR_SET: <ul style="list-style-type: none"> <li>1 - выходной буфер включен на передачу;</li> <li>0 - выходной буфер выключен.</li> </ul>	W1S	0

Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_DIR_CLR: <ul style="list-style-type: none"> <li>1 - выключить выходной буфер;</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_DIR_CLR: <ul style="list-style-type: none"> <li>1 - выходной буфер включен на передачу;</li> <li>0 - выходной буфер выключен.</li> </ul>	W1C	0

**GPIO\_DATA\_SET/GPIO\_DATA\_CLR [0x8 / 0xC]: парные регистры данных GPIO.**

Каждый бит регистра соответствует выводу микросхемы. Чтением GPIO\_DATA\_SET / GPIO\_DATA\_CLR можно считать текущее значение на выводе GPIO. Когда данный вывод с помощью GPIO\_DIR настроен как выход, записью в GPIO\_DATA\_SET / GPIO\_DATA\_CLR можно управлять значением на этом выводе.

Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_DATA_SET: <ul style="list-style-type: none"> <li>1 - подать на выходной буфер логическую единицу;</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_DATA_SET: <ul style="list-style-type: none"> <li>1 - на входе порта уровень логической единицы;</li> <li>0 - на входе порта уровень логического нуля.</li> </ul>	W1S	0

Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_DATA_CLR: <ul style="list-style-type: none"> <li>1 - подать на выходной буфер логический ноль;</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_DATA_CLR: <ul style="list-style-type: none"> <li>1 - на входе порта уровень логической единицы;</li> <li>0 - на входе порта уровень логического нуля.</li> </ul>	W1C	0

**GPIO\_INTEN\_SET / GPIO\_INTEN\_CLR [0x10 / 0x14]: парные регистры разрешения генерации прерывания по событиям на входах GPIO.**

Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_INTEN_SET: <ul style="list-style-type: none"> <li>1 - разрешить генерацию прерывания по событиям на данном входе;</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_INTEN_SET: <ul style="list-style-type: none"> <li>1 - разрешена генерация прерывания по событиям на данном входе;</li> <li>0 - запрещена генерация прерывания по событиям на данном входе.</li> </ul>	W1S	0

Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_INTEN_CLR: <ul style="list-style-type: none"> <li>1 - запретить генерацию прерывания по событиям на данном входе;</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_INTEN_CLR: <ul style="list-style-type: none"> <li>1 - разрешена генерация прерывания по событиям на данном входе;</li> <li>0 - запрещена генерация прерывания по событиям на данном входе.</li> </ul>	W1C	0

**GPIO\_INTTYPE\_SET / GPIO\_INTTYPE\_CLR [0x18 / 0x1C]: парные регистры установки типа прерывания (по фронту/уровню) генерируемого GPIO.**

Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_INTTYPE_SET: <ul style="list-style-type: none"> <li>1 - установить генерацию запроса прерывания по фронту;</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_INTTYPE_SET: <ul style="list-style-type: none"> <li>1 - генерация запроса прерывания осуществляется по фронту;</li> <li>0 - генерация запроса прерывания осуществляется по уровню.</li> </ul>	W1S	0
Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_INTTYPE_CLR: <ul style="list-style-type: none"> <li>1 - установить генерацию прерывания по уровню;</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_INTTYPE_CLR: <ul style="list-style-type: none"> <li>1 - генерация запроса прерывания осуществляется по фронту;</li> <li>0 - генерация запроса прерывания осуществляется по уровню.</li> </ul>	W1C	0

**GPIO\_INTPOL\_SET / GPIO\_INTPOL\_CLR [0x20 / 0x24]:** парные регистры установки полярности события GPIO, по которому генерируется прерывание.

Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_INTPOL_SET: <ul style="list-style-type: none"> <li>1 - установить генерацию по положительному фронту / высокому уровню события (зависит от установок GPIO_INTTYPE);</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_INTPOL_SET: <ul style="list-style-type: none"> <li>1 - генерация запроса прерывания осуществляется по положительному фронту / высокому уровню события;</li> <li>0 - генерация запроса прерывания осуществляется по отрицательному фронту / низкому уровню события.</li> </ul>	W1S	0
Биты	Описание	Доступ	Сброс
31:0	Запись в GPIO_INTPOL_CLR: <ul style="list-style-type: none"> <li>1 - установить генерацию по отрицательному фронту / низкому уровню события (зависит от установок GPIO_INTTYPE);</li> <li>0 - не меняет текущую настройку.</li> </ul> Чтение GPIO_INTPOL_CLR: <ul style="list-style-type: none"> <li>1 - генерация запроса прерывания осуществляется по положительному фронту / высокому уровню события;</li> <li>0 - генерация запроса прерывания осуществляется по отрицательному фронту / низкому уровню события.</li> </ul>	W1C	0

**GPIO\_INT [0x28]:** регистр статуса прерываний GPIO.

Биты	Описание	Доступ	Сброс
31:0	Чтение INT регистра GPIO_INT: <ul style="list-style-type: none"> <li>1 - был зафиксирован фронт или уровень (согласно заданным в регистрах GPIO_INTPOL и GPIO_INTTYPE условиям) на данном выводе;</li> <li>0 - фронт или уровень не был зафиксирован.</li> </ul>	RC	0

При чтении регистр сбрасывается в 0. Если во время чтения, пришло какое-либо событие, то оно будет зафиксировано в соответствующем бите регистра. Фиксация события имеет приоритет над очисткой.

GPIO\_ALTF0, GPIO\_ALTF1, GPIO\_ALTF2 [0x2C / 0x30 / 0x34]: регистры GPIO\_ALTF 0 и 1 управляют GPIO\_MUX.

Значения регистров GPIO\_ALTF для каждого из выводов определяют, используется ли вывод как вывод общего назначения, или соединен с одной из альтернативных функций.

Биты				Описание	Доступ	Сброс
31:0						
Бит X регистра ALTF2 (только для GPIOA)	Бит X регистра ALTF1	Бит X регистра ALTF0	Вывод GPIO X			
0	0	0	Используется как вывод общего назначения (выходным буфером управляет регистр GPIO_DATA)			
0	0	1	Соединен с альтернативной функцией №0 (АФ0)		RW	0
0	1	0	Соединен с альтернативной функцией №1 (АФ1)			
0	1	1	Соединен с альтернативной функцией №2 (АФ2)			
1	0	0	Соединен с альтернативной функцией №3 (АФ3) (только для GPIOA)			

## Контроллер CAN интерфейса

CAN является контроллером протокола CAN 2.0A/2.0B. Контроллер поддерживает режимы BasicCAN (PCA82C200 подобный) и PeliCAN. В режиме PeliCAN поддерживаются функции протокола CAN 2.0B. Выбор режима осуществляется в регистре Clock Divider.

Карта регистров и функциональность контроллера отличаются при работе в различных режимах. Далее представлен перечень регистров для каждого из них. Общие регистры (clock divisor и bus timing) описаны после этого. Перечень регистров также отличается при нахождении контроллера в рабочем режиме или режиме сброса. После выхода из сброса контроллер остается в режиме сброса, ожидая завершения конфигурирования. Рабочий режим активируется сбросом бита CR.RR. Для перехода в режим сброса бит CR.RR следует установить.

Адрес	Basic CAN				PeliCAN			
	Рабочий режим		Режим сброса		Рабочий режим		Режим сброса	
	Чтение	Запись	Чтение	Запись	Чтение	Запись	Чтение	Запись
0x00	Control	Control	Control	Control	Mode	Mode	Mode	Mode
0x04	0xFF	Command	0xFF	Command	0x00	Command	0x00	Command
0x08	Status	-	Status	-	Status	-	Status	-
0x0C	Interrupt	-	Interrupt	-	Interrupt	-	Interrupt	-
0x10	0xFF	-	Acceptance code	Acceptance code	Interrupt enable	Interrupt enable	Interrupt enable	Interrupt enable
0x14	0xFF	-	Acceptance mask	Acceptance mask	0x00	-	0x00	-
0x18	0xFF	-	Bus timing 0	Bus timing 0	Bus timing 0	-	Bus timing 0	Bus timing 0
0x1C	0xFF	-	Bus timing 1	Bus timing 1	Bus timing 1	-	Bus timing 1	Bus timing 1
0x20	0x00	-	0x00	-	0x00	-	0x00	-
0x24	0x00	-	0x00	-	0x00	-	0x00	-
0x28	Tx Id1	Tx Id1	0xFF	-	0x00	-	0x00	-
0x2C	Tx Id2	Tx Id2	0xFF	-	Arbitration Lost Capture	-	Arbitration Lost Capture	-
0x30	Tx data byte 1	Tx data byte 1	0xFF	-	Error Code Capture	-	Error Code Capture	-
0x34	Tx data byte 2	Tx data byte 2	0xFF	-	Error warning limit	-	Error warning limit	Error warning limit
0x38	Tx data byte 3	Tx data byte 3	0xFF	-	RX error counter	-	RX error counter	RX error counter
0x3C	Tx data byte 4	Tx data byte 4	0xFF	-	TX error counter	-	TX error counter	TX error counter
0x40	Tx data byte 5	Tx data byte 5	0xFF	-	RX FI SFF	TX FI SFF	Acceptance code 0	Acceptance code 0
0x44	Tx data byte 6	Tx data byte 6	0xFF	-	RX ID 1	TX ID 1	Acceptance code 1	Acceptance code 1
0x48	Tx data byte 7	Tx data byte 7	0xFF	-	RX ID 2	TX ID 2	Acceptance code 2	Acceptance code 2
0x4C	Tx data byte 8	Tx data byte 8	0xFF	-	RX data 1 / RX ID 3	TX data 1 / TX ID 3	Acceptance code 3	Acceptance code 3
0x50	Rx Id1	-	Rx Id1	-	RX data 2 / RX ID 4	Tx data 2 / TX ID 4	Acceptance mask 0	Acceptance mask 0
0x54	Rx Id2	-	Rx Id2	-	RX data 3 /	Tx data 3 /	Acceptance	Acceptance



Адрес	Basic CAN				PeliCAN			
	Рабочий режим		Режим сброса		Рабочий режим		Режим сброса	
	Чтение	Запись	Чтение	Запись	Чтение	Запись	Чтение	Запись
					RX data 1	Tx data 1	mask 1	mask 1
0x58	Rx data byte 1	-	Rx data byte 1	-	RX data 4 / RX data 2	Tx data 4 / Tx data 2	Acceptance mask 2	Acceptance mask 2
0x5C	Rx data byte 2	-	Rx data byte 2	-	RX data 5 / RX data 3	Tx data 5 / Tx data 3	Acceptance mask 3	Acceptance mask 3
0x60	Rx data byte 3	-	Rx data byte 3	-	RX data 6 / RX data 4	TX data 6 / TX data 4	0x00	-
0x64	Rx data byte 4	-	Rx data byte 4	-	RX data 7 / RX data 5	TX data 7 / TX data 5	0x00	-
0x68	Rx data byte 5	-	Rx data byte 5	-	RX data 8 / RX data 6	TX data 8 / TX data 6	0x00	-
0x6C	Rx data byte 6	-	Rx data byte 6	-	FIFO / RX data 7	- / TX data 7	0x00	-
0x70	Rx data byte 7	-	Rx data byte 7	-	FIFO / RX data 8	- / TX data 8	0x00	-
0x74	Rx data byte 8	-	Rx data byte 8	-	Rx message counter	-	Rx message counter	-
0x78	0x00	-	0x00	-	0x00	-	0x00	-
0x7C	Clock divider	Clock divider	Clock divider	Clock divider	Clock divider	Clock divider	Clock divider	Clock divider

## Регистры режима Basic-CAN

**Control (CR) [0x00]:** Регистр управления содержит биты разрешения прерываний, а также биты управления режимом сброса.

Биты	Название	Описание	Доступ	Сброс
31:6	-	Резерв	R	0
5	-	Резерв	R	1
4	OIE	Overrun Interrupt Enable: • 1 - разрешено; • 0 - запрещено.	R/W	0
3	EIE	Error Interrupt Enable: • 1 - разрешено; • 0 - запрещено.	R/W	0
2	TIE	Transmit Interrupt Enable: • 1 - разрешено; • 0 - запрещено.	R/W	0
1	RIE	Receive Interrupt Enable: • 1 - разрешено; • 0 - запрещено.	R/W	0
0	RR	Reset request. Запись 1 сбрасывает все активные транзакции и переводит контроллер в режим сброса. Запись 0 переводит в рабочий режим.	R/W	1

**Command (CMR) [0x04]:** Запись 1 в соответствующий бит регистра инициирует связанное с битом действие.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:4	-	Резерв	R	1
3	CDO	Clear data overrun: запись 1 сбрасывает статус SR.DOS.	R/W	1
2	RRB	Release receive buffer: запись 1 освобождает буфер приемника для нового приема. Запись 1 должна осуществляться после чтения содержимого буфера приемника. В случае наличия нового сообщения в FIFO будет сгенерировано новое прерывание (если разрешено) и SR.RBS будет снова установлен.	R/W	1
1	AT	Abort transmission: запись 1 отменяет еще не стартовавшую передачу.	R/W	1
0	TR	Transmission request: запись 1 инициирует трансфер сообщения из буфера передатчика. Отменить передачу можно только записью 1 в CMR.AT и только если трансфер еще не начался. Если передача уже началась, она не будет остановлена при записи 1 в CMR.AT, но, в случае возникновения ошибки, не будет начата повторная передача.	R/W	1

**Status (SR) [0x08]:** Регистр отражает текущее состояние контроллера и доступен только для чтения.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	BS	Bus status: • 1 - шина неактивна	R	0

Биты	Название	Описание	Доступ	Сброс
6	ES	Error status: по крайней мере один из счетчиков ошибок достиг или превысил пороговое значение (EWL).	R	0
5	TS	Transmit status: • 1 - идет передача сообщения	R	0
4	RS	Receive status: • 1 - идет прием сообщения	R	0
3	TC	Transmission complete: • 1 - последнее сообщение было успешно передано Сбрасывается при запросе на передачу и не устанавливается до успешного окончания передачи.	R	1
2	TBS	Transmit buffer status: • 1 - буфер передатчика доступен для записи Во время исходящей передачи буфер блокируется и бит равен 0.	R	1
1	DOS	Data overrun status: • 1 - сообщение потеряно из-за заполненности FIFO	R	0
0	RBS	Receive buffer status: • 1 - буфер приемника содержит непрочитанные сообщения Сбрасывается записью 1 в CMR.RRB.	R	0

**Interrupt (IR) [0x0C]:** Регистр содержит информацию о причине прерывания. Биты устанавливаются, только если установлены соответствующие им биты CR.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:5	-		R	0b111
4	-		R	0
3	OI	Data overrun interrupt: устанавливается при изменении SR.DOS из 0 в 1. Сбрасывается при чтении.	R	0
2	EI	Error interrupt: устанавливается при детектировании SR.ES или SR.BS. Сбрасывается при чтении.	R	0
1	TI	Transmit interrupt: устанавливается при освобождении буфера передатчика. Сбрасывается при чтении.	R	0
0	RI	Receive interrupt: устанавливается при наличии сообщений в FIFO приемника. Сбрасывается битом CMR.RRB.	R	0

**Буфер передатчика**

Далее приведена структура буфера передатчика. В режиме **BasicCAN** могут передаваться только стандартные сообщения (**EFF** игнорируются).

Адрес	Название	Биты							
		7	6	5	4	3	2	1	0
0x28	ID byte 1	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5	ID.4	ID.3
0x2C	ID byte 2	ID.2	ID.1	ID.0	RTR	DLC.3	DLC.2	DLC.1	DLC.0
0x30	TX data 1	TX byte 1							
0x34	TX data 2	TX byte 2							
0x38	TX data 3	TX byte 3							
0x3C	TX data 4	TX byte 4							
0x40	TX data 5	TX byte 5							
0x44	TX data 6	TX byte 6							
0x48	TX data 7	TX byte 7							
0x4C	TX data 8	TX byte 8							

Если установлен бит **RTR**, то кадр не будет содержать данные. Но **DLC** все равно является частью кадра и должен быть задан.

Не рекомендуется задавать значение **DLC** больше 8. Значение будет округлено до 8 в такой случае.

**Буфер приёмника**

Буфер, расположенный по адресам 0x50-0x74, является видимой частью 64-байтового **FIFO** приемника. Его структура соответствует структуре передающего буфера.

**Фильтрация сообщений**

Сообщения могут быть отфильтрованы на основании значений регистров **Acceptance Mask** и **Acceptance Code**. Старшие 8 бит 11-битного идентификатора сравниваются с Acceptance Code, только для бит, помеченных 0 в **Acceptance Mask**. Сообщение помещается в **FIFO** в случае совпадения.

## Регистры режима PeliCAN

## Mode (MOD) [0x00]:

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3	AFM	Acceptance filter mode: • 1 - режим единичного фильтра • 0 - режим двойного фильтра	R/W	0
2	STM	Self test mode: • 1 - контроллер в режиме самотестирования	R/W	0
1	LOM	Listen only mode: • 1 - режим пассивного приема	R/W	0
0	RM	Reset mode: Запись 1 приводит к отмене всех исходящих сообщений и переходу в режим сброса. Запись 0 переводит в рабочий режим.	R/W	1

**Command (CMR) [0x04]:** Запись 1 в соответствующий бит регистра инициирует связанное с битом действие.

Биты	Название	Описание	Доступ	Сброс
31:5	-	Резерв	R	0
4	SRR	Self reception request: передать и одновременно принять сообщение. В режиме самотестирования возможно протестировать устройство без наличия других устройств на шине. Сообщение будет одновременно принято и передано без генерации других прерываний.	R/W	0
3	CDO	Clear data overrun: запись 1 сбрасывает статус STATUS.DOS.	R/W	0
2	RRB	Release receive buffer: запись 1 освобождает буфер приемника для нового приема. Запись 1 должна осуществляться после чтения содержимого буфера приемника. В случае наличия нового сообщения в FIFO будет сгенерировано новое прерывание (если разрешено) и SR.RBS будет снова установлен.	R/W	0
1	AT	Abort transmission: запись 1 отменяет еще не стартовавшую передачу.	R/W	0
0	TR	Transmission request: запись 1 инициирует трансфер сообщения из буфера передатчика. Отменить передачу можно только записью 1 в CMR.AT и только если трансфер еще не начался. Если передача уже началась, она не будет остановлена при записи 1 в CMR.AT, но, в случае возникновения ошибки, не будет начата повторная передача.	R/W	0

**Status (SR) [0x08]:** регистр отражает текущее состояние контроллера и доступен только для чтения.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	BS	Bus status: • 1 - шина неактивна	R	0
6	ES	Error status: по крайней мере один из счетчиков ошибок достиг или превысил пороговое значение (EWL).	R	0
5	TS	Transmit status: • 1 - идет передача сообщения	R	0

Биты	Название	Описание	Доступ	Сброс
4	RS	Receive status: • 1 - идет прием сообщения	R	0
3	TC	Transmission complete: • 1 - последнее сообщение было успешно передано Сбрасывается при запросе на передачу и не устанавливается до успешного окончания передачи.	R	1
2	TBS	Transmit buffer status: • 1 - буфер передатчика доступен для записи Во время исходящей передачи буфер блокируется и бит равен 0.	R	1
1	DOS	Data overrun status: • 1 - сообщение потеряно из-за заполненности FIFO	R	0
0	RBS	Receive buffer status: • 1 - буфер приемника содержит непрочитанные сообщения Сбрасывается записью 1 в CMR.RRB.	R	0

**Interrupt (IR) [0x0C]:** регистр содержит информацию о причине прерывания. Биты устанавливаются, только если установлены соответствующие им биты CR.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	BEI	Bus error interrupt: устанавливается при обнаружении ошибки на шине. Сбрасывается при чтении.	R	0
6	ALI	Arbitration lost interrupt: устанавливается при потере арбитража. Сбрасывается при чтении.	R	0
5	EPI	Error passive interrupt: устанавливается при переходе из активной к пассивной ошибке. Сбрасывается при чтении.	R	0
4	-	Резерв	R	0
3	DOI	Data overrun interrupt: устанавливается при изменении SR.DOS из 0 в 1. Сбрасывается при чтении.	R	0
2	EWI	Error warning interrupt: устанавливается при детектировании SR.ES или SR.BS. Сбрасывается при чтении.	R	0
1	TI	Transmit interrupt: устанавливается при освобождении буфера передатчика. Сбрасывается при чтении.	R	0
0	RI	Receive interrupt: устанавливается при наличии сообщений в FIFO приемника. Сбрасывается при опустошении FIFO.	R	0

**Interrupt Enable (IER) [0x10]:** управляет выборочным отключением прерываний. Если событие разрешено, устанавливается соответствующий бит IR и генерируется прерывание.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	x
7	BEI	Bus error interrupt: • 1 - разрешено, • 0 - запрещено	R/W	x
6	ALI	Arbitration lost interrupt: • 1 - разрешено, • 0 - запрещено	R/W	x
5	EPI	Error passive interrupt: • 1 - разрешено, • 0 - запрещено	R/W	x
4	-	Резерв	R	x

Биты	Название	Описание	Доступ	Сброс
3	DOI	Data overrun interrupt: • 1 - разрешено, • 0 - запрещено	R/W	x
2	EWI	Error warning interrupt: • 1 - разрешено, • 0 - запрещено	R/W	x
1	TI	Transmit interrupt: • 1 - разрешено, • 0 - запрещено	R/W	x
0	RI	Receive interrupt: • 1 - разрешено, • 0 - запрещено	R/W	x

**Arbitration Lost Capture (ALC) [0x2C]:** содержит позицию бита в потоке, на котором контроллер потерял арбитраж.

Биты	Название	Описание	Доступ	Сброс
31:5	-	Резерв	R	0
4:0	BN	Bit number: бит, на котором потерян арбитраж. Регистр не изменит значение до того, как будет прочитан.	R	0

**Error Code Capture (ECC) [0x30]:** сохраняет коды ошибки, возникшей на шине. Регистр не меняет состояние, до тех пор, пока не будет прочитан.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:6	EC	Error code: <ul style="list-style-type: none"> <li>0 - Bit error;</li> <li>1 - Form error;</li> <li>2 - Stuff error;</li> <li>3 - Other.</li> </ul>	R	0
5	DIR	Direction: <ul style="list-style-type: none"> <li>1 - прием,</li> <li>0 - передача</li> </ul>	R	0
4:0	SEG	Segment: часть кадра, в которой произошла ошибка: <ul style="list-style-type: none"> <li>0x03 - Start of frame;</li> <li>0x02 - ID.28 - ID.21;</li> <li>0x06 - ID.20 - ID.18;</li> <li>0x04 - Bit SRTR;</li> <li>0x05 - Bit IDE;</li> <li>0x07 - ID.17 - ID.13;</li> <li>0x0F - ID.12 - ID.5;</li> <li>0x0E - ID.4 - ID.0;</li> <li>0x0C - Bit RTR;</li> <li>0x0D - Reserved bit 1;</li> <li>0x09 - Reserved bit 0;</li> <li>0x0B - Data length code;</li> <li>0x0A - Data field;</li> <li>0x08 - CRC sequence;</li> <li>0x18 - CRC delimiter;</li> <li>0x19 - Acknowledge slot;</li> <li>0x1B - Acknowledge delimiter;</li> <li>0x1A - End of frame;</li> <li>0x12 - Intermission;</li> <li>0x11 - Active error flag;</li> <li>0x16 - Passive error flag;</li> <li>0x13 - Tolerate dominant bits;</li> <li>0x17 - Error delimiter;</li> <li>0x1C - Overload flag;</li> </ul>	R	0

**Error Warning Limit (EWL) [0x34]:** управляет порогом генерации статуса ошибок. Доступен для записи только в режиме сброса.

**RX error counter [0x38]:** отображает счетчик ошибок приемника. Доступен по записи только в режиме сброса. Событие bus-off сбрасывает счетчик.

**TX error counter [0x3C]:** отображает счетчик ошибок передатчика. Доступен по записи только в режиме сброса.



**TX frame information [0x40]:** имеет идентичный формат для SFF и EFF.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	FF	Формат кадра: • 0 - стандартный (SFF); • 1 - расширенный (EFF).	W	0
6	RTR	Для передачи Remote Transmission Request (RTR) следует аписать 1.	W	0
5:4	-	Резерв	W	0
3:0	DLC	Data Length Code задает значение байт в кадре - от 0 до 8. При записи значения больше 8 будут переданы 8 байт.	W	0?

#### TX ID

Название	7	6	5	4	3	2	1	0
TX identifier 1 (SFF и EFF)	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
TX identifier 2 (SFF)	ID.20	ID.19	ID.18	-	-	-	-	-
TX identifier 2 (EFF)	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13
TX identifier 3 (EFF)	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5
TX identifier 4 (EFF)	ID.4	ID.3	ID.2	ID.1	ID.0	0	0	0

**RX frame information [0x40]:** имеет идентичный формат для SFF и EFF

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	FF	Формат принятого сообщения: • 0 - SFF; • 1 - EFF.	R	0
6	RTR	Равен 1 для кадре Remote Transmission Request (RTR).	R	0
5:4	-	Резерв	R	0
3:0	DLC	Поле Data Length Code (DLC).	R	0?

#### RX ID

Название	7	6	5	4	3	2	1	0
RX identifier 1 (SFF и EFF)	ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21
RX identifier 2 (SFF)	ID.20	ID.19	ID.18	RTR (1 для RTR)	0	0	0	0
RX identifier 2 (EFF)	ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13
RX identifier 3 (EFF)	ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5
RX identifier 4 (EFF)	ID.4	ID.3	ID.2	ID.1	ID.0	RTR	0	0

**RX message counter [0x74]:** содержит количество сообщений, хранящихся в буфере приемника. Старшие 3 бита всегда равны 0.

**Буфер передатчика**

Буфер передатчика доступен только по записи и расположен по адресам 0x40-0x70. Чтение из указанных адресов приводит к течению буфера приемника. Структура буфера зависит от формата кадра (**SFF** или **EFF**).

#	Write (SFF)	Write (EFF)
0x40	TX frame information	TX frame information
0x44	TX ID 1	TX ID 1
0x48	TX ID 2	TX ID 2
0x4C	TX data 1	TX ID 3
0x50	TX data 2	TX ID 4
0x54	TX data 3	Tx data 1
0x58	TX data 4	Tx data 2
0x5C	TX data 5	Tx data 3
0x60	TX data 6	Tx data 4
0x64	TX data 7	Tx data 5
0x68	TX data 8	Tx data 6
0x6C	-	Tx data 7
0x70	-	Tx data 8

**Буфер приёмника**

#	Read (SFF)	Read (EFF)
0x40	RX frame information	RX frame information
0x44	RX ID 1	RX ID 1
0x48	RX ID 2	RX ID 2
0x4C	RX data 1	RX ID 3
0x50	RX data 2	RX ID 4
0x54	RX data 3	Rx data 1
0x58	RX data 4	Rx data 2
0x5C	RX data 5	Rx data 3
0x60	RX data 6	Rx data 4
0x64	RX data 7	Rx data 5
0x68	RX data 8	Rx data 6
0x6C	RX frame information следующего сообщения в FIFO	Rx data 7
0x70	RX ID1 следующего сообщения в FIFO	Rx data 8

**Аппаратные фильтры**

Аппаратные фильтры могут быть использованы для фильтрации сообщений, не соответствующих выбранным критериям. Отфильтрованное сообщение не помещается в FIFO приемника. Режим фильтрации (единичный или двойной) определяется MOD.AFM.

В единичном режиме используется только один 4-байтный фильтр. В двойном режиме используются два фильтра меньшего размера.

Каждый фильтр состоит из двух частей: кода и маски. Код задает шаблон для сравнения, а маска задает незначащие биты. Аппаратные фильтры используют 8 регистров доступных только в режиме сброса.

Адрес	Описание
0x40	Acceptance code 0 (ACR0)
0x44	Acceptance code 1 (ACR1)
0x48	Acceptance code 2 (ACR2)
0x4C	Acceptance code 3 (ACR3)
0x50	Acceptance mask 0 (AMR0)
0x54	Acceptance mask 1 (AMR1)
0x58	Acceptance mask 2 (AMR2)
0x5C	Acceptance mask 3 (AMR3)

#### Единичный фильтр, стандартный кадр

В данном режиме регистры ACR0-ACR3 сравниваются со входящим сообщением следующим образом:

- ACR0.7-0 и ACR1.7-5 сравниваются с ID.28-18
- ACR1.4 сравнивается с RTR
- ACR1.3-0 не используются
- ACR2 и ACR3 сравниваются с байтами данных 1 и 2

Соответствующие биты **AMR** (если установлены) указывают, что результат сравнения не имеет значения.

#### Единичный фильтр, расширенный кадр

В данном режиме регистры **ACR0-ACR3** сравниваются со входящим сообщением следующим образом:

- ACR0.7-0 и ACR1.7-0 сравниваются с ID.28-13
- ACR2.7-0 и ACR3.7-3 сравниваются с ID.12-0
- ACR3.2 сравниваются с RTR
- **ACR3.1-0 не используются**

#### Двойной фильтр, стандартный кадр

В данном режиме регистры **ACR0-ACR3** сравниваются со входящим сообщением следующим образом:

##### Фильтр 1

- ACR0.7-0 и ACR1.7-5 сравниваются с ID.28-18
- ACR1.4 сравнивается с RTR
- ACR1.3-0 сравниваются со старшим нибблом байта данных 1
- ACR3.3-0 сравниваются с младшим нибблом байта данных 1

##### Фильтр 2

- ACR2.7-0 и ACR3.7-5 сравниваются с ID.28-18

**ACR3.4 сравнивается с RTR**

**Двойной фильтр, расширенный кадр**

В данном режиме регистры **ACR0-ACR3** сравниваются со входящим сообщением следующим образом:

**Filter 1**

- ACR0.7-0 и ACR1.7-0 сравниваются с ID.28-13

**Filter 2**

- ACR2.7-0 и ACR3.7-0 сравниваются с ID.28-13

**Общие регистры контроллера CAN**

Контроллер содержит 3 общих регистра, имеющих идентичную адресацию в режимах Basi-CAN и PeliCAN.

**Clock divider (CDR) [0x7F].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	-	Режим CAN: • 1 - PeliCAN; • 0 - Basi-CAN.	R/W	0
6:0	-	Резерв	R/W	0

**Bus timing 0 (BTR0) [0x18].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:6	SJW	Synchronization jump width. Задаёт количество тактов $t_{scl}$ , на которые может быть проведена пересинхронизация за 1 битовый интервал.	R/W	0
5:0	BRP	Baud rate prescaler. Коэффициент деления частоты контроллера для получения частоты приемо-передатчика. Тактовая частота контроллера расчитывается как: $t_{scl} = 2 * t_{clk} * (BRP + 1)$ , где $t_{clk}$ - это частота тактирования контроллера в системе.	R/W	0

**Bus Timing 1 (BTR1) [0x1C].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	SAM	Режим опроса шины: • 1 - шина опрашивается 3 раза; • 0 - шина опрашивается 1 раз.	R/W	0
6:4	TSEG2	Time segment 2	R/W	0
3:0	TSEG1	Time segment 1	R/W	0

Битовый интервал шины определяется частотой приемо-передатчика и временами сегментов по следующим формулам:

$$t_{tseg1} = t_{scl} * (TSEG1 + 1)$$

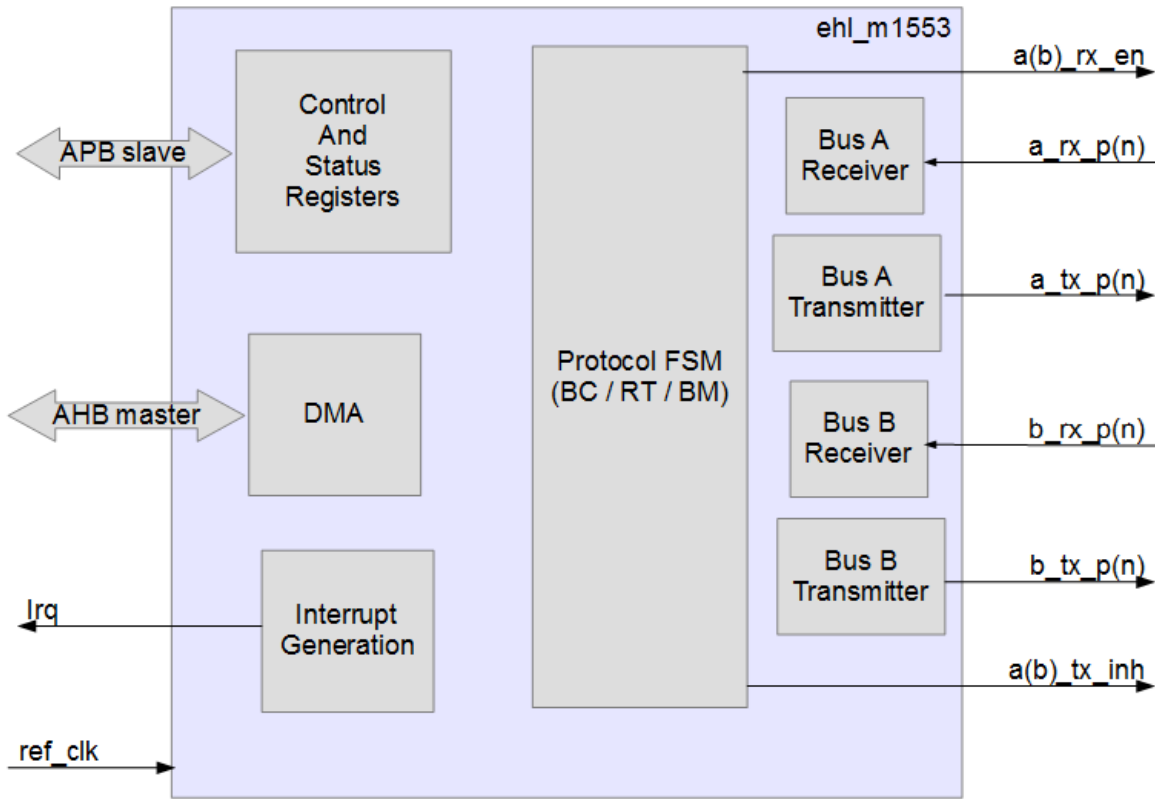
$$t_{tseg2} = t_{scl} * (TSEG2 + 1)$$

$$t_{\text{bit}} = t_{\text{seg1}} + t_{\text{seg2}} + t_{\text{scl}}$$

Точка захвата данных расположена между **TSEG1** и **TSEG2** внутри битового интервала.

Контроллер интерфейса MILSTD1553B (ГОСТ Р 52070-2003)

MILSTD1553B – контроллер интерфейса «Манчестер» Mil-std 1553, ГОСТ Р 52070. Контроллер поддерживает логический уровень интерфейса с выводом сигналов на GPIO в качестве альтернативной функции. Для организации обмена требуется внешняя микросхема приемо-передатчика.



Структурная схема контроллера интерфейса

Перечень выводов

Название	Количество	Тип	Назначение
ref_clk	1	I	Рабочая частота контроллера. 20 МГц.
pclk	1	I	Тактовая частоты шин АHB и APB.
presetn	1	I	Асинхронный сигнал сброса контроллера.
paddr	32	I	Стандартный сигнал шины APB.
pwrite	1	I	Стандартный сигнал шины APB.
psel	1	I	Стандартный сигнал шины APB.
penable	1	I	Стандартный сигнал шины APB.
pwrdata	32	I	Стандартный сигнал шины APB.
pready	1	O	Стандартный сигнал шины APB.
pslverr	1	O	Стандартный сигнал шины APB.
prdata	32	O	Стандартный сигнал шины APB.
haddr	32	O	Стандартный сигнал шины АHB.
htrans	2	O	Стандартный сигнал шины АHB.

Название	Количество	Тип	Назначение
hwrite	1	O	Стандартный сигнал шины АНВ.
hsize	3	O	Стандартный сигнал шины АНВ.
hburst	3	O	Стандартный сигнал шины АНВ.
hprot	4	O	Стандартный сигнал шины АНВ.
hwdata	32	O	Стандартный сигнал шины АНВ.
hrdata	32	I	Стандартный сигнал шины АНВ.
hready	1	I	Стандартный сигнал шины АНВ.
hresp	2	I	Стандартный сигнал шины АНВ.
a_tx_p	1	O	Выход шины А положительной полярности.
a_tx_n	1	O	Выход шины А отрицательной полярности.
a_tx_inh	1	O	Запрет работы передатчика шины А.
a_rx_en	1	O	Разрешение работы приемника шины А.
a_rx_p	1	I	Вход шины А положительной полярности.
a_rx_n	1	I	Вход шины А отрицательной полярности.
b_tx_p	1	O	Выход шины В положительной полярности.
b_tx_n	1	O	Выход шины В отрицательной полярности.
b_tx_inh	1	O	Запрет работы передатчика шины В.
b_rx_en	1	O	Разрешение работы приемника шины В.
b_rx_p	1	I	Вход шины В положительной полярности.
b_rx_n	1	I	Вход шины В отрицательной полярности.
irq	1	O	Прерывание. Активный уровень 1.

### Параметры конфигурации

ehl\_m1553 имеет возможность быть настроенным под задачу пользователя. За настройку отвечают параметры, представленные далее.

Название	Значение по умолчанию	Допустимые значения	Описание
BC_ENA	1	0-1	Использование логики Bus Controller
RT_ENA	1	0-1	Использование логики Remote Terminal
BM_ENA	1	0-1	Использование логики Bus Monitor
SYNC_STAGE	2	0-3	Количество последовательно включенных триггеров в синхронизаторах контроллера.
TECHNOLOGY	0	см. Описание	Выбор технологического процесса: <ul style="list-style-type: none"> <li>0 - RTL;</li> <li>1 - Mikron SOI 018.</li> </ul>

## Функционирование

HL\_M1553 предназначен для работы с шиной Mil-Std 1553B.

**Примечание.** Далее принимается следующая терминология:

Слово - 3-символьная преамбула + 16-битная последовательность + бит четности, переданные (или предназначенные для передачи) по шине.

Сообщение - одно или более слово, переданное по шине, составляющие единую структуру.

Задание - команда передать определенное сообщение по шине, включающая повторную передачу сообщения в случае возникновения ошибок.

Управление режимами работы контроллера, запуск выполнения заданий и работа с прерываниями осуществляется через регистры блока (интерфейс APB). Обмен данными, дескрипторами заданий и результатами их выполнения осуществляется через внешнюю память, доступную блоку через порт ANB.

Контроллер может быть сконфигурирован для работы в режиме Bus Controller или Remote Terminal. Параллельно с ними контроллер может быть включен в режиме Bus Monitor.

Поскольку обмен данными в Mil-Std 1553 происходит с помощью 16-битных слов, то их размещение в памяти соответствует следующему принципу: первое принятое 16-битное слово размещается по меньшему адресу, второе слово имеет адрес +2 относительно первого слова. Такое поведение противоречит [3 4.3.2]: "...the most significant bits shall be transmitted first, followed by the word(s) containing the lesser significant bits...", но может быть исправлено программно.

## Перечень регистров

Регистр	Адрес	Описание
CFG	0x00	Регистр настройки
BC_ADDR	0x04	Регистр настройки адреса задания КШ
CTRL	0x08	Регистр управления
TIME	0x0C	Регистр настройки таймера
LSTAT	0x10	Регистр статуса последнего завершенного сообщения
BM_ADDR_INIT	0x14	Начальный адрес буфера МШ
BM_ADDR_LAST	0x18	Конечный адрес буфера МШ
BM_ADDR_IRQ	0x1C	Адрес генерации прерывания МШ
BM_ADDR_CUR	0x20	Адрес следующей ячейки, записываемой МШ
BM_TIMER	0x24	Регистр таймера МШ
RT_MCC	0x28	Регистр маскирования команд удаленному терминалу
RT_CFG	0x2C	Регистр настройки удаленного терминала
RT_ADDR	0x30	Регистр настройки адреса дескрипторов ОУ
IRQ_ENA	0x34	Регистр разрешения прерываний
IRQ_FLAG	0x38	Регистр флагов прерываний
REVISION	0x3C	Регистр версии контроллера

## Прерывания

ehl\_m1553 позволяет генерировать прерывания для сигнализации различных событий. Для



разрешения генерации прерывания следует установить соответствующее ему разрешение в регистре **IRQ\_ENA**. Тип прерывания зависит от режима работы контроллера.

Для всех режимов работы характерно прерывание от DMA. Для получения заданий, чтения и записи данных, контроллер использует АНВ DMA. В случае получения ошибочного отклика по шине АНВ, поступлении следующей команды с шины, или отсутствия отклика в течение длительного периода времени (120 тактов АНВ), генерируется прерывание

### Контроллер Шины

Контроллер позволяет использовать прерывания для сигнализации завершения выполнения заданий. Выбор задания, по которому генерируется прерывание задается в дескрипторе задания (для **BC Config.IOE**, **Config.IOC**). При выполнении условия генерации прерывания происходит установка флага прерывания в регистре **IRQ\_FLAG** и генерация внешнего прерывания на выводе **irq** контроллера. Для сброса флага прерывания следует записать 1 в соответствующий разряд **IRQ\_FLAG**. Если одновременно со сбросом флага прерывания сгенерируется новое прерывание, оно будет сохранено в регистре флагов прерываний.

### Оконечное Устройство

Контроллер позволяет генерировать прерывания при получении следующих команд: Dynamic Bus Control, Reset Remote Terminal и Synchronize. Прерывание генерируется при завершении команды приема. Обработка флагов прерываний идентична режиму КШ.

### Монитор Шины

Контроллер позволяет генерировать прерывание при совпадении адреса, по которому МШ пишет данные с запрограммированным. Это позволяет отслеживать заполненность буфера и обеспечить непрерывность его использования.

Для Bus Controller прерывание управляется битами **IRQ\_ENA.BCED** и **IRQ\_FLAG.BCID**. Задание, при выполнении которого возникла ошибка DMA прекращается, также как и все последующие задания. Статус дескриптора не обновляется (ввиду потенциальных ошибок DMA).

Bus Controller начинает передачу, убедившись, что линия не занята. Чтобы проверить занятость линии при использовании внешнего трансивера следует установить **RXEN** (иначе принимаются неактивные состояния). Таким образом, любое лишнее слово на шине приемника не будет детектировано как занятая линия. Т.е. проверку следует проводить разрешив прием, подождав некоторое время (необходимое для детектирования активности) и проверив статус. В то же время активность детектируется наличием переключений, и при ключении в случайный момент, принимаемые данные могут быть незадетектированы сразу (но линия занята).

Данный вид проверки остается на откуп пользователю. Приемник будет генерировать **LOOP\_ERROR** в данном случае.

### Регистры

#### Config [0x0]: Регистр настройки

Биты	Название	Описание	Доступ	Сброс
31	BC_ENA	Значение параметра BC_ENA.	R	* x
30	RT_ENA	Значение параметра RT_ENA.	R	* x
29	BM_ENA	Значение параметра BM_ENA.	R	* x
28:5	-	Резерв	R	0

Биты	Название	Описание	Доступ	Сброс
4	DB	Disable Broadcast. Отключение приема широковещательных сообщений для ОУ: <ul style="list-style-type: none"> <li>0 - прием разрешен;</li> <li>1 - прием выключен. Позволяет отключать широковещательные сообщения для применений, в которых они нежелательны [9 50.5.7.1.b].</li> </ul>	R/W	0
3	ECO	Echo Check Off. Отключение режима проверки принятых данных при передаче: <ul style="list-style-type: none"> <li>0 - проверка осуществляется;</li> <li>1 - проверка выключена. В случае несовпадения переданных и принятых данных генерируется статус ECHO_LOOP_ERROR - ошибка в петле обратной связи в дескрипторе КШ / ОУ.</li> </ul>	R/W	0
2	MS	Mode Select. Режим работы контроллера: <ul style="list-style-type: none"> <li>0 - Remote Terminal;</li> <li>1 - Bus Controller. Замечание. Подконтроллеры BC и RT находятся под сбросом и с отключенным синхросигналом, если не активированы с помощью бит MS и BE. После включения соответствующего режима, следует выдерживать задержку перед обращением к блоку из расчета 16 тактов частоты APB + 16 тактов частоты ядра (20 МГц).</li> </ul>	R/W	0
1	BE	Bus Enable. Разрешение работы с шиной. Режим работы определяется битом MS: <ul style="list-style-type: none"> <li>0 - запрещено;</li> <li>1 - разрешено.</li> </ul>	R/W	0
0	BME	Bus Monitor Enable. Разрешение работы в режиме Bus Monitor: <ul style="list-style-type: none"> <li>0 - запрещено;</li> <li>1 - разрешено.</li> </ul>	R/W	0

**BC\_ADDR [0x4]: Регистр начального адреса очереди заданий КШ (по записи).**

При чтении отображает адрес текущего обрабатываемого задания, либо последнее обработанное задание (**Control.TS=0**).

Биты	Название	Описание	Доступ	Сброс
31:4	ADDR	Адрес в памяти, по которому расположен список заданий для КШ. Выровнен до границы 16 байт. При чтении отображает адрес текущего обрабатываемого задания. Старшие 12 бит используются в качестве базового адреса. Изменения младших 16 бит происходит в пределах базового адреса.	R/W	x
3:0	-	Резерв.	R	0

**Control [0x8]: Регистр управления**

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв	R	0

Биты	Название	Описание	Доступ	Сброс
2	TS	Task State. Состояние обработки задания: • 0 - свободен; • 1 - занят.	R/W	0
1	STOP	Остановка задания. Запись 1 приводит к остановки выполнения задания, текущее сообщение будет завершено. Сбрасывается вместе с TS.	R/W	0
0	RUN	Запись 1 приводит к запуску обработки задания. Адрес задания в BC_ADDR. Бит сбрасывается автоматически (по чтению всегда равен 0). Установка приводит к сбросу TS. Не допускается запись 1 в регистр при установленном TS.	R/W	0

**Timer [0xC]: Регистр настройки таймера позволяет сконфигурировать временные параметры работы шины, отличные от стандартных значений.**

Биты	Название	Описание	Доступ	Сброс
31:29	-	Резерв	R	0
28:20	IWGmin	Inter Word Gap (min). Минимальный интервал между приемом слова и началом передачи (от 4 мкс до 12 мкс по спецификации).	R/W	80
19	-	Резерв	R	0
18:10	IWGmax	Inter Word Gap (max). Максимальный интервал ожидания между окончанием передачи и началом приема для определения отсутствия отклика (14 мкс по спецификации).	R/W	280
9	-	Резерв	R	0
8:0	IMG	Inter Message Gap. Интервал между передачей сообщений в тактах опорной частоты 20 МГц. Значение по умолчанию соответствует 4 мкс. Интервал используется КШ для разделения сообщений во времени.	R/W	80

**LSTAT [0x10]: Регистр статуса последнего завершенного сообщения содержит копию статусного слова дескриптора.**

В случае ошибки DMA, контроллер не пишет результат в АНВ и результат доступен в регистрах контроллера.

**BM\_ADDR\_INIT [0x14]: Начальный адрес буфера МШ.**

Биты	Название	Описание	Доступ	Сброс
31:3	ADDR	Адрес, с которого контроллер начинает работать в режиме МШ. Загружается в контроллер при установке бита Config.BME. Не допускается менять содержимое регистра при включенном МШ (Config.BME = 1).	R/W	x
2:0	-	Резерв	R	0

**BM\_ADDR\_LAST [0x18]: Конечный адрес буфера МШ.**

При достижении границы МШ переходит к адресу **BM\_ADDR\_INIT**. Старшие 12 бит не доступны пользователю и равны старшим 12 битам **BM\_ADDR\_INIT**, т.е. размер буфера ограничен 1 МБ.

Биты	Название	Описание	Доступ	Сброс
31:20	-	Резерв	R	0
19:3	ADDR	Конечный адрес буфера МШ.	R/W	x
2:0	-	Резерв	R	0

**BM\_ADDR\_IRQ [0x1C]: Адрес генерации прерывания МШ.**

Прерывание генерируется при записи по адресу, указанному в **BM\_ADDR\_IRQ**.

**BM\_ADDR\_CUR [0x20]: Адрес следующей ячейки, записываемой МШ.****BM\_TIMER [0x24]: Регистр таймера МШ**

Биты	Название	Описание	Доступ	Сброс
31:27	-	Резерв	R	0
26	UPD	Сбрасывается при установке CPT. Устанавливается после обновления TMR.	R	0
25	CPT	Запись 1 приводит к загрузке значения таймера МШ в TMR. Сбрасывается аппаратно.	R/W/SC	0
24	LOAD	Запись 1 приводит к загрузке значения TMR в таймер МШ. Сбрасывается аппаратно.	R/W/SC	0
23:0	TMR	Таймер МШ. Значение, записанное в регистр, перезаписывается в таймер МШ при установке LOAD. Значение таймера МШ загружается в TMR при установке CPT.	R/W	0

**RT\_MCC [0x28]: Регистр маскирования команд удаленному терминалу**

позволяет отключить прием отдельных Mode Code, не предназначенных для использования в выбранном применении. Если установлен соответствующий бит регистра, то принятая команда получает статус ILLEGAL. Для нее устанавливается бит MESSAGE\_ERROR в статусном слове и статусное слово передается в ответ на команду (если команда не BROADCAST). Передача данных для ILLEGAL команд не производится.

Минимальный набор поддерживаемых команд задается в [6 30.4.2.1] [9 50.5.7.2.c]: TRANSMIT STATUS WORD, TRANSMITTER SHUTDOWN, OVERRIDE TRANSMITTER SHUTDOWN, RESET REMOTE TERMINAL. Согласно [9 50.5.5.c] RESET REMOTE TERMINAL и INITIATE SELF TEST могут быть отключены для некоторых режимов работы. eh\_m1553 не позволяет отключить прием команд TRANSMIT LAST COMMAND и TRANSMIT STATUS WORD. Все остальные команды могут быть отключены - переведены в ILLEGAL.

Для команд имеющих инверсные команды (Override) отключение производится одним битом на обе команды.

Биты	Название	Описание	Доступ	Сброс
31:22	-	Резерв	R	0b1111111111

Биты	Название	Описание	Доступ	Сброс
21	-	Резерв	R	0
20	-	[Override] Selected Transmitter Shutdown: 0 - прием разрешен. Поддерживаются только Dual-redundant системы.	R/W	0
19	-	Transmit BIT Word: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0
18	-	Transmit Last Command: 0 - прием разрешен.	R	0
17	-	Synchronize with data word: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0
16	-	Transmit Vector Word: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0
15:9	-	Резерв.	R	0b1111111
8	-	Reset Remote Terminal: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0
7	-	Резерв	R	0
6	-	[Override] Inhibit Terminal Flag: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0
5	-	Резерв	R	0
4	-	[Override] Transmitter Shutdown: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0
3	-	Initiate self test: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0
2	-	Transmit Status Word: 0 - прием разрешен.	R	0
1	-	Synchronize: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0
0	-	Dynamic Bus Control: 0 - прием разрешен, 1 - принятая команда игнорируется.	R/W	0

**RT\_CFG [0x2C]: Регистр настройки удаленного терминала**

Биты	Название	Описание	Доступ	Сброс
31:16	SW	Synchronization Word. Содержимое поля данных, полученное с командой SYNCHRONIZE с данными. При получении команды SYNCHRONIZE без данных записывается 0.	R	0
15:10	-	Резерв	R	0
9	EDBC	Enable Dynamic Bus Control. Значение бита Dynamic Bus Control Acceptance в статусном слове в ответ на команду Dynamic Bus Control. Если равен 1, то прерывание IRQ_DBC может быть сгенерировано.	R/W	0
8:6	-	Резерв	R	0
5	RTACT	Remote Terminal Active. Сброшен, если ОУ находится в состоянии IDLE. Формируется с учетом задержки на	R	0

Биты	Название	Описание	Доступ	Сброс
		пересинхронизацию между доменами APB и ядра контроллера.		
4:0	RTA	Remote Terminal Address. Адрес устройства в режиме удаленного терминала.	R/W	0

**RT\_ADDR [0x30]: Регистр настройки адреса дескрипторов ОУ**

Биты	Название	Описание	Доступ	Сброс
31:4	ADDR	Адрес в памяти, по которому расположены дескрипторы субадресов ОУ контроллера. Выровнен до границы 16 байт. Таблица дескрипторов содержит информацию о допустимых действиях при обращении к выбранному субадресу ОУ. Старшие 12 бит используются в качестве базового адреса. Изменения младших 16 бит происходит в пределах базового адреса.-->	R/W	0
3:0	-	Резерв.	R	0

**IRQ\_ENA [0x34]: Регистр разрешения прерываний**

Биты	Название	Описание	Доступ	Сброс
31:18	-	Резерв	R	0
17	BMED	Bus Monitor Enable IRQ DMA. Разрешение прерывания от Bus Monitor при ошибке AHB DMA. • 0 - запрещено; • 1 - разрешено.	R/W	0
16	BMEBA	Bus Monitor Enable IRQ on Buffer Access. Разрешение прерывания от Bus Monitor при пересечении адреса BM_ADDR_IRQ. • 0 - запрещено; • 1 - разрешено.	R/W	0
15:12	-	Резерв	R	0
11	RTED	Remote Terminal Enable IRQ DMA. Разрешение прерывания от Remote Terminal при ошибке AHB DMA. • 0 - запрещено; • 1 - разрешено.	R/W	0
10	RTSYN	Remote Terminal Enable IRQ Synchronize. Разрешение прерывания от Remote Terminal при получении валидной команды Synchronize. • 0 - запрещено; • 1 - разрешено.	R/W	0
9	RTERRT	Remote Terminal Enable IRQ Reset Remote Terminal. Разрешение прерывания от Remote Terminal при получении валидной команды Reset Remote Terminal. • 0 - запрещено;	R/W	0

Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"> <li>1 - разрешено.</li> </ul>		
8	RTEDBC	Remote Terminal Enable IRQ Dynamic Bus Control. Разрешение прерывания от Remote Terminal при получении валидной команды Dynamic Bus Control. <ul style="list-style-type: none"> <li>0 - запрещено;</li> <li>1 - разрешено.</li> </ul>	R/W	0
7:3	-	Резерв	R	0
2	BCED	Bus Controller Enable IRQ DMA. Разрешение прерывания от Bus Controller при ошибке AHB DMA. <ul style="list-style-type: none"> <li>0 - запрещено;</li> <li>1 - разрешено.</li> </ul>	R/W	0
1	BCER	Bus Controller Enable IRQ Error. Разрешение прерывания от Bus Controller при неудачном завершении задания. <ul style="list-style-type: none"> <li>0 - запрещено;</li> <li>1 - разрешено.</li> </ul>	R/W	0
0	BCOC	Bus Controller Enable IRQ Completion. Разрешение прерывания от Bus Controller при завершении задания. <ul style="list-style-type: none"> <li>0 - запрещено;</li> <li>1 - разрешено.</li> </ul>	R/W	0

**IRQ\_FLAG [0x38]: Регистр флагов прерываний.**

Запись 1 в любой из бит регистра приводит к сбросу флага.

Биты	Название	Описание	Доступ	Сброс
31:18	-	Резерв	R	0
17	BMID	Bus Monitor IRQ DMA. Флаг прерывания от Bus Monitor при ошибке AHB DMA.	R/W1C	0
16	BMID	Bus Monitor IRQ DMA. Флаг прерывания от Bus Monitor при пересечении адреса BM_ADDR_IRQ.	R/W1C	0
15:14	-	Резерв	R	0
13	RTIRX	Remote Terminal Receive IRQ. Флаг прерывания от Remote Terminal при окончании приема (обновлении дескриптора), если разрешено в Config.RXIRQ.	R/W1C	0
12	RTITX	Remote Terminal Transmit IRQ. Флаг прерывания от Remote Terminal при окончании передачи (обновлении дескриптора), если разрешено в Config.TXIRQ.	R/W1C	0
11	RTID	Remote Terminal IRQ DMA. Флаг прерывания от Remote Terminal при ошибке AHB DMA.	R/W1C	0
10	RTSYN	Remote Terminal IRQ Synchronize. Флаг прерывания от Remote Terminal при приеме валидной команды Synchronize.	R/W1C	0
9	RTRRT	Remote Terminal IRQ Reset Remote Terminal. Флаг прерывания от Remote	R/W1C	0

Биты	Название	Описание	Доступ	Сброс
		Terminal при приеме валидной команды Reset Remote Terminal.		
8	RTDBC	Remote Terminal IRQ Dynamic Bus Control. Флаг прерывания от Remote Terminal при приеме валидной команды Dynamic Bus Control.	R/W1C	0
7:3	-	Резерв	R	0
2	BCID	Bus Controller IRQ DMA. Флаг прерывания от Bus Controller при ошибке AHB DMA.	R/W1C	0
1	BCIE	Bus Controller IRQ Error. Флаг прерывания от Bus Controller при неудачном завершении задания.	R/W1C	0
0	BCIC	Bus Controller IRQ Completion. Флаг прерывания от Bus Controller при выполнении задания.	R/W1C	0

**REVISION [0x3C]: Регистр версии контроллера**

Биты	Название	Описание	Доступ	Сброс
31:0	VER	Версия контроллера	R	0x20220223

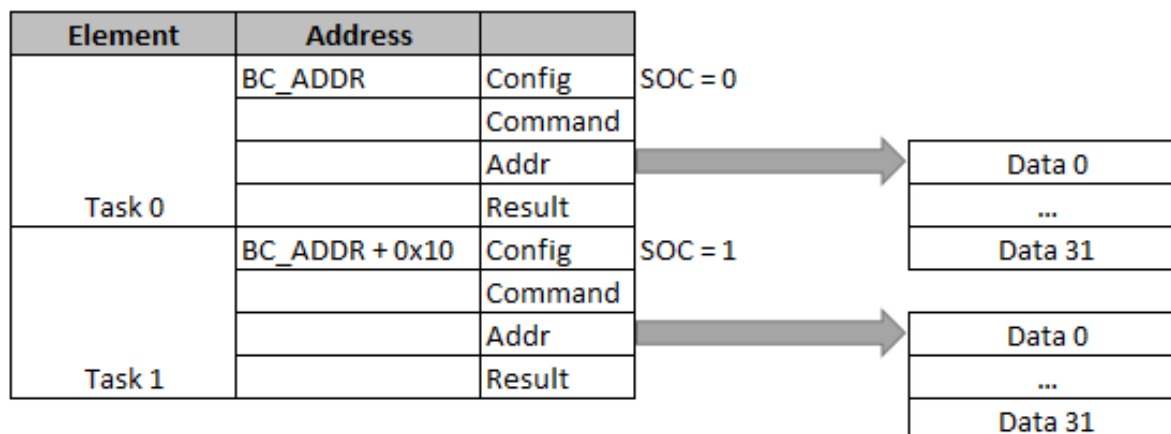


### Режим контроллера шины

Для перевода ehl\_m1553 в режим Контроллера Шины / Bus Controller следует задать CFG.MS=1 и CFG.BE=1.

Работа КШ организована в виде последовательной обработки заданий из списка (каждое задание состоит из передачи сообщения по шине). Начало обработки заданий инициируется записью 1 в CTRL.RUN.

Перед началом работы список заданий следует проинициализировать. Адрес первого задания задается в BC\_ADDR перед запуском контроллера. Размещение заданий в памяти (доступной через AHB порт контроллера) представлено на рисунке.



Формат заданий соответствует формату дескрипторов КШ. Каждое задание содержит 4 32-битных слова (**Config**, **Command**, **Addr** и **Result**). Пользователь заполняет первые 3 поля перед запуском задания. Если задание является последним в списке, для него должен быть установлен **Config.SOC**. По окончании выполнения задания контроллер заполняет **Result** (за исключением случая ошибки DMA).

Контроллер последовательно опрашивает дескрипторы (и выполняет задания), пока не обработает дескриптор с **CONFIG.SOC=1**, либо не случится ошибки в обработке задания при установленном **CONFIG.SOE=1**.

### Дескрипторы

ehl\_m1553 обрабатывает задания (в режиме Bus Controller) в соответствии с дескрипторами, расположенными в памяти. Доступ к дескрипторам, а также доступ к данным осуществляется с помощью DMA, являющегося ведущим на шине АНВ. DMA в процессе работы зачитывает дескрипторы и данные, требуемые для следующего трансфера. В процессе работы DMA возможны следующие проблемы:

- длительное чтение дескрипторов - приводит к подвисанию шины, в случае, если чтение не выполнилось за 120 тактов **ref\_clk**, генерируется прерывание **irq\_dma** и модуль прекращает обработку заданий;
- ERROR RESPONSE с шины АНВ при чтении дескрипторов - обрабатывается аналогично выставлением **irq\_dma** и прекращением обработки заданий;
- длительное чтение данных - приводит к неготовности данных к моменту начала передачи, контроллер передает данные с испорченным parity и продолжает работу с шиной. Такой подход не вносит сигнальных и протокольных ошибок в шину (кроме ошибок parity, детектируемых устройствами на шине), снижая вероятность некорректного декодирования RT состояний шины.

По завершении попытки задание повторяется (если не достигнут предел);

- ERROR RESPONSE с шины AHB при чтении данных - обрабатывается аналогично длительному чтению данных - передачей данных с испорченным parity;
- длительная запись данных - обрабатывается аналогично длительному чтению;
- ERROR RESPONSE с шины AHB при записи данных - обрабатывается аналогично длительной записи;
- длительная запись дескриптора или Error Response при записи дескрипторов — установка **irq\_dma**;

**Замечание.** В случае обнаружения ошибок DMA контроллер прекращает обработку цепочки заданий вне зависимости от количества повторений (**Config.RC**) и наличия элементов в цепочке (**Config.SOC**). В случае данного вида ошибок, содержимое **Result** дескриптора не валидно. Актуальное значение содержится в регистре **LSTAT**.

Название	Смещение	Доступ	Описание
Config	0x0	W	Настройка задания
Command	0x4	W	Командное слово
Addr	0x8	W	Адрес буфера данных. Должен быть выравнен до границы 2 байт.
Result	0xC	R	Результат выполнения задания

#### Config [0x0]: Настройка задания

Биты	Название	Описание	Доступ
31:9	-	Резерв	R
8:6	RC	Retry Count. Количество попыток передать сообщение (повторная попытка переслать сообщение в случае возникновения ошибки при значении больше 1). Допустимые значения от 1 до 7.	R/W
5	IOE	Interrupt On Error. Разрешение генерации прерывания в случае ошибки при передаче сообщения: • 0 - запрещено; • 1 - разрешено.	R/W
4	IOC	Interrupt On Completion. Разрешение генерации прерывания при завершении передачи сообщения: • 0 - запрещено; • 1 - разрешено.	R/W
3	SOE	Suspend On Error. Завершение выполнения заданий в случае ошибки в сообщении: • 0 - запрещено; • 1 - разрешено.	R/W
2	SOC	Suspend On Completion. Завершение выполнения заданий после обработки сообщения: • 0 - запрещено; • 1 - разрешено. Следует установить для последнего задания в цепочке.	R/W
1	RM	Retry mode. Выбор шины для повторной передачи сообщения: • 0 - та же шина (CONFIG.BS); • 1 - по обеим линиям поочередно.	R/W
0	BS	Bus selection. Выбор шины для передачи сообщения: • 0 - bus A; • 1 - bus B.	R/W

## Command [0x4]: Командное слово

Биты	Название	Описание	Доступ
31:26	-	Резерв	R
25:21	RTA2	RT Address 2. Адрес удаленного терминала для передачи RT-RT.	R/W
20:16	RTS2	RT Subaddress 2 для передачи RT-RT.	R/W
15:11	RTA	RT Address. Адрес удаленного терминала.	R/W
10	TR	Transmit (1)/receive (0)	R/W
9:5	RTS	RT Subaddress	R/W
4:0	WC	Word count / Mode code	R/W

Далее приведено значение полей **Command** для различных видов трансферов.

Transfer type	RTA	RTS	RTA2	RTS2	WC	TR	Data buffer direction
Data, BC-to-RT	RT address(0-30)	RT subaddr (1-30)	-	0	Word Count	0	Read (2-64 bytes)
Data, RT-to-BC	RT address(0-30)	RT subaddr (1-30)	-	0	Word Count	1	Write (2-64 bytes)
Data, RT-to-RT	Recv. RT address(0-30)	Recv. RT subaddr(1-30)	Xmit RT Addr (0-30)	Xmit RT Subaddr (1-30)	Word Count	0	Write (2-64 bytes)
Mode, no data	RT address(0-30)	0 or 31	-	-	Mode Code	1	Unused
Mode, RT-to-BC	RT address(0-30)	0 or 31	-	-	Mode Code	1	Write (2 bytes)
Mode, BC-to-RT	RT address(0-30)	0 or 31	-	-	Mode Code	0	Read (2 bytes)
Broadcast Data, BC-to-RTs	31	RTs subaddr (1-30)	-	0	Word Count	0	Read (2-64 bytes)
Broadcast Data, RT-to-RTs	31	Recv RTs subaddr(1-30)	Xmit RT Addr (0-30)	Xmit RT Subaddr (1-30)	Word Count	0	Write (2-64 bytes)
Broadcast Mode, no data	31	0 or 31	-	-	Mode Code	1	Unused
Broadcast Mode, BC-to-RT	31	0 or 31	-	-	Mode Code	0	Read (2 bytes)

## Result [0xC]: Результат

Биты	Название	Описание	Доступ
31:24	-	Резерв	R
23:16	SW2	Status Word 2. Принятое статусное слово принимающего удаленного терминала в RT-to-RT. Кодировается аналогично SW.	R
15:8	SW	Status Word. Принятое статусное слово удаленного терминала (передающего терминала в RT-to-RT). <ul style="list-style-type: none"> <li>15 - Message Error. 1 в данном поле приводит к ошибке статусного слова (Result.ST);</li> <li>14 - Instrumentation. 1 в данном поле приводит к ошибке статусного слова (Result.ST);</li> <li>13 - Service Request;</li> <li>12 - broadcast command received;</li> </ul>	R

Биты	Название	Описание	Доступ
		<ul style="list-style-type: none"> <li>11 - busy. 1 в данном поле приводит к ошибке статусного слова (Result.ST);</li> <li>10 - subsystem flag;</li> <li>9 - dynamic bus control acceptance;</li> <li>8 - terminal flag;</li> </ul>	
7:4	RC	Retry Count. Количество повторений, после которых завершено задание.	R
3	TB	Task Bus. Шина, использованная для последней передачи в задании.	R
2:0	ST	Status. Состояние задания: <ul style="list-style-type: none"> <li>000 - завершено успешно;</li> <li>001 - Remote Terminal не ответил на команду (BC-to-RT или передающий RT в RT-to-RT);</li> <li>010 - принимающий Remote Terminal не ответил на команду (RT-to-RT);</li> <li>011 - ошибка в статусном слове, полученном от Remote Terminal (один или более бит Message Error, Instrumentation, Busy, reserved установлен в 1);</li> <li>100 - ошибка протокола (нарушение тайм-аутов, ошибка декодирования, некорректная длина слов);</li> <li>101 - некорректный дескриптор задания ( 1. Mode Code без данных и TR=0; 2. RT-2-RT с адресом передатчика 31; 3. RT-2-RT с одинаковыми адресами приемника и передатчика; 4. RT-2-RT с субадресом передатчика 0 или 31; 5. широковещательный запрос с TR=1; 6. Mode Code с неподдерживаемым атрибутом TR, Broadcast; 7. Mode Code в режиме RT-2-RT; 8. RT-2-RT с TR=1).</li> <li>110 - ошибочный отклик буфера DMA или превышение тайм-аута (данный тип ошибок доступен только в регистре LSTAT, в память АНВ он не пишется);</li> <li>111 - ошибка в петле обратной связи - принимаемые данные отличны от передаваемых.</li> </ul>	R

### Режим оконечного устройства

В данном режиме (CFG.MS=0) контроллер начинает работу с шиной после записи 1 в CFG.BE. Контроллер проверяет поступающие команды на предмет совпадения адреса с RT\_CFG.RTA (или совпадение с широковещательным адресом при установленном RT\_CFG.BE). Команды адресуемые контроллеру проверяются на предмет поддерживаемого значения субадреса и обрабатываются. В случае неподдерживаемого субадреса происходит сигнализация Message Error.

Замечание. Согласно [3 4.3.3.5.4] ОУ должно сбросить статусное слово после приема валидного CW за исключением 4.3.3.5.1.7. В 4.3.3.5.1.7 говорится об опциональных кодах, которые согласно стандарту трактуются как валидные (если не было протокольных ошибок), но нелегальные. Такое ограничение вносит неясность в процедуру детектирования ОУ приема невалидной команды, поэтому eh1\_m1553 сбрасывает статусное слово при приеме всех валидных команд (легальных и нелегальных), за исключением TRANSMIT STATUS WORD и TRANSMIT LAST COMMAND.

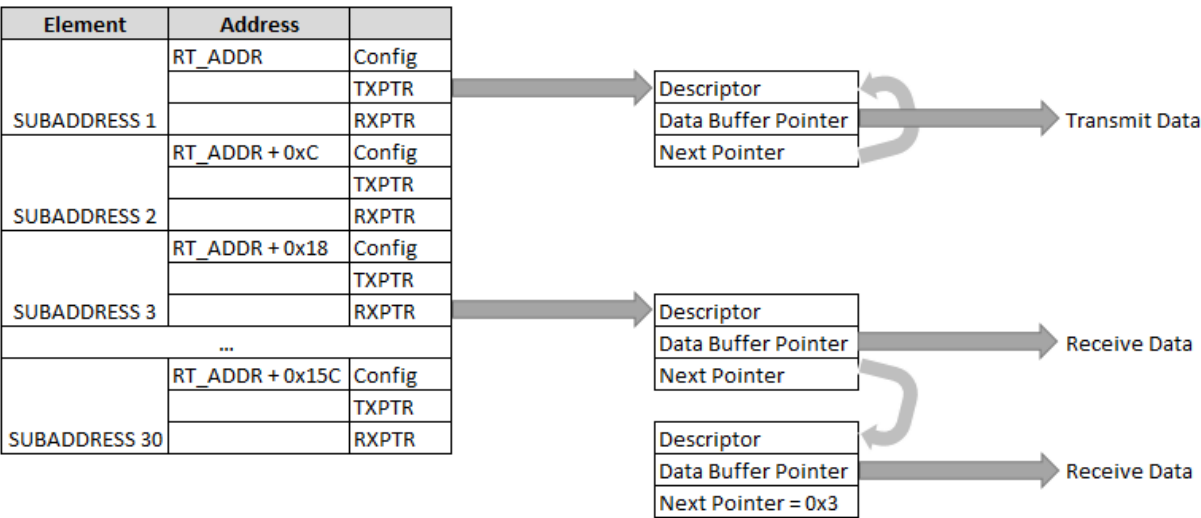
### Отключение передатчика

Передатчик ОУ может быть отключен на основной или дублирующей шине. Если команда TRANSMITTER\_SHUTDOWN поступила по шине А, то будет отключен передатчик на шине В. Если команда поступила по шине В, то будет отключен передатчик на шине А. При это повторная попытка отключить передатчик будет приводить к STATUS\_WORD.MESSAGE\_ERROR = 1. После получения команды отключения передатчика, ОУ перестает принимать команды по отключенной шине. Это позволяет избежать отключения активной шины.

Дескрипторы

Обработка команд приема/передачи данных в контроллере (в режиме ОУ) происходит в соответствии со значениями дескрипторов, расположенных в памяти по адресу, задаваемому в **RT\_ADDR**. Дескрипторы содержат состояние устройства для заданного субадреса (**Config**), а также указатели на буферы чтения и записи. Перед началом работы с ОУ необходимо проинициализировать таблицу дескрипторов.

Data Buffer Pointer должен быть выровнен до границы 2 байта, т.е. младший разряд адреса должен быть равен 0.



Config [0x0]: Настройка дескриптора субадреса

В случае получения команды, адресуемой субадресу с отключенной поддержкой данного типа транзакции (**RXEN** или **TXEN**) или неподдерживаемым размером транзакции (**RXSZ** или **TXSZ**) в STATUS слове будет установлен бит MESSAGE\_ERROR, содержимое передаваемых данных следует считать невалидным.

Биты	Название	Описание
31:18	-	Резерв
17	IGNDV	Ignore Data Valid. Игнорировать флаг Data Valid дескриптора буфера при записи в него.
16	BCRXEN	Broadcast Receive Enable. Разрешение приема широкоэвщательных команд для выбранного субадреса.
15	RXEN	Receive Enable. Разрешение приема для выбранного субадреса.
14	-	Резерв
13	RXIRQ	Receive IRQ. Разрешение прерывания по окончании приема: <ul style="list-style-type: none"><li>0 - запрещено;</li><li>1 - разрешено.</li></ul>
12:8	RXSZ	Receive Size. Максимальный поддерживаемый размер приема (в 16-битных словах) для выбранного субадреса.
7	TXEN	Transmit Enable. Разрешение передачи для выбранного субадреса.
6	-	Резерв
5	TXIRQ	Transmit IRQ. Разрешение прерывания по окончании передачи: <ul style="list-style-type: none"><li>0 - запрещено;</li><li>1 - разрешено.</li></ul>

Биты	Название	Описание
4:0	TXSZ	Transmit Size. Максимальный поддерживаемый размер передачи (в 16-битных словах) для выбранного субадреса.

**Descriptor: Дескриптор буфера.**

Заполняется по окончании выполнения команды на шине.

Биты	Название	Описание
31	DV	Data Valid. Записывается 1 при заполнении буфера контроллером. Должен быть записан 0 программно перед использованием буфера. Запись в заполненный буфер не производится, если только не сброшен CONFIG.IGNDV.
30:10	-	Резерв.
9	BC	Broadcast. Записывается, в случае широковещательного трансфера.
8	BUS	Шина, по которой пришло сообщение: 0 - bus A; 1 - bus B.
7:3	SZ	Size. Количество 16-битных слов в команде (0 соответствует 32 словам). Значение поля не валидно в случае Superseded.
2:0	TRES	Transfer Result. Результат выполнения команды: <ul style="list-style-type: none"> <li>0 - успешно;</li> <li>1 - команда прекращена в связи с поступлением новой команды (superseded);</li> <li>2 - ошибка DMA - дескриптор не прочитан до поступления следующего слова по шине;</li> <li>3 - ошибка протокола;</li> <li>4 - ошибка протокола верхнего уровня (доступ к shutdown устройству, несоответствие между содержимым команд в RT2RT);</li> <li>7 - ошибка в петле обратной связи - принимаемые данные отличны от передаваемых.</li> </ul>

**Режим Data Wrap-around**

Согласно [6 30.7] ОУ должно реализовать механизм Data Wrap-around. **ehl\_m1553** не имеет специального оборудования для поддержки указанного функционала. Тем не менее он может быть реализован программно. Для этого следует настроить субадрес 30 с одним и тем же адресом буфера для приема и передачи сообщений. Сам буфер при этом должен работать в циклическом режиме. Такая настройка позволяет читать из ОУ те же данные, которые только что были в него записаны.

**Монитор шины**

МШ может быть включен независимо от КШ/ОУ с помощью **Config.BME** и позволяет записывать принятые с шины данные с метками времени в кольцевой буфер. Для работы в режиме МШ следует также запрограммировать регистры **BM\_ADDR\_INIT**, **BM\_ADDR\_LAST** и **BM\_ADDR\_IRQ**.

Каждая запись, сделанная МШ в память, состоит из 2 слов: метки времени и данных. Записи имеют следующий формат:

Биты	Название	Описание
63	DVAL	Признак валидности слова. МШ записывает 1 при записи слова.
62:51	-	Резерв
50	BUS	Шина: 0 - A, 1 - B.
49	WST	Word Status. 0 - OK, 1 - ошибка протокола (четность, кодировка Манчестер, длина слова).
48	WTP	Word Type. Тип слова: <ul style="list-style-type: none"> <li>0 - данные;</li> <li>1 - команда / статус.</li> </ul>
47:32	WD	Word Data. Содержимое слова данных.

Биты	Название	Описание
31	VAL	Признак валидности метки времени. МШ записывает 1 при заполнении памяти.
30:24	-	Резерв.
23:0	TS	Time Stamp. Метка времени.

## I2C

Основные характеристики модуля I2C: только две линии - последовательная линия данных (i2c\_sda) и последовательная линия синхронизации (i2c\_scl); возможность работы в multi-master среде; последовательная передача данных по 8 бит; скорости передачи данных: 100 кбит/с, 400 кбит/с; фильтрация сигналов на линиях передачи данных (i2c\_sda, i2c\_scl) от помех.

Все операции на шине I2C осуществляются при помощи двух проводов i2c\_sda и i2c\_scl. Как i2c\_sda, так и i2c\_scl являются двунаправленными линиями, которые необходимо подсоединить к положительному источнику питания через подтягивающий резистор. Когда шина свободна, обе линии за счет подтягивающих резисторов принимают высокий логический уровень.

Выходные каскады устройств, подключенных к шине, должны иметь открытый сток или открытый коллектор для обеспечения функции монтажного «И».

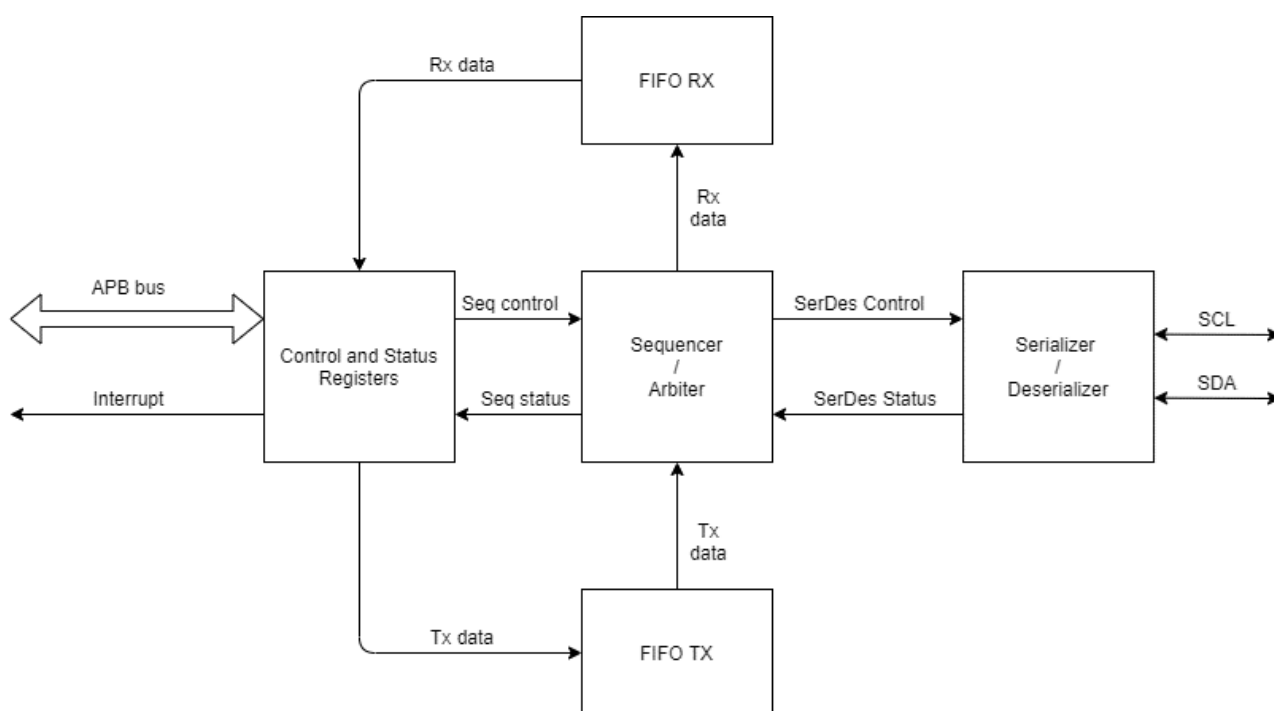
Интерфейс I2C применяется для связи между собой однокристальных микроконтроллеров, ЖК-индикаторов, портов ввода-вывода, микросхем памяти, аналого-цифровых и цифро-аналоговых преобразователей и т.д.

### Основные характеристики модуля

- только две линии - последовательная линия данных (**I2Cx\_SDA**) и последовательная линия синхронизации (**I2Cx\_SCL**);
- возможностью работы в multi-master среде;
- последовательная передача данных по 8 бит;
- скорости передачи данных: 100 кбит/с, 400 кбит/с;
- фильтрация сигналов на линиях передачи данных (**I2Cx\_SDA**, **I2Cx\_SCL**) от помех.

Все операции на шине I2C осуществляются при помощи двух проводов **I2Cx\_SDA** и **I2Cx\_SCL**. Как **I2Cx\_SDA**, так и **I2Cx\_SCL** являются двунаправленными линиями, которые необходимо подсоединить к положительному источнику питания через подтягивающий резистор. Когда шина свободна, обе линии за счет подтягивающих резисторов принимают высокий логический уровень. Выходные каскады устройств, подключенных к шине, должны иметь открытый сток или открытый коллектор для обеспечения функции "монтажного И".

### Структурная схема





- **Control and status registers** - блок содержащий в себе регистры управления I2C и статусные регистры;
- **FIFO TX** – буфер передатчика;
- **FIFO RX** – буфер приёмника;
- **Sequencer/Arbiter** – управляющий автомат интерфейса;
- **Serializer/Deserializer** – блок предназначен для преобразования параллельного потока данных от управляющего автомата в последовательный. А также для преобразования последовательно потока с внешней шины, в параллельный – для управляющего автомата.

## Регистры I2C

### Карта регистров интерфейса I2C

Смещение от базового адреса блока	Аббревиатура	Название	Описание
0x0	CFG	Configuration Register	Регистр конфигурации
0x4	CTRL	Control Register	Регистр управления
0x8	ST	Status Register	Регистр статуса
0xC	ADDR	Address Register	Содержит адрес приемника и сравнивает его с полученным адресом
0x10	PRSC	Prescaler Register	Предделитель. Определяет временную диаграмму, разворачиваемую на интерфейсной шине.
0x14	MSK	Mask Register	Регистр маски прерываний
0x18	TX_FIFO	Transmit FIFO	Буфер данных на передачу
0x1C	RX_FIFO	Receive FIFO	Буфер принятых данных
0x20	TX_WORDS	Transmit FIFO unread words	Количество не переданных слов в буфере передачи
0x24	RX_THRESHOLD	Receive FIFO threshold	Указывает количество непрочитанных слов в буфере приема, при котором формируется соответствующий признак в регистре ST

### Configuration Register [0x00]:

Биты	Название	Описание	Доступ	Сброс
31:6	-	Резерв	R	0
5:1	FILT_DEPTH	Настройка глубины фильтров для входных сигналов SDA, SCL. Выбирается таким образом, чтобы $FILT\_DEPTH * T_{core} > (T_{noise} + T_{core})$ , где $T_{core}$ - период частоты тактирования IP-ядра, $T_{noise}$ - максимальная длительность помехи, которую необходимо отфильтровать. При значении '0' фильтр отключен.	R/W	0
0	EN_OV	Разрешение перезаписи fifo буфера приемника при переполнении. При приеме в режиме MASTER:	R/W	0

Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"> <li>если переполнение разрешено, контроллер может запрашивать из SLAVE новые слова и записывать их поверх старых. При возникновении переполнения поднимается признак RXF_OV;</li> <li>если переполнение запрещено, мастер остановит прием после появления признака RXF_FULL.</li> </ul> При приеме в режиме SLAVE: <ul style="list-style-type: none"> <li>если переполнение разрешено, SLAVE может записывать новые слова поверх старых. При возникновении переполнения установится признак RXF_OV;</li> <li>если переполнение запрещено, SLAVE не будет записывать новые слова поверх старых, однако при приеме нового слова и наличии признака RXF_FULL, установится признак RXF_OV.</li> <li>0-перезапись данных запрещена</li> <li>1-перезапись данных разрешена</li> </ul>		

**Control Register [0x04].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	RST_RX_PNTRS	Запись единицы приводит к сбросу указателей FIFO_RX	R/W	0
6	RST_TX_RD_P	Запись единицы приводит к сбросу указателя чтения FIFO_TX	R/W	0
5	RST_TX_PNTRS	Запись единицы приводит к сбросу указателей FIFO_TX	R/W	0
4	ADDR_MOD	<ul style="list-style-type: none"> <li>0 - 7-ми битная адресация;</li> <li>1 - 10-ти битная адресация.</li> </ul>	R/W	0
3	ACK	Если этот бит равен единице, то после приема байта (адрес или данные) отправляется подтверждение: <ul style="list-style-type: none"> <li>0 - не отправлять подтверждение,</li> <li>1 - отправлять подтверждение. При переводе бита в '0' следующий за этим байт записывается в буфер приема, и в линию передается сигнал NACK.</li> </ul>	R/W	0
2	STOP	Запись 1 в режиме MASTER приводит к отправке сигнала STOP. Сбрасывается аппаратно после отправки сигнала STOP.	R/W	0
1	START	Запись 1 в режиме MASTER приводит к отправке сигнала START. Сбрасывается аппаратно после отправки сигнала START.	R/W	0
0	EN	<ul style="list-style-type: none"> <li>0 - отключить модуль;</li> <li>1 - включить модуль.</li> </ul>	R/W	0

**Status Register [0x08].**

Биты	Название	Описание	Доступ	Сброс
31:17	-	Резерв	R	0
16	TXF_END_EMPTY	признак того, что в момент завершения передачи слова буфер передачи был пуст. Возникает в момент завершения приема ACK, либо NACK после передачи слова. ('1' - активный уровень)	RC	0

Биты	Название	Описание	Доступ	Сброс
15	RX_TH_PA SS	количество непрочитанных слов в буфере приема больше либо равно значению регистра RX_THRESHOLD: <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	R	0
14	MODE_MS T/SLV	<ul style="list-style-type: none"> <li>0 - блок находится в режиме SLAVE;</li> <li>1 - блок находится в режиме MASTER.</li> </ul>	R	0
13	BUS_CLEA R	Признак того, что на линии не идет обмен. Линия будет считаться занятой, если зафиксирован низкий уровень на линиях SDA или SCL, и свободной - если произошло событие STOP. <ul style="list-style-type: none"> <li>0 - линия занята;</li> <li>1 - линия свободна.</li> </ul>	R	0
12	SLV_RX	в режиме SLAVE устройство адресовано на запись: <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	R	0
11	SLV_TX	в режиме SLAVE устройство адресовано на чтение: <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	R	0
10	TXF_EMPTY Y	FIFO буфер TX пуст: <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	R	0
9	TXF_FULL	FIFO буфер TX полностью заполнен: <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	R	0
8	RXF_NOT EMPTY	FIFO буфер RX не пуст. При наличии RXF_OV обнуляется, контролироваться не должен. <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	R	0
7	RXF_OV	FIFO буфер RX переполнен: <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	R	0
6	RXF_FULL	FIFO буфер RX полностью заполнен. При наличии RXF_OV обнуляется, контролироваться не должен. <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	R	0
5	ARB_LOST	Признак потери арбитража (только для режима «ведущий»). При потере арбитража в процессе передачи слова данных MASTER продолжает формировать импульсы SCL на все 8 битов данных, затем освобождает линию SCL, и после этого формирует признак ARB_LOST. <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак.</li> </ul>	RC	0
4	BUS_ERR	Не своевременное событие START или STOP. Признак появляется, если зафиксирован перепад линии SDA на фоне единичного уровня SCL при время передаче байта данных или бита подтверждения. <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак</li> </ul>	RC	0
3	ACK_FAIL	не было подтверждения после передачи байта данных/адреса. <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак</li> </ul>	RC	0
2	BTF	Обмен байтом завершен.	RC	0

Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"> <li>1 - обмен байтом завершился успешно</li> <li>0 - обмена байтом не было / завершился не успешно</li> </ul> Переводится в единицу, если: <ul style="list-style-type: none"> <li>- при приеме, когда принят новый байт и отправлен ACK/NACK</li> <li>- при передаче, если байт был передан и получен ACK</li> </ul> Примечание: если при передаче байта принят NACK, BTF не переводится в единицу.		
1	STOP	в режиме MASTER сгенерирован признак остановки в режиме SLAVE на линии обнаружен признак остановки <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак</li> </ul>	RC	0
0	START	в режиме MASTER сгенерирован признак старта в режиме SLAVE на линии обнаружен признак старта <ul style="list-style-type: none"> <li>0 - нет признака;</li> <li>1 - есть признак</li> </ul>	RC	0

**Address Register [0x0C].**

Биты	Название	Описание	Доступ	Сброс
31:10	-	Резерв	R	0
9:0	ADDR	Slave - присвоенный адрес	R/W	0

**Prescaler Register [0x10].**

Биты	Название	Описание	Доступ	Сброс
31	-	Резерв	R	0
30:26	TRISE	В значении TRISE должна содержаться максимальная длительность петли обратной связи линии SCL в режиме мастера. Используется с целью установить корректную скважность на линии SCL, которая не будет зависеть от длительности переднего фронта SCL на входе устройства и глубины фильтрации. Значение должно быть установлено как максимальное время нарастания переднего фронта SCL согласно спецификации, выраженное в периодах частоты тактирования IP-ядра плюс значение FILT_DEPTH.	R/W	0
25:16	SLV_HLD	В режиме SLAVE через поле SLV_HLD определяется длительность удержания уровня SDA после заднего фронта SCL, когда IP-ядро контролирует линию SDA. $T_{hold} = T_{filt} + T_{fall} + T_{core} * (PRSC[SLV\_HLD] + 3)$	R/W	0
15	F/S	<ul style="list-style-type: none"> <li>0 - Sm mode;</li> <li>1 - Fm mode.</li> </ul>	R/W	0
14	DUTY	<ul style="list-style-type: none"> <li>0 - <math>T_{low}/T_{high} = 2</math>;</li> <li>1 - <math>T_{low}/T_{high} = 16/9</math></li> </ul>	R/W	0
13:12	-	Резерв	R	0
11:0	PRSC	При Sm mode: <ul style="list-style-type: none"> <li><math>T_{high} = 2 * PRSC * T_{core}</math></li> <li><math>T_{low} = 2 * PRSC * T_{core}</math></li> </ul> При Fm mode и DUTY = 0: <ul style="list-style-type: none"> <li><math>T_{high} = PRSC * T_{core}</math></li> <li><math>T_{low} = 2 * PRSC * T_{core}</math></li> </ul> При Fm mode и DUTY = 1: <ul style="list-style-type: none"> <li><math>T_{high} = 9 * PRSC * T_{core}</math></li> <li><math>T_{low} = 16 * PRSC * T_{core}</math></li> </ul> Где	R/W	0

Биты	Название	Описание	Доступ	Сброс
		$T_{filt} = T_{core} * (CFG[FILT\_DEPTH] + 1)$ $T_{rise}$ - длительность переднего фронта SCL на входе устройства $T_{core}$ - частота тактирования IP-ядра $T_{high}$ - время удержания единичного уровня SCL $T_{low}$ - время удержания нулевого уровня SCL Поле PRSC определяет параметры временной диаграммы, разворачиваемой на интерфейсной шине в режиме MASTER. Смена значения на линии SDA в режиме MASTER, если он передает, происходит в момент времени $T_{low}/2$		

**Mask Register [0x14]:**

Каждый бит в регистре соответствует биту в регистре ST:

1 - прерывание формируется;

0 - прерывание не формируется.

Биты	Название	Описание	Доступ	Сброс
31:17	-	Резерв	R	0
16	TXF_END_EMPTY		R/W	0
15	RX_TH_PASS		R/W	0
14	MODE_MST/SLV		R/W	0
13	BUS_CLEAR		R/W	0
12	SLV_RX		R/W	0
11	SLV_TX		R/W	0
10	TXF_EMPTY		R/W	0
9	TXF_FULL		R/W	0
8	RXF_NOT_EMPTY		R/W	0
7	RXF_OV		R/W	0
6	RXF_FULL		R/W	0
5	ARB_LOST		R/W	0
4	BUS_ERR		R/W	0
3	ACK_FAIL		R/W	0
2	BTF		R/W	0
1	STOP		R/W	0
0	START		R/W	0

**Transmit FIFO [0x18].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:0	TX_FIFO	Буфер передаваемых данных (8 слов) Примечание: В режиме MASTER, для обращения к SLAVE необходимо записать в FIFO адрес и бит R/W. Формат данных: адрес – [7:1], R/W[0].	W	0

**Receive FIFO [0x1C].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0

Биты	Название	Описание	Доступ	Сброс
7:0	RX_FIFO	Буфер для принимаемых данных (8 слов)	R	0

**Transmit FIFO Unread Words [0x20].**

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3:0	TX_WORDS	Количество непрочитанных слов в буфере передатчика (от 0 до 8)	R	0

**Receive FIFO Threshold [0x24].**

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3:0	RX_THRES	Количество непрочитанных слов в буфере приемника (от 0 до 8), при котором формируется соответствующий признак в регистре статуса	R/W	0

**Программирование I2C****Работа I2C в режиме мастер:**

1) Помещаем в буфер передатчика слово, состоящее из 7-битного адреса (7 старших бит), и бита RX/TX (младший бит).

2) Помещаем в **PRSC** значение характеризующее параметры временной диаграммы. Запись в данный регистр обязательна в режиме MASTER (с учетом ограничений блока).

**ПРИЁМ:**

1) Записываем значение в **RX\_TRESHOLD** (от 0 до 8), в случае приема данных по I2C.

2) Записываем значение 0x0000000B в регистр **CTRL** (разрешаем работу на линии, подаем сигнал START, подаем и разрешаем подтверждение).

3) После окончания приема необходимо подать сигнал STOP, записью в регистр **CTRL**.

**ПЕРЕДАЧА:**

1) Разрешаем передачу записью соответствующего значения в регистр **CTRL**.

2) Записываем данные в регистр **TX\_FIFO**, записываем их после начала передачи с целью передачи 8 байт данных, вместо 7. В противном случае одно слово из буфера передатчика будет занято адресной командой.

3) Ожидаем окончания передачи и подаем сигнал STOP, записью в регистр **CTRL**.

## OWI

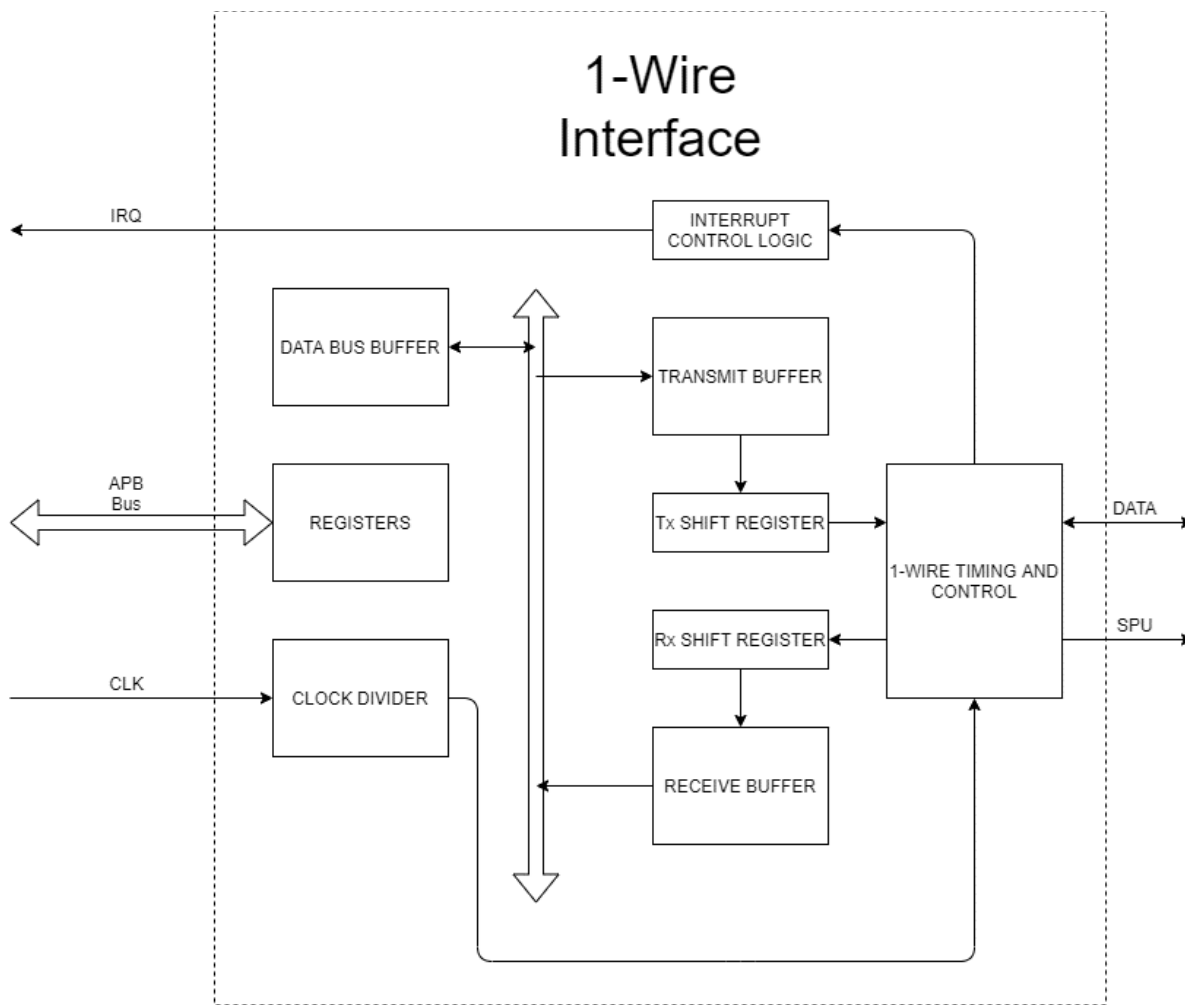
OWI или 1-Wire – это интерфейс, который представляет собой двунаправленную шину связи для устройств с низкоскоростной передачей данных, в которой данные передаются по цепи питания. Используется всего два провода, один общий, а второй для питания и данных.

Основные характеристики модуля: только режим «ведущий»; поддержка стандартной и повышенной скоростей обмена; поддержка набора стандартных команд; подсчет CRC-8; расширенный набор статусов и прерываний.

### Основные характеристики модуля

- только режим «ведущий»;
- поддержка стандартной и повышенной скоростей обмена;
- поддержка набора стандартных команд;
- подсчет CRC-8;
- расширенный набор статусов и прерываний.

### Структурная схема



### Перечень блоков OWI

OWI состоит из следующих блоков:

- REGISTERS – регистровый модуль. Предназначен для хранения управляющих данных и

статусов;

- DATA BUS BUFFER – модуль-мультиплексор. Распределяет данные между блоками системы;
- INTERRUPT CONTROL LOGIC – модуль управления формированием прерываний;
- TRANSMIT BUFFER – буфер передатчика;
- Tx SHIFT REGISTER – сдвиговый регистр передатчика;
- RECEIVE BUFFER – буфер приемника;
- Rx SHIFT REGISTER – сдвиговый регистр приемника;
- CLOCK DIVIDER – модуль деления частоты для owi;
- 1-WIRE TIMING AND CONTROL – основной блок интерфейса. Отвечает за корректное функционирование. Содержит управляющие автоматы.

В интерфейсе присутствует вывод SPU (Strong Pull-up) управления затвором транзистора р-канала, который обходит слабый нагрузочный резистор, чтобы обеспечить ведомое устройство источником питания с большим выходным током, для управления шиной данных.

#### Регистры OWI

Смещение от базового адреса блока	Аббревиатура	Название	Доступ	Описание
0x0	OWI_CFG	Configuration	RW	Регистр конфигурации
0x4	OWI_BUF	Data	RW	Регистр данных
0x8	OWI_ST	Status	R	Регистр статуса
0xC	OWI_MSK	Mask	RW	Регистр масок
0x10	OWI_PRCR	Clock Divider	RW	Регистр делителя частоты
0x14	OWI_CTRL	Control	RW	Регистр управления

#### Предупреждение

#### OWI\_CFG [0x00]

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3	OW_IN	One Wire Input. Текущее значение на линии 1-wire: • 1 - значение на линии соответствует логической единице; • 0 - значение на линии соответствует логическому нулю.	R/W	0
2	FOW	Force One Wire. Прямое управление линией 1-wire: • 1 - переводит линию в состояние логического нуля; • 0 - оставляет линию в состоянии высокого импеданса.	R/W	0
1	SRA	Search ROM Accelerator. Разрешение альтернативной функции порта: • 1 - блок работает в режиме «search ROM»; • 0 - блок работает в нормальном режиме.	R/W	0
0	1WR	One Wire Reset. Управление генерацией сигнала reset:	R/W	0



Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"> <li>1 - выставление бита инициирует передача сигнала reset по линии, бит автоматически очищается после завершения передачи сигнала;</li> <li>0 - сигнализирует о том, что в настоящий момент сигнал reset не генерируется.</li> </ul>		

## OWI\_BUF [0x04]

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:0	DATA	Двухнаправленный регистр данных.	R/W	0

## OWI\_ST [0x08]

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	OW_LOW	One Wire Low. Прием импульса присутствия от подчиненного устройства: <ul style="list-style-type: none"> <li>1 - импульс присутствия был принят в момент нахождения устройства в состоянии ожидания, бит очищается при чтении;</li> <li>0 - импульс присутствия принят не был.</li> </ul>	R	0
6	OW_SHORT	One Wire Short. Ошибка передачи временного слота или сигнала сброса по шине, поскольку в момент передачи линия уже находилась в состоянии логического нуля: <ul style="list-style-type: none"> <li>1 - зафиксирована ошибка передачи, бит очищается при чтении;</li> </ul>	R	0
5	RSRF	Receive Shift Register Full. Готовность принятых данных в сдвиговом регистре: <ul style="list-style-type: none"> <li>1 - сдвиговый регистр полон, бит очищается после чтения данных из сдвигового регистра;</li> <li>0 - сдвиговый регистр пуст или заполняется в настоящий момент.</li> </ul>	R	0
4	RBF	Receive Buffer Full. Заполнение буфера приемника: <ul style="list-style-type: none"> <li>1 - в буфере приемника находится байт с новыми данными;</li> <li>0 - в буфере приемника нет новых данных.</li> </ul>	R	0
3	TEMT	Transmit Shift Register Empty. Сдвиговый регистр передатчика пуст и готов принять новый байт данных: <ul style="list-style-type: none"> <li>1 - регистр пуст;</li> <li>0 - в настоящее время ведется передача данных по линии из сдвигового регистра передатчика.</li> </ul>	R	1
2	TBE	Transmit Buffer Empty. Буфер передатчика пуст: <ul style="list-style-type: none"> <li>1 - буфер передатчика пуст и готов к приему нового байта;</li> <li>0 - буфер передатчика ожидает готовности сдвигового регистра передатчика к приему данных.</li> </ul>	R	1
1	PDR	Presence Detect Result. Фиксация импульса «presence detect» после передачи сигнала сброса на линии: <ul style="list-style-type: none"> <li>1 - присутствие подчиненного устройства не обнаружено;</li> <li>0 - обнаружено присутствие подчиненного устройства.</li> </ul>	R	0

Биты	Название	Описание	Доступ	Сброс
0	PD	Presence Detect. Окончание передачи сигнала сброса по линии: • 1 - передача сигнала сброса завершена, бит очищается при чтении; • 0 - с момента предыдущего чтения регистра передача сигнала сброса линии не завершалась.	R	0

**OWI\_MSK [0x0C]**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	EOWL	Enable One Wire Low Interrupt. Разрешение прерывания по статусу OW_LOW: • 1 - прерывание включено; • 0 - прерывание отключено.	R/W	0
6	EOWSH	Enable One Wire Short Interrupt. Разрешение прерывания по статусу OW_SHORT: • 1 - прерывание включено; • 0 - прерывание отключено.	R/W	0
5	ERSF	Enable Receive Shift Register Full Interrupt. Разрешение прерывания по статусу ERSF: • 1 - прерывание включено; • 0 - прерывание отключено.	R/W	0
4	ERBF	Enable Receive Buffer Full Interrupt. Разрешение прерывания по статусу ERBF: • 1 - прерывание включено; • 0 - прерывание отключено.	R/W	0
3	EITEMT	Enable Transmit Shift Register Empty Interrupt. Разрешение прерывания по статусу EITEMT: • 1 - прерывание включено; • 0 - прерывание отключено.	R/W	0
2	ETBE	Enable Transmit Buffer Empty Interrupt. Разрешение прерывания по статусу ETBE: • 1 - прерывание включено; • 0 - прерывание отключено.	R/W	0
1	IAS	INTR Active State. Разрешение прерывания по статусу IAS: • 1 - прерывание включено; • 0 - прерывание отключено.	R/W	0
0	EPD	Enable Presence Detect Interrupt. Разрешение прерывания по статусу EPD: • 1 - прерывание включено; • 0 - прерывание отключено.	R/W	0

## OWI\_PRCR [0x10]

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7	CLK_EN	Clock Enable. Разрешение работы предделителя частоты. Выставляется одновременно с установкой предделителя: • 1 - предделитель частоты включен; • 0 - предделитель частоты выключен.	R/W	0
6:5	-	Резерв	R	0
4:2	DIV	Divider. Выбор коэффициента делителя частоты: • 111 - коэффициент 128; • ... • 011 – коэффициент 8; • 010 – коэффициент 4; • 001 – коэффициент 2; • 000 – коэффициент 1.	R/W	0
1:0	PRE	Prescaler Divider. Выбор коэффициента предделителя частоты: • 11 – коэффициент 7; • 10 – коэффициент 5; • 01 – коэффициент 3; • 00 – коэффициент 1.	R/W	0

## OWI\_CTRL [0x14]

Биты	Название	Описание	Доступ	Сброс
31:7	-	Резерв	R	0
6	OD	Overdrive. Режимом ускоренной передачи по линии: • 1 - режим ускоренной передачи по линии выключен; • 0 - режим ускоренной передачи по линии включен.	R/W	0
5	BIT_CTL	Bit Control. Управление режимом побитовой передачи: • 1 - режим побитовой передачи включен, передается только младший значащий бит из каждого поступившего в буфер байта; • 0 - режим побитовой передачи выключен.	R/W	0
4	STP_SPLY	Strong Pull-up Supply. Управление состоянием линии SPU во время нахождения блока в режиме ожидания: • 1 - по линии SPU передается логическая единица во время нахождения блока в состоянии ожидания, если линия усиленной подтяжки включена; • 0 - по линии SPU передается логический ноль во время нахождения блока в состоянии ожидания.	R/W	0
3	STPEN	Strong Pull-up Enable. Управление линией усиленной подтяжки: • 1 - линия усиленной подтяжки включена; • 0 - линия усиленной подтяжки выключена.	R/W	0
2	EN_FOW	Enable Force One Wire. Управление режимом функционирования бита FOW регистра OWIx_CFG: • 1 - включает функционирование бита FOW регистра OWIx_CFG; • 0 - отключает функционирование бита FOW регистра OWIx_CFG.	R/W	0
1	PPM	Presence Pulse Masking Mode. Управление режимом маскирования импульса присутствия:	R/W	0

Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"><li>1 - включает режим маскирования импульса присутствия;</li><li>0 - отключает режим маскирования импульса присутствия.</li></ul>		
0	LLM	Long Line Mode. Управление режимом длинной линии: <ul style="list-style-type: none"><li>1 - включает режим длинной линии;</li><li>0 - отключает режим длинной линии.</li></ul>	R/W	0

## SPI

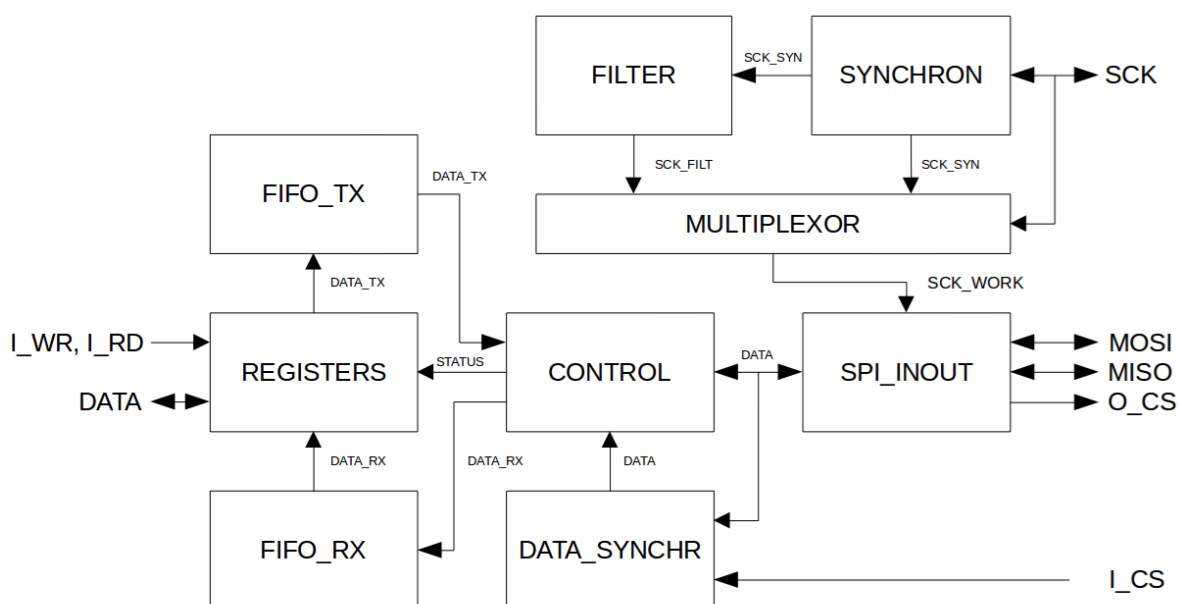
SPI – последовательный синхронный стандарт передачи данных в режиме дуплекса, предназначенный для обеспечения простого высокоскоростного сопряжения микроконтроллера и периферии.

Основные особенности: работа в режиме «ведущий» и «ведомый»; режим полнодуплексной синхронной передачи информации по трем линиям; симплексная синхронная передача по двум линиям; 8 или 16-битный формат обмена; работа на линии с несколькими мастерами; программируемая полярность и фаза синхросигнала; программируемый порядок следования битов - старшим вперед или младшим вперед; отдельные буферы на приём и передачу глубиной 8 слов; фильтр по линии синхросигнала; выделенные флаги приема и передачи; флаги ошибок режима ведущего.

### Основные особенности

- работа в режиме «ведущий» и «ведомый»;
- режим полнодуплексной синхронной передачи информации по трем линиям;
- симплексная синхронная передача по двум линиям;
- 8- или 16-битный формат обмена;
- отдельные буферы на приём и передачу глубиной 8 слов размером 16 бит;
- работа на линии с несколькими мастерами;
- программируемая полярность и фаза синхросигнала;
- программируемый порядок следования битов - старшим вперед или младшим вперед;
- фильтр по линии синхросигнала;
- выделенные флаги приема и передачи;
- флаги ошибок режима ведущего.

### Структурная схема



**Перечень блоков SPI**

- SPI состоит из следующих блоков:
- REGISTERS – регистровый модуль. Предназначен для хранения управляющих данных и статусов;
- FIFO\_TX – буфер передатчика;
- FIFO\_RX – буфер приёмника;
- CONTROL – управляющий автомат модуля SPI;
- SPI\_INOUT – блок управления линиями интерфейса;
- DATA\_SYNCHR – синхронизация данных. Производит синхронизацию данных с системной частоты на частоту SCK и в обратном направлении;
- SYNCHRON – синхронизатор. Производит синхронизацию линии SCK, когда это требуется пользователю;
- FILTER – фильтр. Фильтрует линии SCK, избавляя её от помех;
- MULTIPLEXOR – мультиплексор. Выбирает частоту тактирования блока SPI\_INOUT в зависимости от входных настроек.

**Режим «ведущий» (master)**

В режиме «ведущий» модуль автоматически генерирует синхросигнал для ведомого устройства по линии SCK.

**Процедура работы с «ведущим»:**

- в регистре **SPI\_CFG**:
  - установить биты **BR** для определения скорости обмена.
  - установить биты **POL** и **PHAZ** для определения соответствия между данными и синхросигналом. Настройка должна быть одинакова для ведущего и для ведомого;
  - настроить формат кадра при помощи бита **BO**;
  - установить бит **MOD** для перехода в режим «ведущий».
  - разрешить работу ведомому. При записи 1 в данный бит(**SADDR**), линия **O\_CS** перейдёт в значение 0, таким образом разрешив работу ведомому устройству. Примечание: Не рекомендуется записывать значение данного бита, до настройки **POL** и **PHAZ**, подобная запись может привести к ошибкам.
  - определить формат данных с помощью бита **DWW**;
- в регистре **SPI\_MSK**:
  - разрешить необходимые прерывания с помощью бит **NULL\_BUF**, **MOD\_FAIL**, **TXB\_E**, **RXB\_OV**.
- для режимов, связанных с передачей данных, необходимо записать отправляемую информацию в регистр **SPI\_BUFTX**;

- в регистре **SPI\_CTRL**:
  - разрешить работу модуля битом **EN**.
- принимаемые данные можно вычитать из регистра **SPI\_BUFRX**.

В данном формате пин **MOSI** работает как выход данных, пин **MISO** как вход данных.

### Особенности работы в режиме «ведущий»

В режиме «ведущий», если была разрешена работа модуля при пустом буфере передатчика и при нулевой фазе, то в сдвиговый регистр будет сохранено слово заданной разрядности значения 0. Данное слово будет передано первым при записи данных в буфер передатчика, затем будут переданы записанные данные.

Аналогичное поведение будет при работе в нулевой фазе в случае, когда были переданы все имеющиеся данные из буфера передатчика, но после этого модуль не был выключен. По последнему фронту синхросигнала в регистр передатчика будет записано слово заданной разрядности значения 0. Данное слово будет передано первым при записи новых данных в буфер передатчика, затем будут переданы записанные данные.

### Режим «ведомый» (slave)

В режиме «ведомый» синхросигнал поступает от ведущего устройства. Значение бита BR не влияет на работу модуля. Рекомендуется включить модуль до того, как начнет передаваться синхросигнал, иначе будут приняты сдвинутые данные. Сдвиговый регистр передатчика должен быть заполнен до первого изменения синхросигнала.

### Процедура работы с «ведомым»:

- в регистре **SPI\_CFG**:
  - установить биты **POL** и **PHAZ** для определения соответствия между данными и синхросигналом. Настройка должна быть одинакова для ведущего и для ведомого;
  - настроить формат кадра при помощи бита **BO**;
  - сбросить бит **MOD** для перехода в режим «ведомый»;
  - определить формат данных с помощью бита **DWW**.
- в регистре **SPI\_MSK**:
  - разрешить необходимые прерывания с помощью бит **NULL\_BUF**, **TXB\_E**, **RXB\_OV**.
- для режимов, связанных с передачей данных, необходимо записать отправляемую информацию в регистр **SPI\_BUFTX**;
- в регистре **SPI\_CTRL**:
  - разрешить работу модуля битом **EN**.
  - принимаемые данные можно вычитать из регистра **SPI\_BUFRX**.

В данном формате пин **MOSI** работает как вход данных, пин **MISO** как выход данных.

### Особенности работы в режиме «ведомый»

Модуль SPI может вести приём и передачу данных на системной частоте и близкой к ней. Однако это накладывает ограничение на работу с буфером передатчика в режиме «ведомый». Данные в буфер должны быть гарантированно записаны за два такта до выдачи последнего бита данных из сдвигового регистра передатчика, либо после полного окончания передачи. В противном случае, слово, которое было записано в нарушении этих требований, может быть искажено. Данное ограничение можно избежать, если разрешить работу синхронизатора.

Модуль SPI оснащен синхронизатором синхросигнала. Для включения синхронизатора необходимо записать единицу в бит **SPI\_CFG.SYN**. Синхронизатор вносит задержку в два такта системной частоты. Значит для корректного приёма и передачи данных синхрочастота должна быть уменьшена и соотноситься с системной как  $F_{sys}/7$ .

Модуль SPI оснащен фильтром синхрочастоты. Для включения фильтра необходимо записать единицу в бит **SPI\_CFG.FIL**. Не рекомендуется включать фильтр независимо от синхронизатора во избежание коллизий. Фильтр может сгладить один сбой на промежутке времени, соответствующем трём тактам системной частоты. При этом частота синхросигнала также должна быть понижена.

С учётом включенного синхронизатора и фильтра частота синхросигнала должна быть меньше системной частоты минимум в 13 раз.

### Дуплексный и симплексный режимы работы

SPI может работать в двух конфигурациях:

две однонаправленные шины данных;

одна двунаправленная шина данных (режим «только приём» или «только передача»).

#### Две однонаправленные линии данных

Стандартный режим. Активируется, когда бит **SPI\_CFG.TR\_MOD** равен нулю. Бит **SPI\_CFG.RXO** задаёт режим передачи данных. Если **SPI\_CFG.RXO** неактивен, то модуль работает как на приём, так и на передачу. Если **SPI\_CFG.RXO** активен, то модуль работает только на приём. Линия передачи остается незадействованной. У ведомого это линия **MISO**, у ведущего – **MOSI**.

#### Одна двунаправленная линия данных

Режим активируется, когда бит **SPI\_CFG.TR\_MOD** равен единице. В данном режиме модуль SPI работает только на приём, либо только на передачу, что определяется битом **EN\_TX**.

В режиме «только передача» данные передаются по пину **MOSI** у ведущего, по пину **MISO** у ведомого устройств. Принимающий пин **MISO** у ведущего или **MOSI** у ведомого не используется. В этом случае необходимо игнорировать состояние принимающего буфера.

В режиме «только приём» данные не передаются. Ведущий постоянно генерирует синхросигнал для ведомого. Пин **MISO** у ведущего или **MOSI** у ведомого не используется.

### Передача данных

В соответствии с требованиями протокола SPI, в зависимости от настройки модуля, данные передаются на линию либо по фронту, либо по спаду синхросигнала. Передача начинается при наличии данных в буфере передатчика. Данные поступают в сдвиговый регистр и далее биты слова передаются последовательно старшим или младшим вперёд в зависимости от настройки **BO**. Установка бита **TXB\_E**, сообщает о том, что буфер передатчика опустел, но в данные ещё могут находиться в сдвиговом регистре, следовательно, будет продолжаться их передача. Об окончании передачи последнего слова данных сообщает бит **NULL\_BUF**. С этого момента генерация



синхросигнала будет остановлена, до новой записи в буфер передатчика.

Буфер рассчитан на 8 слов. При переполнении буфера передатчика устанавливается бит SPI\_ST.TXB\_OV. Записываемое слово будет утеряно. Пользователь имеет возможность очистить буфер передатчика путём записи в бит TXBS\_RST единицы.

По битам TXB\_E, NULL\_BUF, TXB\_OV может быть настроено прерывание в регистре SPI\_CFG.

В режиме «только приём» генерация синхросигнала происходит в течении всей работы модуля.

### Приём данных

В соответствии с требованиями протокола SPI, в зависимости от настройки модуля, данные фиксируются по фронту, либо по спаду синхросигнала. Биты последовательно заполняют сдвиговый регистр. Затем из сдвигового регистра принятые данные поступают в буфер приёмника. Когда в буфер передано хотя бы одно слово данных, в регистре SPI\_ST выставляется бит RX\_NE.

Буфер рассчитан на 8 слов данных. Пользователь может последовательно вычитать принятые данные или очистить буфер путём записи бита RXBS\_RST в регистр SPI\_CTRL во избежание переполнения. В случае, когда буфер заполнен, приход нового слова данных вызовет установку бита RX\_OV регистра SPI\_ST. Само слово записано не будет.

В режиме «только передача» данные в буфере приёмника не фиксируются.

### Режим работы с входом выбора микросхемы

Модуль SPI имеет в своем составе вход выбора микросхемы – spi\_i\_cs.

В режиме «ведущий» пользователю необходимо держать высокий логический уровень на входе spi\_i\_cs на протяжении всей передачи данных. Сброс в нулевое значение приведёт к выключению модуля и переходу в режим «ведомый» с формированием статуса SPI\_ST.MOD\_FAIL.

В режиме «ведомый» необходимо держать низкий логический уровень на входе spi\_i\_cs на протяжении всей передачи. При появлении высокого логического уровня на данном входе, регистр приёмника будет очищен, а работа модуля остановлена до появления требуемого значения на spi\_i\_cs.

Пользователь может перейти в программный режим управлением входа spi\_i\_cs с помощью бита SPI\_CFG.SS\_CTRL. Далее пользователю достаточно имитировать необходимый логический уровень с помощью бита SPI\_CTRL.SSS. В остальном принцип работы модуля полностью соответствует работе в аппаратном режиме.

### Статусы и прерывания модуля SPI

Регистр SPI\_ST содержит семь типов статусов модуля SPI. На основании четырех из них могут быть сформированы прерывания.

Бит NULL\_BUF сообщает о том, что в сдвиговом регистре передатчика отсутствуют данные. На основании данного бита может быть сгенерировано прерывание. Для этого пользователю необходимо разрешать данное прерывание в регистре SPI\_MSK, записав единицу в бит NULL\_BUF.

Бит BUSY имеет разное поведение в зависимости от того, в каком режиме находится сейчас модуль - «ведущий» или «ведомый». В режиме «ведомый» данный статус выставляется по первому принятому биту и сбрасывается только при выключении модуля. В режиме «ведущий» статус переходит в активный уровень только при наличии данных в сдвиговом регистре передатчика или при обмене данными. Прерывание по данному биту отсутствует.

Бит TXB\_OV сообщает о том, что пользователь пытается записать слово данных в уже заполненный буфер передатчика. Записываемое слово будет утеряно. Прерывание по данному биту отсутствует. Бит сбрасывается при чтении.

Бит RX\_NE сообщает пользователю о том, что в буфере приёмника содержится хотя бы одно слово данных. Если пользователь сбросит буфер или вычитает все слова, то бит перейдет в неактивное состояние. Прерывание по данному статусу отсутствует.

Бит MOD\_FAIL сообщает пользователю о том, что вход spi\_i\_cs был переведен в низкий логический уровень в режиме «ведущий». Бит сбрасывается при чтении. На основании данного бита может быть сгенерировано прерывание. Для этого пользователю необходимо разрешать данное прерывание в регистре SPI\_MSK, записав единицу в бит MOD\_FAIL.

Бит TXB\_E сообщает пользователю о том, что буфер передатчика пуст. Статус переходит в неактивный уровень, когда в буфер будет записано хотя бы одно слово данных. Стоит учитывать, что записанное слово переместится в сдвиговый регистр, как только тот опустеет и буфер опять окажется пустым. На основании данного бита может быть сгенерировано прерывание. Для этого пользователю необходимо разрешать данное прерывание в регистре SPI\_MSK, записав единицу в бит TXB\_E.

Бит RXB\_OV сообщает пользователю о том, что буфер приемника переполнен, а значит он уже потерял одно из принимаемых слов данных. Для того, чтобы такой ситуации не происходило, необходимо вычитывать принимаемые данные из буфера через регистр SPI\_BUFRX. Статус сбрасывается при чтении. На основании данного бита может быть сгенерировано прерывание. Для этого пользователю необходимо разрешать данное прерывание в регистре SPI\_MSK, записав единицу в бит RXB\_OV.

#### Карта регистров модуля SPI

Смещение от базового адреса блока	Аббревиатура	Доступ	Описание
0x0	SPI_CTRL	RW	Регистр управления
0x4	SPI_CFG	RW	Регистр конфигурации
0x8	SPI_MSK	RW	Регистр маски
0xC	SPI_ST	RW	Регистр статуса
0x10	SPI_BUFTX	W	Регистр для записи данных в буфер передатчика
0x10	SPI_BUFRX	R	Регистр для чтения принятых данных

#### Регистры

##### SPI\_CTRL [0x00].

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3	SSS	Значение этого бита принудительно заменяет состояние входа n_cs, которое при этом игнорируется. воздействует на устройство только при установленном бите SS_CTRL (при программном управлении n_cs): <ul style="list-style-type: none"> <li>1 - имитировать высокий логический уровень;</li> <li>0 - имитировать низкий логический уровень.</li> </ul>	R/W	0
2	TXBS_RST	сброс состояния буфера передатчика. Сброс записанной «1» происходит аппаратно через один такт: <ul style="list-style-type: none"> <li>1 - сброс состояния буфера;</li> <li>0 - сброс неактивен.</li> </ul>	R/W	0
1	RXBS_RST	сброс состояния буфера приёмника. Сброс записанной «1» происходит аппаратно через один такт: <ul style="list-style-type: none"> <li>1 - сброс состояния буфера;</li> <li>0 - сброс неактивен.</li> </ul>	R/W	0
0	EN	Enable. Разрешение работы.	R/W	0

Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"> <li>1 - включить модуль;</li> <li>0 - отключить модуль.</li> </ul>		

**SPI\_CFG [0x04].**

Биты	Название	Описание	Доступ	Сброс
31:25	-	Резерв	R	0
24	SADDR	выбор ведомого устройства: <ul style="list-style-type: none"> <li>1 - включить ведомое;</li> <li>0 - отключить ведомое.</li> </ul>	R/W	0
23:16	BR	выбор скорости обмена. <ul style="list-style-type: none"> <li>0, 1 - Fpclk/2;</li> <li>2 - Fpclk/4;</li> <li>3 - Fpclk/6;</li> <li>...</li> <li>255 - Fpclk/510.</li> </ul>	R/W	0
15:11	-	Резерв	R	0
10	FIL	разрешение работы фильтра по i_sck. Фильтр необходимо включать вместе с синхронизатором. Скорость сигнала sck при включенном синхронизаторе и фильтре должна быть не более Fpclk/10. <ul style="list-style-type: none"> <li>1 - i_sck передается через фильтр;</li> <li>0 - i_sck передается без фильтрации.</li> </ul>	R/W	0
9	SYN	разрешение работы синхронизаторов по i_sck. Скорость сигнала sck при включенном синхронизаторе должна быть не более Fpclk/6: <ul style="list-style-type: none"> <li>1 - i_sck передается через два триггера;</li> <li>0 - i_sck передается напрямую.</li> </ul>	R/W	0
8	RXO	только прием. В комбинации с битом TR_MOD задает направление передачи данных <ul style="list-style-type: none"> <li>1 - выход отключен (только прием данных);</li> <li>0 - полнодуплексный режим (передача и прием данных).</li> </ul>	R/W	0
7	EN_TX	в режиме двунаправленного обмена данными по одной линии, разрешает или запрещает передачу данных <ul style="list-style-type: none"> <li>1 - выход активен (только передача данных);</li> <li>0 - выход неактивен (только прием данных). В режиме Master используется вывод MOSI, в режиме Slave – MISO.</li> </ul>	R/W	0
6	TR_MOD	разрешает или запрещает использование двунаправленного режима обмена данными по одной линии <ul style="list-style-type: none"> <li>1 - режим 1-ой двунаправленной линии данных;</li> <li>0 - режим 2-х однонаправленных линий данных.</li> </ul>	R/W	0
5	SS_CTRL	программное управление выбором устройства (n_cs). Когда этот бит установлен, вместо уровня на входе n_ss контролируется состояние бита SSS <ul style="list-style-type: none"> <li>1 - включить программный контроль;</li> <li>0 - отключить программный контроль.</li> </ul>	R/W	0
4	DWW	формат данных <ul style="list-style-type: none"> <li>1 - 16 бит;</li> <li>0 - 8 бит.</li> </ul>	R/W	0
3	BO	порядок передачи данных	R/W	0

Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"> <li>1 - первым передается младший значащий разряд – LSB;</li> <li>0 - первым передается старший значащий разряд – MSB.</li> </ul>		
2	POL	полярность тактового сигнала <ul style="list-style-type: none"> <li>1 - высокий уровень в режиме ожидания на выводе SCK;</li> <li>0 - низкий уровень в режиме ожидания на выводе SCK. В режиме Master используется вывод MOSI, в режиме Slave – MISO.</li> </ul>	R/W	0
1	PHA	фаза тактового сигнала <ul style="list-style-type: none"> <li>1 - строб данных происходит по второму перепаду тактового сигнала;</li> <li>0 - строб данных происходит по первому перепаду тактового сигнала.</li> </ul>	R/W	0
0	MOD	выбор режима работы <ul style="list-style-type: none"> <li>1 - режим Master;</li> <li>0 - режим Slave.</li> </ul>	R/W	0

**SPI\_MSK [0x08].**

Биты	Название	Описание	Доступ	Сброс
31:5	-	Резерв	R	0
4	NULL_BUF	разрешение вызова прерывания при пустом буфере передатчика на момент окончания передачи слова <ul style="list-style-type: none"> <li>1 - прерывание разрешено;</li> <li>0 - прерывание запрещено.</li> </ul>	R/W	0
3	RX_NE	разрешение вызова прерывания, когда буфер приемника стал непустой <ul style="list-style-type: none"> <li>1 - прерывание разрешено;</li> <li>0 - прерывание запрещено.</li> </ul>	R/W	0
2	MOD_FAIL	разрешение вызова прерывания при неверной настройке модуля SPI <ul style="list-style-type: none"> <li>1 - прерывание разрешено;</li> <li>0 - прерывание запрещено.</li> </ul>	R/W	0
1	TXB_E	разрешение вызова прерывания, когда буфер передатчика пустой <ul style="list-style-type: none"> <li>1 - прерывание разрешено;</li> <li>0 - прерывание запрещено.</li> </ul>	R/W	0
0	RXB_OV	разрешение вызова прерывания по переполнению буфера приёмника <ul style="list-style-type: none"> <li>1 - прерывание разрешено;</li> <li>0 - прерывание запрещено.</li> </ul>	R/W	0

**SPI\_ST [0x0C].**

Биты	Название	Описание	Доступ	Сброс
31:7	-	Резерв	R	0
6	NULL_BUF	зафиксирован пустой буфер передатчика на момент окончания выдачи слова <ul style="list-style-type: none"> <li>1 - буфер пустой;</li> <li>0 - буфер непустой.</li> </ul>	RC	0

Биты	Название	Описание	Доступ	Сброс
5	BUSY	флаг занятости. В режиме "Master" установлен, когда происходит обмен данными или регистр передатчика содержит данные. В режиме "Slave" устанавливается при начале обмена(по первому принятому биту), сбрасывается при выключении устройства. • 1 - возникло одно из перечисленных событий • 0 - перечисленные события отсутствуют.	R	0
4	TXB_OV	переполнение буфера передатчика • 1 - буфер переполнен; • 0 - буфер в нормальном состоянии.	RC	0
3	RX_NE	устанавливается, когда буфер приемника содержит данные, то есть не пустой: • 1 - буфер приёмника содержит данные; • 0 - буфер приёмника пустой.	R	0
2	MOD_FAIL	флаг ошибки настройки ведущего • 1 - ошибка настройки ведущего; • 0 - ведущий настроен верно.	RC	0
1	TXB_E	Устанавливается, когда буфер передатчика пустой. Примечание: Флаг устанавливается при переходе слова данных в выходной FIFO. • 1 - буфер передатчика пустой • 0 - буфер передатчика содержит данные	R	0
0	RXB_OV	переполнение буфера приёмника • 1 - буфер переполнен; • 0 - буфер в нормальном состоянии.	RC	0

**SPI\_BUFTX / SPI\_BUFRX [0x10]**

Данные на запись в буфер передатчика. Данный регистр предназначен для чтения принятых данных из буфера приёмника. Первое принятое слово данных сразу же выставляется на выходе данного регистра и меняется на следующее при чтении из данного регистра. При чтении из пустого буфера будут вычитаны нули.

Биты	Название	Описание	Доступ	Сброс
31:16	-	Резерв	R	0
15:0	F/S	Данные на запись / Данные на чтение.	R/W	0

## UART

UART (Универсальный Асинхронный Приемо-Передатчик) осуществляет асинхронный полнодуплексный обмен данными по последовательным линиям **rx** и **tx** с другими устройствами UART.

Основные возможности:

1. Изменение скорости передачи заданием коэффициента делителя частоты;
2. Изменение формата посылки. От 1 до 8 бит в слове данных, 1 или 2 стоп-бита, бит контроля четности (4 режима: odd, parity, space, mark);
3. Входной и выходной FIFO буферы (глубина настраивается параметром) помогают снизить количество прерываний от UART. Глубина буфера, при которой формируется прерывание, задается программно;
4. Тестовые режимы: эхо - режим, режимы внутренней и внешней петли;
5. 9-бит режим с автоматической сверкой адреса для систем из нескольких UART;
6. Высокоскоростной режим (четыре сэмпла на бит вместо шестнадцати);
7. Аппаратный контроль обмена через сигналы RTS и CTS;
8. Детектирование и формирование break-сигнала;
9. Тайм-аут программируемой длительности. Возможность инвертирования логических уровней передачи сигнала.

Uart (УАПП, Универсальный Асинхронный Приемо-Передатчик) осуществляет асинхронный полнодуплексный обмен данными по последовательным линиям **rx** и **tx** с другими устройствами uart.

### Основные возможности

- Изменение скорости передачи заданием коэффициента делителя частоты;
- Изменение формата посылки. От 1 до 8 бит в слове данных, 1 или 2 стоп-бита, бит контроля четности (4 режима: odd, parity, space, mark);
- Входной и выходной FIFO буферы по 8 байт помогают снизить количество прерываний от uart. Глубина буфера, при которой формируется прерывание, задается программно;
- Тестовые режимы: эхо - режим, режимы внутренней и внешней петли;
- 9-бит режим с автоматической сверкой адреса для систем из нескольких uart;
- Высокоскоростной режим (четыре строка на бит вместо шестнадцати);
- Аппаратный контроль обмена через сигналы **rts** и **cts**;
- Детектирование и формирование сигнала break;
- Тайм-аут программируемой длительности;
- Возможность инвертирования логических уровней передачи сигнала.

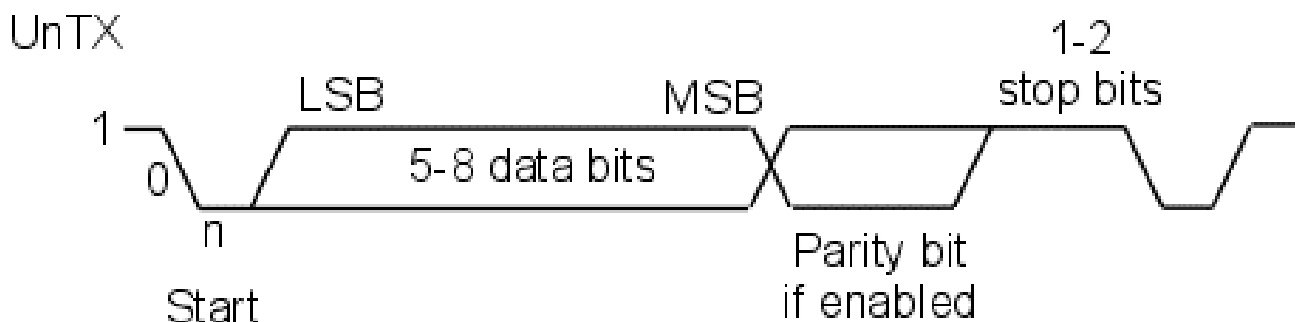
### Формат посылки

Обмен данными по последовательным линиям **rx** и **tx** происходит посылками. Посылка состоит из слова данных (иногда также называемого просто байтом) и служебных сигналов: старт-бита, бита контроля четности и стоп-бита(ов). Перед началом обмена в обоих взаимодействующих uart должен

быть выставлен одинаковый режим обмена, который включает в себя:

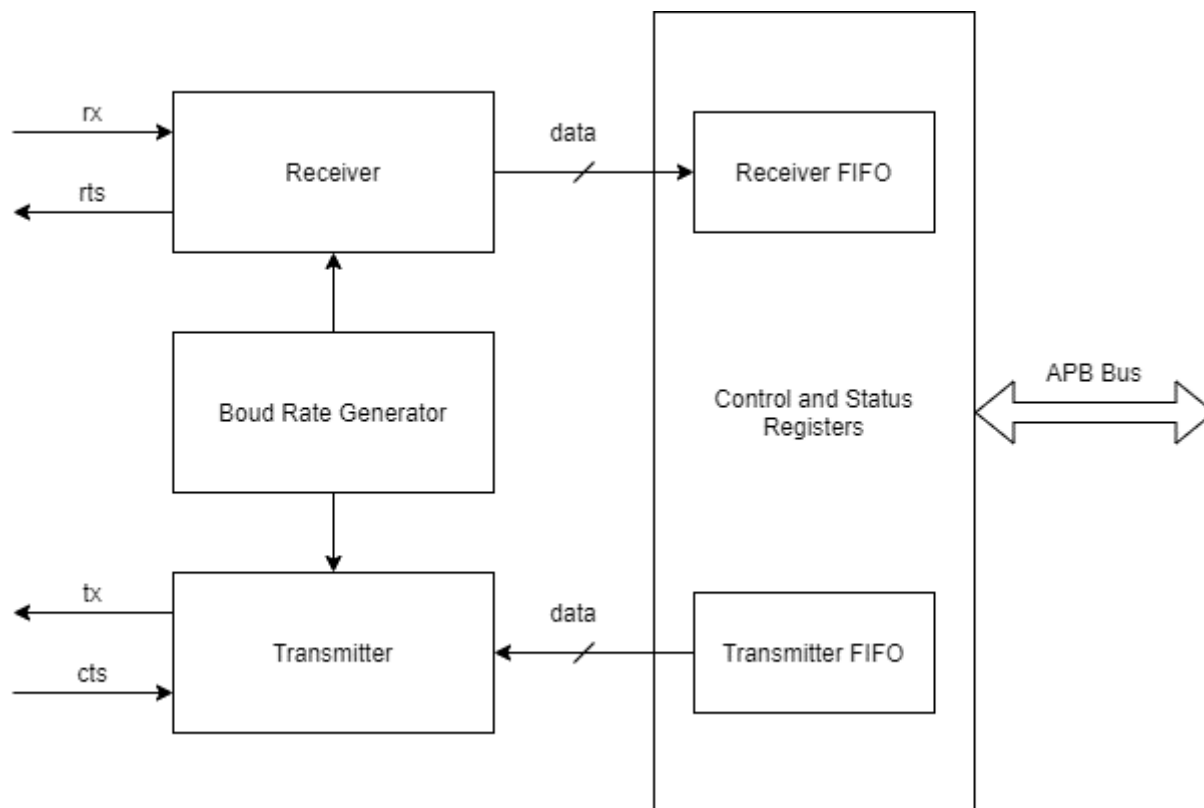
1. Скорость обмена
2. Количество бит в слове данных
3. Наличие бита контроля четности и его тип

Только тогда обмен может пройти успешно. Временная диаграмма обмена по uart представлена на рисунке 1.

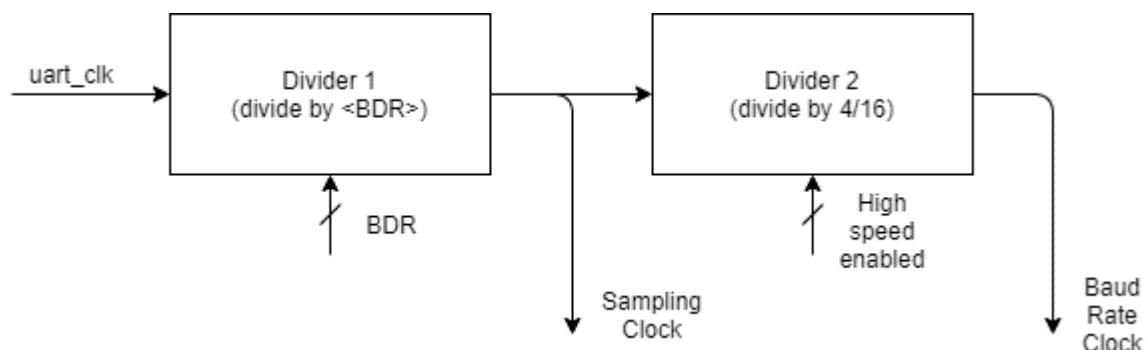


В режиме ожидания линия находится в состоянии логической единицы. Посылка всегда начинается с так называемого старт-бита (уровень логического нуля), показывающего, что далее идет слово данных (1-8 бит) младшими битами вперед. Затем следует бит контроля четности, необходимый для обнаружения одиночных ошибок передачи, и 1-2 стоп-бита (уровень логической единицы), обозначающие конец посылки. Наличие старт и стоп-битов при последовательном асинхронном обмене необходимо для синхронизации тактовой частоты взаимодействующих uart.

#### Структурная схема



## Делитель частоты



Делитель частоты `uart` состоит из двух делителей. Первый делитель осуществляет деление частоты синхроимпульсов `uart` на `BDR` - содержимое специального регистра `BDR` (от 0 до 65536). Этот делитель определяет скорость обмена. Полученная частота - `sampling_clock` - это частота, с которой приемник сканирует линию `rx` до приема валидного стартового бита.

Второй делитель делит частоту `sampling_clock` на 16 или на 4 в высокоскоростном режиме. Он определяет, сколько сканирований линии `rx` приходится на время передачи одного бита. С полученной частотой - `baud_rate_clock` - передатчик выдает биты на линию `tx`, а приемник сканирует линию `rx` в процессе приема посылки (см. рисунок 4). На время передачи одного бита приходится несколько (16 или 4) семплов линии `rx`, для того чтобы приемник мог синхронизироваться по изменению `rx` в процессе работы. Частота работы взаимодействующих `uart`'ов ресинхронизируется (счетчик-делитель сбрасывается) по стартовому биту и каждый раз, когда меняется уровень на линии `rx` в процессе приема посылки. Это позволяет справиться с «уходом» тактовых частот `uart`'ов. При `BDR = 0` делитель частоты не работает, соответственно не работают приемник и передатчик.

Для установления необходимой скорости обмена необходимо записать в регистр **BDR** соответствующий коэффициент деления, который рассчитывается по формуле:

$$BDR = \frac{f_{clk}}{16 * desired\_baud\_rate}$$

где

`BDR` - содержимое регистра `BDR`.

`fclk` - тактовая частота `uart`.

`desired_baud_rate` - желаемая скорость передачи (в бодах или, что то - же самое в данном случае, в бит/сек).

При этом если по формуле получилось не целое число, то в результате округления реальная частота обмена будет несколько отличаться от желаемой. Реальная частота обмена (`baud_rate_clock`) в бодах рассчитывается по формуле:

$$baud\_rate\_clock = \frac{f_{clk}}{16 * BDR}$$



Для самого длинного из возможных форматов посылки (12 бит) и если ресинхронизации в процессе происходить не будет (то есть посылка состоит из одних единиц/нулей кроме старт/стоп-бита) максимально допустимая разница частот взаимодействующих `uart` - 3.6% (2% в высокоскоростном режиме). При меньшем формате посылки максимально допустимая разница частот соответственно увеличивается.



Предупреждение

Для работы в высокоскоростном режиме следует использовать стабильный источник синхросигнала. Следует учитывать в расчетах нестабильность источников RC / PLL.

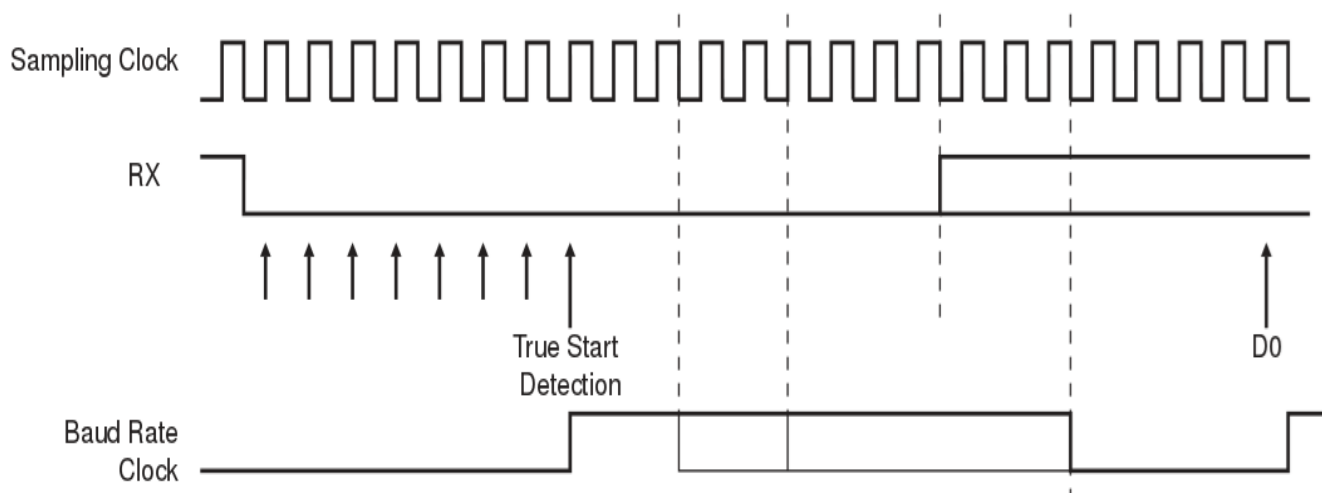
### Высокоскоростной режим

При выставлении бита «Высокоскоростной режим» регистра **CFG** в единицу (см. раздел «Карта регистров») меняется коэффициент второго делителя частоты с 16 на 4. Таким образом, в высокоскоростном режиме делитель частоты осуществляет деление на  $4 \cdot \text{BDR}$ , скорость передачи увеличивается в 4 раза. Формулы выше остаются справедливыми, если заменить число 16 на число 4. При этом синхронизация происходит менее точно. Этот режим рекомендуется использовать только при полном отсутствии помех на линии.

### Приёмник

После сброса устройства приемник выключен, программист должен включить его, записав единицу в бит `CNTR.ER` (Enable Receiver). После этого приемник начинает сканировать линию `rx` с частотой `sampling_clock`, ожидая старт-бита (уровня логического нуля). При выключении приемника записью в `DR` (Disable Receiver) единицы `uart` перестает сканировать линию, однако если в этот момент шла операция приема данных, она будет закончена, и данные будут помещены в буфер приемника.

`Uart` считает старт-битом логический нуль длительностью больше, чем половина времени передачи одного бита ( $9/16 \text{ baud\_rate\_clock}$ , то есть 9 семплов `sampling_clock`). Если уровень логического нуля держится меньшее время, это считается помехой, и `uart` продолжает сканировать линию на наличие старт-бита. Если логический нуль держится 9 сэмплов, то приемник начинает сканировать линию с меньшей в 16 раз частотой `baud_rate_clock` (таким образом, что сканирование на временной диаграмме происходит в предполагаемой середине каждого бита), помещая в регистр сдвига значение линии `rx`. Когда вся посылка принята (длина посылки определяется форматом посылки, задающимся в регистре `CFG`), приемник формирует флаги ошибок четности, стоп-бита и состояния `break` (а также соответствующие прерывания) и помещает их и принятое слово данных в буфер приемника. Если битов в слове данных выставлено меньше восьми в том же регистре `CFG`, то старшие разряды данных заполняются нулями. Если буфер заполнен, формируется прерывание переполнения буфера приемника.



Между приемником и выводом **gx** находится простейший фильтр нижних частот: три последних значения сигнала **gx** (взятых с частотой **sampling\_clock**) попадают в мажоратор 2 из 3, результат поступает в приемник. Таким образом, одиночные помехи линии длиной меньше периода **sampling\_clock** не поступают в приемник. Помехи длиной больше периода **sampling\_clock** могут привести к ошибочной ресинхронизации и ошибке приема.

### Буфер приёмника

Принятые приемником данные попадают в буфер приемника, откуда могут быть считаны программно через регистр Receiver Data Buffer Register (RX). Разряды регистра ST (Status Register) RBNE, RBRPL, RBF позволяют контролировать заполненность буфера:

Бит RBNE (Receiver Buffer Not Empty) равен 1, когда в буфере есть хотя бы одно слово данных

Бит RBRPL (Receiver Buffer Reached Preprogrammed Level) равен 1, когда количество слов в буфере больше или равно заданному в регистре RXFIFOLVL значению.

Бит RBF (Receiver Buffer Full) равен 1, когда буфер полностью заполнен. Если приемник примет еще одно слово данных, то это приведет к ошибке переполнения буфера Overrun Error.

Все эти биты равны 0, если приемник выключен.

При переходе одного из этих битов из 0 в 1 генерируется соответствующее прерывание, если его формирование включено в регистре MSK (Interrupt Mask Register).

Если выбрана конфигурация без FIFO, то буфер представляет собой только один регистр. Биты RBNE и RBF при этом имеют одинаковое значение, а RBRPL всегда равен 0 (не используется).

Буфер имеет 11 разрядов, 8 битов данных и 3 разряда под флаги ошибок стоп-бита, состояния break и ошибки четности/совпадения адреса данного слова.

### Передатчик

После сброса устройства передатчик выключен, программист должен включить его, записав единицу в бит ET (Enable Transmitter) в CNTR (см. раздел «Карта регистров»). При выключении передатчика записью в DT (Disable Transmitter) единицы uart сначала завершает передачу текущего слова данных.

Передатчик выдает биты слова данных, полученного из буфера передатчика, с частотой **baud\_rate\_clock** в соответствии с форматом посылки, указанным в CFG.

**Предупреждение**

Если передатчик и буфер передатчика пусты, от момента записи в TX до начала передачи может пройти время меньше или равное периоду `baud_rate_clock` (время передачи одного бита). Это происходит из-за того, что передатчик ждет следующего импульса `baud_rate_clock` чтобы начать передачу посылки.

**Буфер передатчика**

Передатчик берет данные для отправки из буфера передатчика, куда их можно поместить записью в регистр TX (Transmitter Data Buffer Register). Разряды регистра ST (Status Register) TBNF, TBRPL, TBE, TSRE позволяют контролировать заполненность буфера:

Бит TBNF (Transmitter Buffer Not Full) равен 1 когда в буфере есть свободное место;

Бит TBRPL (Transmitter Buffer Reached Preprogrammed Level) равен 1 когда количество слов в буфере меньше или равно заданному в регистре TXFIFOLVL значению;

Бит TBE (Transmitter Buffer Empty) равен 1 когда буфер передатчика пуст;

Бит TI (Transmitter Idle) равен 1 когда буфер передатчика пуст, а сам передатчик закончил отправку последнего слова данных и ожидает новых данных.

Все эти биты равны 0, если передатчик выключен. При переходе одного из этих битов из 0 в 1 генерируется соответствующее прерывание, если его формирование включено в MSK (Interrupt Mask Register). Таким образом, если после начальной конфигурации и задания маски прерываний включить передатчик, сработают все эти прерывания.

FIFO передатчика имеет 9 разрядов, 8 из них для передаваемого слова данных. 9-ый бит используется только в 9-бит режиме как идентификатор адреса и передается вместо бита четности.

**Флаги и события**

Вся информация о состоянии `uart` доступна в регистре состояния ST (Status Register). Регистр разделен на 2 части: флаги и события. Разница между ними в том, что флаги сбрасываются когда указанное условие перестает выполняться, а биты соответствующие событиям сбрасываются по чтению ST.

**Флаги:**

7 флагов, показывающих состояние буферов приемника и передатчика (подробнее см. выше);

Флаг CRWE (Config Registers Write Enable) равен 1, когда доступна программная запись в регистры CFG и BDR. Во время работы `uart` запись в эти регистры запрещена аппаратно;

Флаг CTSI (Clear To Send Image) показывает значение входного сигнала `cts`.

**События:**

OE (Overrun Error) Ошибка переполнения буфера приемника;

RTO (Receiver Timeout) Тайм-аут приемника;

BD (Break Detected) Зафиксирован сигнал `break`;

PE/AM (Parity Error/Address Match) Ошибка четности/совпадение адреса в 9-бит режиме;

FE (Frame Error) Ошибка стоп-бита;

CTSIC (Clear To Send Input Change) Изменение входного сигнала `cts`.

## Прерывания

По любому из битов ST можно разрешить формировать прерывание, записав 1 в соответствующий бит регистра макси прерываний MSK (расположение битов в регистрах одинаково).

Прерывание по биту-флагу формируется в момент перехода бита ST из 0 в 1 (но не наоборот). Если бит ST уже находился в 1, и в этот момент разрешается прерывание по этому биту, то прерывание тут же срабатывает.

Прерывание по биту-событию срабатывает в момент возникновения данного события. Следует иметь в виду, что бит в регистре ST установится в 1 от первого из событий, и, если его не сбросить (прочитав ST), останется в 1. Прерывания же формируются и от последующих событий тоже, неважно, в 0 или в 1 бит в регистре ST.

Прерывание возникает, как только соответствующее событие было зафиксировано. Из-за наличия буфера может быть не очевидно, при приеме какого именно слова возникла ошибка, поэтому флаги ошибки четности/совпадения адреса, ошибки стоп-бита и состояния break данного слова доступны также через старшие разряды регистра RX.

Все сигналы прерываний объединены через логическое ИЛИ в общее прерывание `uart` — сигнал `o_irq`. Все сигналы прерываний имеют длину в 1 такт. Чтобы идентифицировать, какое именно событие вызвало прерывание `uart`, необходимо прочитать содержимое ST. При этом следует иметь в виду, что бит в ST выставляется в 1 даже если формирование прерывания не разрешено в MSK, то есть ST показывает статус немаскированных прерываний.

Также возможно организовать работу `uart` без включенных в MSK прерываний с помощью программного поллинга ST.

## Дополнительные функции

### Таймер тайм-аута

В режиме сканирования линии приемником каждый период `baud_rate_clock` инкрементируется счетчик-таймер тайм-аута. Он не работает, если приемник выключен.

Таймер сбрасывается в 0, как только зафиксирован валидный старт-бит и остается в нуле во время приема данных. После приема стоп-бита, он начинает считать снова.

Таймер сбрасывается в 0, при записи 1 в разряд CNTR.RTT (Reset Timeout Timer) и остается в нуле до момента записи 0 в RTT.

Максимальное значение, до которого досчитывает таймер, определяется в разрядах CFG.TV (Timeout Value). Возможные значения: 2, 4, 8, 16, 32, 64, 128, 256 периодов `baud_rate_clock`. Досчитав до этого значения, таймер останавливается, бит RTO (Receiver Timeout) переходит в 1 и генерируется соответствующее прерывание, если его формирование разрешено в регистре MSK.

## Генерация и распознавание сигнала break

Сигнал break, то есть уровень логического нуля на линии на время большее времени передачи посылки, используется как индикатор серьезного сбоя в работе. Даже если никакая информация не может быть передана из-за слишком большой разности в частотах двух `uart`, долгий логический 0 на линии может быть правильно принят и интерпретирован приемником. Также в большинстве физических реализаций при разрыве линии сигнал `tx` также принимает значение логического нуля.

Приемник засчитывает за сигнал break посылку, в которой старт-бит, все биты данных, бит четности (если, конечно, бит четности включен в регистре CFG) и стоп-бит равны 0 (ошибка стоп-бита при этом не возникает). В момент приема последнего из битов BD (Break Detected) переходит в 1 и генерируется соответствующее прерывание, если его формирование включено в регистре MSK. В буфер приемника при этом записывается одно слово из восьми нулей с выставленным в 1 битом BD. Break сигнал может

длиться и дольше, приемник ждет конца сигнала, то есть перехода линии **gx** в 1, и только потом продолжает работу.

Передатчик выдает сигнал **break** при записи 1 в бит **CNTR.SB** (Send Break). Если запись произошла в момент передачи слова, то передача будет закончена перед выдачей сигнала **break**. Линия **tx** удерживается в состоянии логического нуля до момента записи 0 в разряд **SB**, таким образом продолжительность сигнала определяется программно. После окончания сигнала линия **tx** удерживается в 1 на 12 периодов **baud\_rate\_clock**, чтобы удаленный приемник смог корректно определить конец сигнала, предпринять необходимые действия и начать прием следующей посылки.

### Особые режимы работы

Выбор режима работы происходит записью в разряды **CFG.MODE** (Configuration Register).

#### Эхо-режим

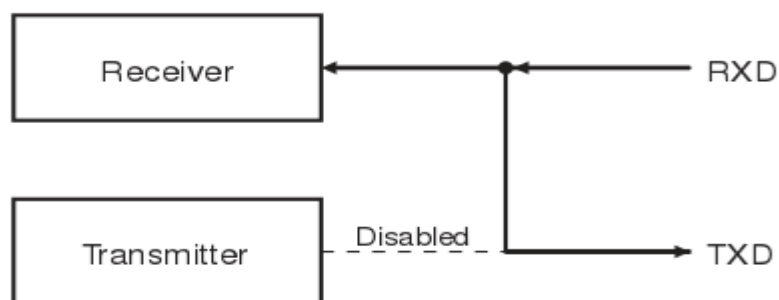
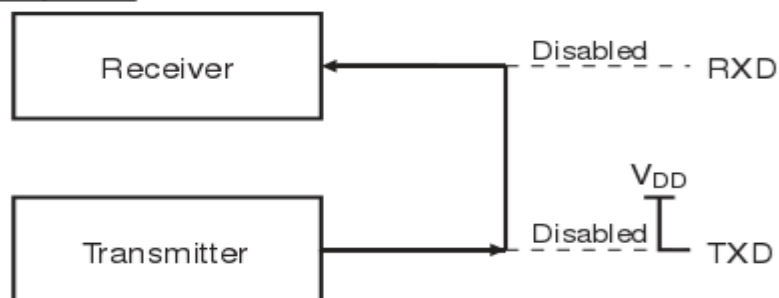
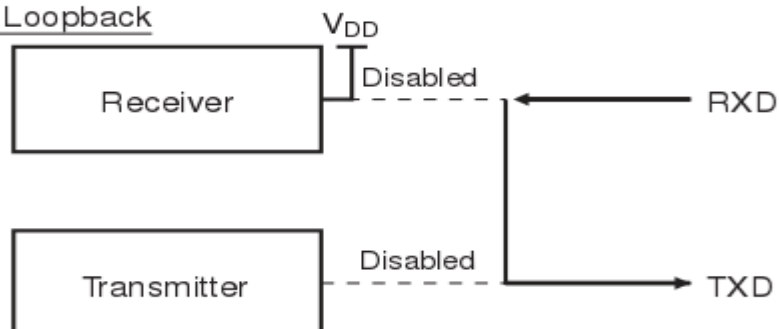
В этом режиме все данные принятые **uart** с линии **gx** побитово ретранслируются на линию **tx**. Передатчик при переходе в этот режим работы автоматически выключается, программно включить его в регистре **CTRL** нельзя.

#### Режим внутренней петли

Линии **uart tx** и **gx** замыкаются внутри устройства как показано на рисунке 5, таким образом, выдаваемая передатчиком информация может быть принята приемником. При этом на линию **tx** выдается высокий уровень. Этот режим применяется для тестирования работы **uart** без использования линии и другого **uart**.

#### Режим внешней петли

Линии **tx** и **gx** замыкаются внутри устройства как показано на рисунке 5, таким образом, вся информация, принятая по линии **gx** попадает, на линию **tx**. Отличие от эхо-режима в том, что эта информация не фиксируется приемником **uart**. Приемник и передатчик при переходе в этот режим работы автоматически выключаются, программно включить их записью в **CTRL** нельзя. Первые 3 режима работы иллюстрирует рисунок:

Automatic EchoLocal LoopbackRemote Loopback**9-битный режим**

9-бит режим, или режим с аппаратным детектированием и сверкой адреса, необходим для соединения по uart одного ведущего устройства с несколькими ведомыми. Этот режим работы может быть полезен при реализации сетевых протоколов, например Modbus RTU. При работе в этом режиме формат послыки тот же, что и в обычном режиме, но вместо бита четности ведущий uart передает бит, определяющий тип данных в послыке: 1 - в послыке содержится адрес ведомого устройства, 0 - в послыке обычное слово данных (настройки бита четности PE и PT в регистре CFG игнорируются в 9-бит режиме). Ведомое устройство распознает свой адрес, принимает последующие данные и/или передает ведущему устройству свои данные.

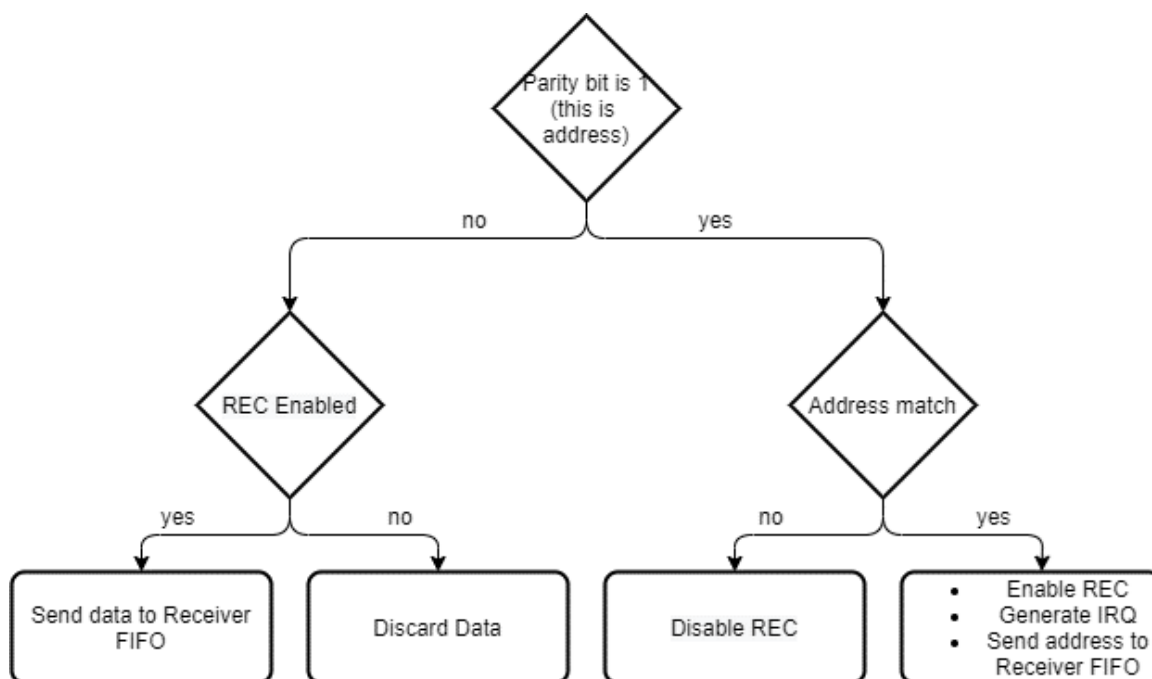
По умолчанию после включения 9-бит режима приемник не принимает слова данных, а адреса

(слова данных с 1 в девятом бите) сравнивает с заданными в регистре NBADDR (Nine Bit Mode Address Register) адресами устройства NBA1 и NBA2 (обычно один адрес индивидуальный для ведомого, а второй - широковещательный, общий для всех ведомых). Если адрес совпадает, то приемник посылает его в RX (при этом бит PE/AM регистра RX установлен в 1, показывая что это слово - адрес) формирует прерывание по совпадению адреса и устанавливает соответствующий флаг (PE/AM) в регистре ST. После этого приемник начинает принимать последующие слова данных (с 0 вместо бита четности) и продолжает делать это, пока не получит новую посылку с адресом (1 вместо бита четности). Если в формате посылки выставлено меньше восьми битов в слове, то перед сверкой идет дополнение нулями до восьми разрядов (в 9-бит режиме рекомендуется все же выставить 8 битов в слове, хотя это не обязательно). Приемник сравнивает только биты адреса, маскированные в NBMSK (Nine Bit Mode Mask Register). Алгоритм работы приемника показан на рисунке:

В 9 бит режиме передатчик игнорирует настройки четности в CFG. Вместо бита четности передается значение бита CTRL.SADDR.

### Аппаратный контроль обмена

Для установления аппаратного контроля обмена необходимо соединить вывод rts (Request To Send, запрос на отправку данных, выходной сигнал) с выводом cts того uart, с которым происходит обмен, и соединить вывод cts (Clear To Send, запрос на получение данных, входной сигнал) с выводом rts того uart, с которым происходит обмен.



При работе в этом режиме передатчик начинает отправку посылки, только если сигнал **cts** равен нулю.

При работе в этом режиме при приеме старт-бита оценивается состояние буфера приемника. Если в результате приема данного слова буфер приемника будет полностью заполнен, то сигнал **rts** переходит из 0 в 1, запрещая удаленному передатчику отправлять следующие данные. Чтение регистра **RX** переводит **rts** обратно из 1 в 0 разрешая удаленному передатчику дальнейшую отправку. Если приемник выключен, **rts** равно 1.

В других режимах работы сигнал **rts** управляется битами **CRTS** и **SRTS** регистра **CNTR**. Этот сигнал можно использовать для управления внешним приемопередатчиком физического уровня, например приемопередатчиком RS-485.

## Карта регистров

Смещение от базового адреса блока	Аббревиатура	Название	Описание
0x0	CFG	Configuration Register	Задание формата посылки и режима работы uart
0x4	BDR	Baud Rate Register	Шестнадцатитбитный коэффициент делителя частоты
0x8	TXFIFOLVL	Tx FIFO Level Register	Задание уровня буфера передатчика
0xC	RXFIFOLVL	Rx FIFO Level Register	Задание уровня буфера приемника
0x10	NBMSK	Nine Bit Mode Mask Register	Маска адреса в режиме 9-бит
0x14	NBADDR	Nine Bit Mode Address Register	Адрес устройства в режиме 9-бит
0x18	MSK	Interrupt Mask Register	Регулирует формирование прерываний
0x1C	CTRL	Control Register	Управление uart в процессе его работы
0x20	TX	Transmitter Data Buffer Register	Данные для передачи в линию
0x24	RX	Receiver Data Buffer Register	Данные, принятые из линии
0x28	ST	Status Register	Информация о состоянии uart и активных в данный момент прерываниях

Запись в регистры **CFG** и **BDR** запрещена аппаратно (не имеет эффекта) во время работы uart, т.к. изменение содержимого этих регистров во время работы может привести к ошибкам передачи или приема. Поэтому запись в эти регистры производится один раз в начале работы. Если необходимо изменить содержимое этих регистров, то приемник и передатчик должны быть выключены, и текущие операции приема/передачи должны закончиться. Когда эти условия выполнены, переходит в 1 бит **CRWE** (Config Registers Write Enable) регистра **ST**, показывая, что запись в регистры **CFG** и **BDR** разрешена.

## Configuration Register (CFG) [0x00].

Чтение/запись

Содержимое этого регистра определяет формат посылки и режим работы uart.

Запись в регистры **CFG** и **BDR** запрещена аппаратно (не имеет эффекта) во время работы uart, т.к. изменение содержимого этих регистров во время работы может привести к ошибкам передачи или приема. Поэтому запись в эти регистры производится один раз в начале работы. Если необходимо изменить содержимое этих регистров, то приемник и передатчик должны быть выключены, и текущие операции приема/передачи должны закончиться. Когда эти условия выполнены, переходит в 1 бит **CRWE** (Config Registers Write Enable) регистра **ST**, показывая, что запись в регистры **CFG** и **BDR** разрешена.

Биты	Название	Описание	Доступ	Сброс
31:16	-	Резерв	R	0
15	INVE	Inversion Enabled. Инверсия сигналов линии:	R/W	0



Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"> <li>0 - сигналы линии обмена не инвертированы;</li> <li>1 - сигналы линии обмена (rx, tx, rts_n, cts_n) инвертированы. Теперь активный уровень для rx и tx - 0, для rts_n и cts_n - 1.</li> </ul>		
14:12	MODE	Режим работы: <ul style="list-style-type: none"> <li>000 - обычный режим работы;</li> <li>001 - эхо-режим;</li> <li>010 - режим внутренней петли;</li> <li>011 - режим внешней петли;</li> <li>100 - 9-бит режим;</li> <li>101 - аппаратный контроль обмена.</li> </ul>	R/W	0
11:9	CHRL	Character Length. Размер слова данных: <ul style="list-style-type: none"> <li>000 - 1 бит;</li> <li>001 - 2 бита;</li> <li>010 - 3 бита;</li> <li>011 - 4 бита;</li> <li>100 - 5 битов;</li> <li>101 - 6 битов;</li> <li>110 - 7 битов;</li> <li>111 - 8 битов.</li> </ul>	R/W	7
8	PE	Parity Enabled. Контроль четности: <ul style="list-style-type: none"> <li>0 - контроль четности выключен и бит четности не формируется;</li> <li>1 - контроль четности включен и бит четности передается после слова данных.</li> </ul>	R/W	0
7:6	PT	Parity Type. Тип контроля четности: <ul style="list-style-type: none"> <li>00 - even, бит четности формируется как XOR всех битов данных в слове;</li> <li>01 - odd, бит четности формируется как XNOR всех битов данных в слове;</li> <li>10 - space, бит четности всегда равен 0;</li> <li>11 - mark, бит четности всегда равен 1.</li> </ul>	R/W	0
5	NSB	Number Of Stop Bits. Количество стоп-битов в посылке: <ul style="list-style-type: none"> <li>0 - Один стоп-бит;</li> <li>1 - Два стоп-бита. Приемник не проверяет наличие второго стоп-бита, эта настройка влияет только на передатчик.</li> </ul>	R/W	0
4	HSE	High Speed Enabled. Высокоскоростной режим: <ul style="list-style-type: none"> <li>0 - высокоскоростной режим выключен. Коэффициент делителя частоты равен 16;</li> <li>1 - высокоскоростной режим включен. Коэффициент делителя частоты равен 4.</li> </ul>	R/W	0
3	-	Резерв	RO	0
2:0	TV	Time-Out Value. Длительность тайм-аута (в периодах baud_rate_clock): <ul style="list-style-type: none"> <li>000 - 2;</li> <li>001 - 4;</li> <li>010 - 8;</li> <li>011 - 16;</li> <li>100 - 32;</li> <li>101 - 64;</li> <li>110 - 128;</li> <li>111 - 256.</li> </ul>	R/W	4

**Baud Rate Register (BDR) [0x04].**

Чтение/запись

Запись в регистры CFG и BDR запрещена аппаратно (не имеет эффекта) во время работы uart, т.к. изменение содержимого этих регистров во время работы может привести к ошибкам передачи или приема. Поэтому запись в эти регистры производится один раз в начале работы. Если необходимо изменить содержимое этих регистров, то приемник и передатчик должны быть выключены, и текущие операции приема/передачи должны закончиться. Когда эти условия выполнены, переходит в 1 бит CRWE (Config Registers Write Enable) регистра ST, показывая, что запись в регистры CFG и BDR разрешена.

Биты	Название	Описание	Доступ	Сброс
31:16	-	Резерв	R	0
15:0	CD		R/W	0

В этот регистр записывается шестнадцатитрибитный CD (Clock Divider) коэффициент делителя частоты. Для установления нужной скорости обмена необходимо записать в регистр BDR соответствующий коэффициент деления, который рассчитывается по формуле:

где

$$BDR = \frac{f_{clk}}{16 * desired\_boud\_rate}$$

BDR - содержимое регистра BDR;

desired\_baud\_rate - желаемая скорость передачи (в бодах или, что то - же самое в данном случае, в бит/сек);

fclk - входная частота uart.

Пока в регистр BDR записано начальное значение (0), Baud Rate Generator не будет работать.

**Tx FIFO Level Register (TXFIFOLVL) [0x08].**

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв	R	0
2:0	TXFIFOLVL	Transmitter FIFO Buffer Level. Заданный уровень буфера передатчика (от 0 до 7).	R/W	0

Когда количество слов в буфере передатчика меньше или равно заданному уровню, разряд **TBRPL** регистра **ST** равен 1. Таким образом, в момент опустошения буфера до заданного уровня, бит **TBRPL** регистра **ST** переходит из 0 в 1 и формируется соответствующее прерывание.

**Rx FIFO Level Register (RXFIFOLVL) [0x0C].**

Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв	R	0
2:0	RXFIFOLVL	Receiver FIFO Buffer Level. Заданный уровень буфера приемника (от 0 до 7).	R/W	0

Когда количество слов в буфере приемника больше или равно заданному уровню, разряд **ST.RBRPL** равен 1. Таким образом, в момент заполнения буфера до заданного уровня, бит **ST.RBRPL** переходит из 0 в 1 и формируется соответствующее прерывание.

**Nine Bit Mode Mask Register (NBMSK) [0x10].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:0	NBMSK	Nine Bit Mode Address Mask. Маска адреса в 9-бит режиме: <ul style="list-style-type: none"> <li>0 - этот разряд не сравнивается при приеме адреса в 9-бит режиме;</li> <li>1 (по умолчанию) - этот разряд сравнивается при приеме адреса в 9-бит режиме.</li> </ul>	R/W	0xFF

**Nine Bit Mode Address Register (NBADDR) [0x14].**

Содержимое этого регистра имеет эффект только в 9-бит режиме. Когда приемник принимает адрес (девятый бит установлен в единицу), он сравнивает его с обоими адресами **NBA1** и **NBA2**, заданными в этом регистре. При этом сравниваются только разряды адреса, маскированные в регистре **NBMSK**. При совпадении uart генерирует прерывание **PE/AM** (прерывание по совпадению адреса).

Биты	Название	Описание	Доступ	Сброс
31:16	-	Резерв	R	0
15:8	NBA2	Nine Bit Mode Address 2. Адрес устройства в 9-бит режиме №2. Значение по умолчанию: 00H.	R/W	0
7:0	NBA1	Nine Bit Mode Address 1. Адрес устройства в 9-бит режиме №1. Значение по умолчанию: 00H.	R/W	0

**Interrupt Mask Register (MSK) [0x18].**

Этот регистр управляет формированием прерываний от uart. Для каждого из битов справедливо:

0 (по умолчанию) - Данное прерывание не формируется;

1 - Данное прерывание формируется.

Возможно формирование прерывания по любому биту статусного регистра **ST**. Расположение битов в **ST** и **MSK** аналогично. Прерывание формируется в тот момент, когда соответствующий бит **ST** переходит из 0 в 1 (но не наоборот) для битов-флагов, и непосредственно в момент возникновения события для битов-событий.

Биты	Название	Описание	Доступ	Сброс
31:15	-	Резерв	R	0
14	CTSI		R/W	0
13	CRWE		R/W	0
12	TBNF		R/W	0
11	TBRPL		R/W	0
10	TBE		R/W	0
9	TI		R/W	0
8	RBNE		R/W	0

Биты	Название	Описание	Доступ	Сброс
7	RBRPL		R/W	0
6	RBF		R/W	0
5	OE		R/W	0
4	RTO		R/W	0
3	BD		R/W	0
2	PE/AM		R/W	0
1	FE		R/W	0
0	CTSIC		R/W	0

**Control Register (CTRL) [0x1C].**

Биты	Название	Описание	Доступ	Сброс
31:6	-	Резерв	R	0
5	RE	Receiver Enabled. • 0 (по умолчанию) - приемник выключен (если в момент выключения происходил прием слова, он будет завершен перед выключением); • 1 - приемник включен.	R/W	0
4	TE	Transmitter Enabled. • 0 (по умолчанию) - передатчик выключен (если в момент выключения происходила передача слова, она будет завершена перед выключением); • 1 - передатчик включен.	R/W	0
3	SB	Send Break. Послать сигнал break. Запись 1 устанавливает линию tx в 0. Если в этот момент происходила передача посылки, она будет завершена перед выдачей break сигнала. Сигнал заканчивается при записи 0 в этот бит, таким образом длительность сигнала break задается программно. После конца сигнала break линия tx устанавливается в единицу на 12 периодов baud_rate_clock. После этого передатчик продолжает работу.	R/W	0
2	RTS	Request To Send. Управляет сигналом rts_n. Значение этого бита не имеет эффекта в режиме работы «Аппаратный контроль обмена», так как в этом режиме uart сам управляет сигналом rts_n. • 0 (по умолчанию) - сигнал rts_n установлен в 0 (это активный уровень!); • 1 - сигнал rts_n установлен в 1.	R/W	0
1	RTT	Reset Timeout Timer. Сбросить таймер тайм-аута: • 0 (по умолчанию) - таймер тайм-аута работает; • 1 - таймер тайм-аута сброшен в 0 и не работает.	R/W	0
0	SADDR	Send Address. Послать адрес. Этот бит имеет эффект только в 9-бит режиме. • 0 (по умолчанию) - все записываемые в TX слова отправляются как слова данных (с 0 вместо бита четности); • 1 - все записываемые в TX слова отправляются как адреса (с 1 вместо бита четности).	R/W	0

**Transmitter Data Buffer Register (TX) [0x20].**

Запись в этот регистр запрещена (не имеет эффекта), если буфер передатчика полон (бит TBNF регистра ST равен 0). Стоит избегать записи в заполненный буфер.

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:0	CHTBT	Character to be Transmitted. Слово данных для передачи. Если количество битов в слове (биты CHRL в регистре CFG) выбрано меньше 8, то передаваемые биты - младшие. Например если выбрано 5 битов в слове, а в CHTBT записано 00001111B, то передатчик отправит на линию tx 01111B (младшими битами вперед).	W	0

### Receiver Data Buffer Register (RX) [0x24]

Чтение буфера приемника **RX**, если тот пуст (бит **ST.RBNE** равен 0) приводит к выдаче нулей во всех разрядах. Стоит избегать чтения пустого буфера.

Биты	Название	Описание	Доступ	Сброс
31:11	-	Резерв	R	0
10	BD	Break Detected. Это слово данных состоит только из 0, даже вместо стоп-бита, и поэтому было интерпретировано как сигнал break.	R	0
9	PE/AM	Parity Error/Address Match. Значение этого бита зависит от режима работы. Во всем режимах кроме 9-бит значение этого бита - ошибка четности при приеме данного слова. В 9-бит режиме, где бит четности не используется, этот бит показывает содержимое девятого бита принятого слова. <ul style="list-style-type: none"> <li>0 - Это слово - слово данных;</li> <li>1 - Это слово - адрес. Адрес посылается в RDB только если он совпал при сверке.</li> </ul>	R	0
8	FE	Frame Error. При приеме этого слова произошла ошибка стоп-бита, то есть вместо 1 на месте старт бита принят 0.	R	0
7:0	RCH	Received Character. Принятое слово данных. Если количество бит в слове (биты CHRL в регистре CFG) выбрано меньше 8, то принятые биты дополняются нулями слева до 8 перед записью в RCH. Например, если выбрано 5 битов данных в слове и приемник принял 10110B, тогда RCH = 00010110B.	R	0

### Status Register (ST) [0x28]

В этом регистре находится информация о состоянии uart. Регистр поделен на 2 части.

В старших битах (ЧАСТЬ 1) находятся флаги текущего состояния uart. Эти биты устанавливаются в 1 как только условие выполнено и сбрасываются в 0 как только оно перестает выполняться. Эти биты не сбрасываются по чтению. Если включить соответствующее прерывание в регистре маски прерываний **MSK**, то прерывание формируется при переходе соответствующего разряда **ST** из 0 в 1 (но не наоборот). Если бит **ST** уже находился в 1, и в этот момент разрешается прерывание по этому биту, то прерывание тут же срабатывает.

В младших битах (ЧАСТЬ 2) находится информация о случившихся событиях и ошибках передачи. Эти биты устанавливаются в 1 как только событие происходит и сбрасываются в 0 при чтении регистра **ST**. Если включить соответствующее прерывание в регистре маски прерываний **MSK**, то прерывание формируется как только происходит событие.

Биты	Название	Описание	Доступ	Сброс
31:15	-	Резерв	R	0
14	CTSI	Clear To Send Image. Текущее значение сигнала CTS. 1 - удаленный приемник готов принимать данные. 0 - удаленный	R	0

Биты	Название	Описание	Доступ	Сброс
		приемник не готов принимать данные. Значение этого бита берется из входного сигнала CTS_n. Если инверсия сигналов линии (бит INVE регистра CFG) выключена, то он равен инверсии CTS_n, если включена, то самому сигналу CTS_n.		
13	CRWE	Config Registers Write Enable. Запись в регистры CFG и BDR разрешена. Этот бит равен 1 при условиях: <ul style="list-style-type: none"> <li>1. Приемник и передатчик выключены;</li> <li>2. Приемник и передатчик закончили свои операции.</li> </ul>	R	0
12	TBNF	Transmitter Buffer Not Full. Буфер передатчика не полон, в нем есть место еще для как минимум одного слова данных (и передатчик включен).	R	0
11	TBRPL	Transmitter Buffer Reached Preprogrammed Level. В буфере передатчика осталось количество слов меньше или равно заданному в регистре TXFIFOLVL уровню (и передатчик включен). Если буфер синтезирован как одиночный регистр, то этот разряд всегда равен 0.	R	0
10	TBE	Transmitter Buffer Empty. Буфер передатчика пуст (и передатчик включен).	R	0
9	TI	Transmitter Idle. Буфер передатчика пуст, передатчик закончил все операции и у него нет больше слов для отправки (и передатчик включен).	R	0
8	RBNE	Receiver Buffer Not Empty. В буфере приемника есть как минимум одно неп прочитанное слово данных (и приемник включен).	RC	0
7	RBRPL	Receiver Buffer Reached Preprogrammed Level. В буфере приемника количество слов больше или равно заданному в регистре RXFIFOLVL уровню (и приемник включен). Если буфер синтезирован как одиночный регистр, то этот разряд всегда равен 0.	RC	0
6	RBF	Receiver Buffer Full. Буфер приемника полон (и приемник включен). Если приемник закончит прием еще одного слова данных, это приведет к ошибке Overrun Error.	RC	0
5	OE	Overrun Error. С момента последнего чтения ST произошла ошибка переполнения приемника, то есть стоп-бит был принят в момент, когда буфер приемника был полон, в результате чего принятое слово было утеряно.	RC	0
4	RTO	Receiver Timeout. С момента последнего чтения ST произошел тайм-аут приемника, то есть приемник не зафиксировал старт-бита после приема последнего слова количество периодов baud_rate_clock, заданное в разрядах TV регистра CFG.	RC	0
3	BD	Break Detected. С момента последнего чтения ST обнаружен сигнал break, то есть линия gx удерживалась в 0 в течение времени, большего времени передачи посылки (иными словами, принята посылка, состоящая только из нулей, даже в стоп-бите). При этом приемник записывает в FIFO-буфер одно слово из восьми нулей, вне зависимости от того как долго держится 0 на gx. Прием следующей посылки начнется только после перехода линии в 1, а затем приема старт-бита. Сигнал break не считается ошибкой стоп-бита.	RC	0
2	PE/AM	Parity Error/Address Match. С момента последнего чтения ST была зафиксирована ошибка четности/ совпадение адреса в 9-бит режиме. Значение этого бита зависит от режима работы uart. Во всех режимах работы, кроме 9-бит режима, значение этого бита - ошибка четности, то есть содержимое бита четности принятой посылки не соответствовало биту, сформированному приемником для проверки в соответствии	RC	0

Биты	Название	Описание	Доступ	Сброс
		с выбранным типом контроля четности (смотри CFG, тип контроля четности). В 9-бит режиме значение этого бита - совпадение адреса, то есть приемник принял адрес (посылка с 1 вместо бита четности), совпадающий с одним из адресов, заданных в регистре NBADDR. При этом сравниваются только разряды адреса, маскированные в NBMSK.		
1	FE	Frame Error. С момента последнего чтения ST была зафиксирована ошибка стоп-бита, то есть в принятой посылке значение стоп-бита было равно 0. Этот бит не устанавливается в 1, если все остальные биты слова также равны 0, так как это воспринимается как сигнал break.	RC	0
0	CTSIC	Clear To Send Input Change. С момента последнего чтения ST произошло изменение сигнала CTS <sub>n</sub> . Это событие не возникает, если передача запрещена, чтобы избежать ложной генерации этого события после сброса.	RC	0



## Примечание

Биты BD, PE/AM, FE при приеме слова выставляются в 1 даже в случае, если в буфере приемника не было места, чтобы сохранить принятое слово.

## SPW (SpaceWire)

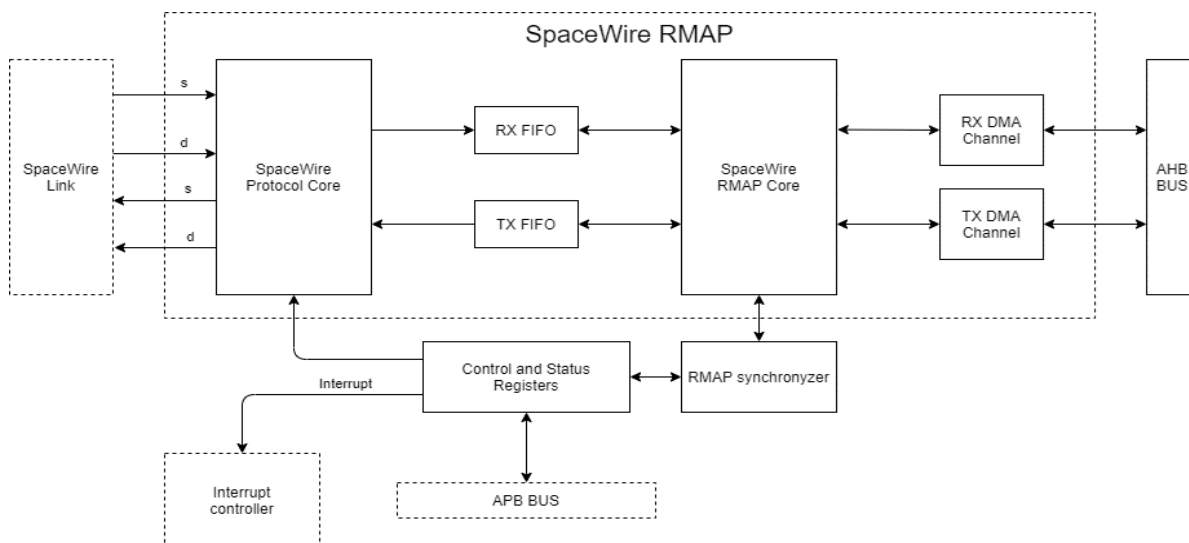
Контроллер интерфейса SpaceWire предназначен для обеспечения аппаратной поддержки функций внутрисистемных коммуникаций с использованием протокола SpaceWire.

Основные особенности: контроллер разработан в соответствии с международными стандартами ECSS-E-ST-50-52C и ECSS-E-50-12C; до 47 байт размер принимаемых данных в пакетах с проверкой CRC; работа счетчиков маркеров времени и их логическая обработка осуществляется на частоте  $F_{apb}$ ; работа счетчиков пакетов прерываний\подтверждений прерываний и их логическая обработка осуществляется на  $F_{apb}$ .

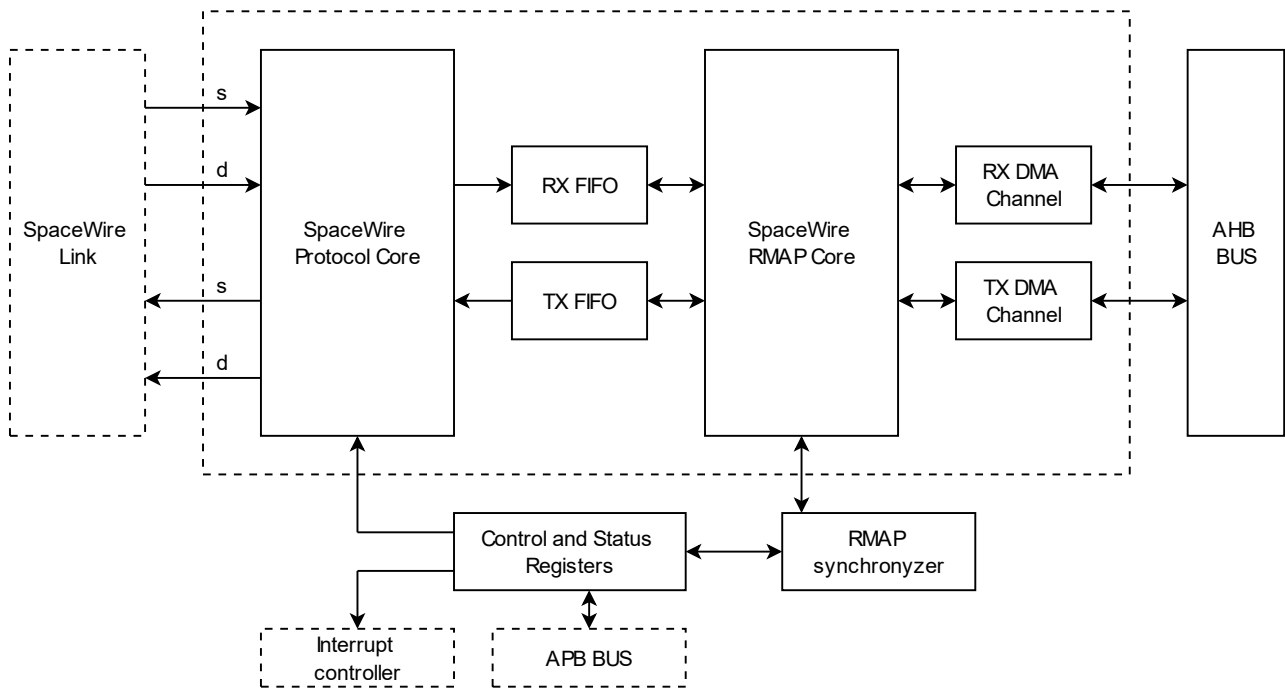
### Основные особенности

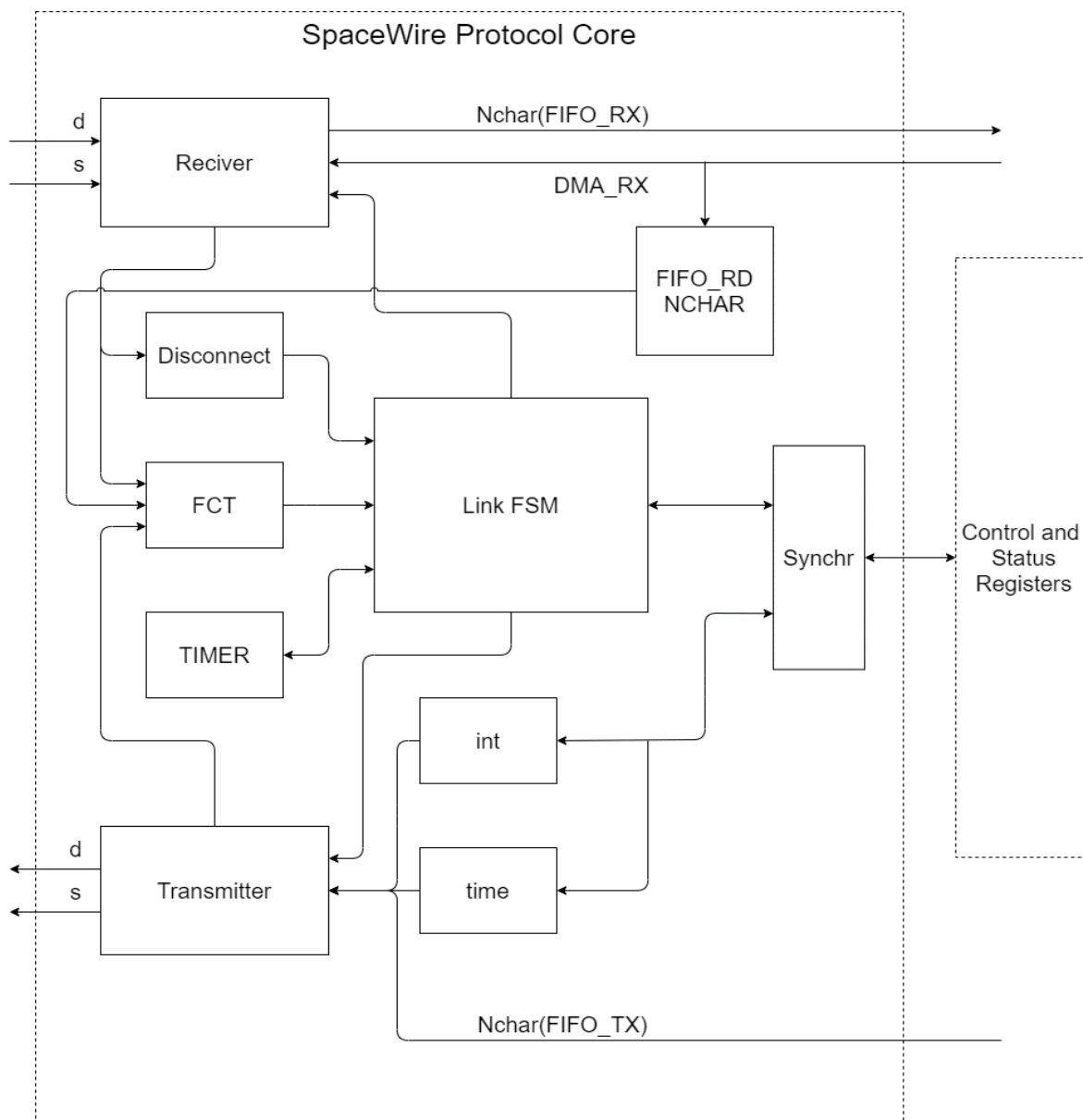
- контроллер разработан в соответствии с международным стандартом ECSS-E-ST-50-52C;
- контроллер разработан в соответствии с международным стандартом ECSS-E-50-12C;
- до 47 байт размер принимаемых данных в пакетах с проверкой CRC;
- работа счетчиков маркеров времени и их логическая обработка осуществляется на частоте  $F_{apb}$ ;
- работа счетчиков пакетов прерываний\подтверждений прерываний и их логическая обработка осуществляется на  $F_{apb}$ .

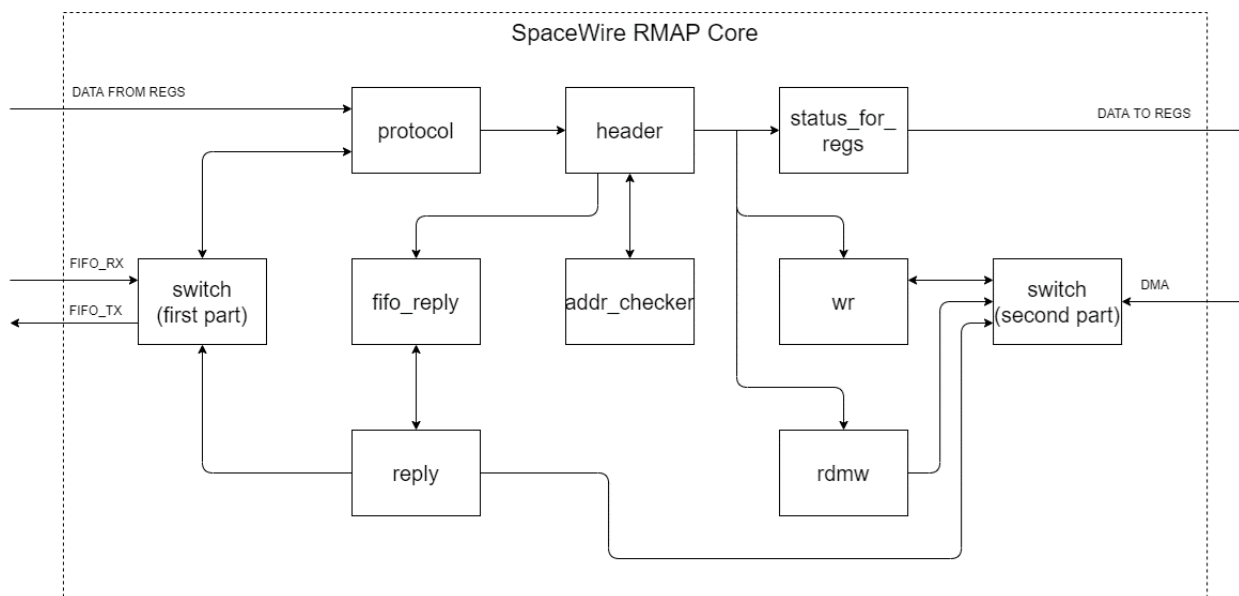
### Структурная схема











### Описание основных блоков

- SpaceWire Protocol Core - основной модуль управления spacewire
- RX\_FIFO - буфер для принятых пакетов данных
- TX\_FIFO - буфер для передаваемых пакетов данных
- RX\_DMA Channel - канал DMA для принятых пакетов данных
- TX\_DMA Channel - канал DMA для передаваемых пакетов данных
- Control and Status Register - регистровый модуль
- Receiver - модуль декодирования принятых данных из сети
- Disconnect - модуль фиксации разрыва связи
- FCT - модуль управления символами FCT
- Timer - модуль подсчета времени переходов между состояниями
- Transmitter - модуль управления передаваемыми данными
- Link FSM - модуль управления состояниями spacewire
- int - модуль управления прерываниями и подтверждениями
- time - модуль управления временными метками
- FIFO\_RD\_NCAR - модуль синхронизации сигнала чтения данных DMA
- Synchr - модуль синхронизации данных между доменами ip-модуля и регистрового модуля
- switch - модуль переключения потока данных между DMA и внутренними модулями SPW
- protocol - модуль определения текущего протокола
- header - модуль управления головной частью пакета rmap
- addr\_checker - модуль управления доступными адресами rmap
- fifo\_reply - fifo-буфер для формирования головной части ответного слова rmap

- reply - модуль управления ответным словом rmap
- status\_for\_regs - модуль статусов rmap
- wr - модуль обработки команды записи rmap
- rdmw - модуль обработки команды чтения и чтения-модификации-записи

### Установка скорости передачи данных

Управление скоростью передачи осуществляется посредством регистра SPW\_DIV. При установке соединения на внутренний делитель подается коэффициент DIV\_10MGz из разрядов регистра SPW\_DIV. Полученная частота при делении должна соответствовать 10 Мбит/с. После установки соединения на делитель будет подаваться коэффициент DIV\_TX. При разрыве соединения переход на коэффициент DIV\_10MGz выполняется автоматически, при повторной установке соединения переход на DIV\_TX так же выполняется автоматически.

### Маркеры времени

Маркеры времени – системная функция стандарта SpaceWire. Они предназначены для синхронизации системных часов взаимодействующих систем. При передаче данных маркеры времени имеют наивысший приоритет. Передача маркера времени в сеть может осуществляться двумя способами - автоматический и вручную. Счетчик автоматической отправки работает на Fapb.

Для автоматической отправки пользователю необходимо записать значение в регистр SPW\_TIME\_CNT, который определяет через сколько тактов частоты APB будет осуществлена отправка маркера времени в сеть. Затем нужно установить биты EN\_TIME и AUTOTIME. После данных действий маркер времени будет автоматически отсылаться в сеть. Первый маркер времени при автоматической отправки будет выдан через SPW\_TIME\_CNT + 3 или 4 такта частоты APB относительно перехода в состояние RUN. Погрешность в 1 такт APB вызвана синхронизацией. Значение первого отправленного маркера равно 1.

Для ручной отправки маркера времени необходимо записать требуемое значение времени в регистр SPW\_TIME поле TIME\_TX. Затем необходимо записать в этот же регистр бит TIME\_TX\_EN. Маркер из поля TIME\_TX записывается в поле текущего времени сети TIME\_NOW и готовится к отправке. SPW дожидается окончания передачи символа данных или служебного символа и начинает передачу маркера времени, после окончания передачи маркера времени продолжается передача потока данных. После отправки маркера времени выставляется статус TIME\_SENT.

В канале приема маркер времени выделяется из потока данных и при безошибочном приеме заносится в поле TIME\_RX с выставлением статуса GOTTIME\_GOOD, если маркер времени является корректным. Корректным признается маркер времени на 1 больше, чем предыдущий (значение которого отображается в регистр в TIME\_NOW). Если предыдущий маркер времени имел значение 63, то следующий корректный маркер времени должен иметь значение 0. Значение маркера времени заносится в TIME\_RX и TIME\_NOW. Если маркер времени не является корректным, то его значение так же заносится в TIME\_RX и TIME\_NOW, но в статусном регистре выставляется бит GOTTIME\_BAD. В начале работы устройства или после сброса маркер времени со значением 1 рассматривается как корректный. Значение текущего времени сети отображается в регистре SPW\_TIME поле TIME\_NOW

### Работа с пакетами прерываний и подтверждений прерываний

SPW поддерживает работу с пакетами прерываний и подтверждений прерываний. Пользователь может разрешить работы с пакетами прерываний путем записи «1» в EN\_INT. Пользователь может разрешить работы с пакетами подтверждений прерываний путем записи «1» в EN\_ACK. Работа

счетчиков системы прерываний происходит на частоте APB.

### Работа с пакетами прерываний

За фиксацию пакетов прерываний системы отвечает регистр SPW\_INTPACK. Регистр SPW\_INTPACK содержит информацию о принятых и отправленных кодах распределенных прерываний и подтверждения. Если из сети получено распределенное прерывание, то бит регистра SPW\_INTPACK, соответствующий номеру распределенного прерывания устанавливается в 1 (если он уже не был установлен в 1). Аналогично, если в регистр SPW\_INT\_ACK осуществляется запись кода распределенного прерывания, соответствующий бит регистра SPW\_INTPACK устанавливается в 1. Если из сети получен код подтверждения, то бит регистра SPW\_INTPACK, соответствующий номеру кода подтверждения устанавливается в 0 (если он уже не был установлен в 0). Аналогично, если в регистр SPW\_INT\_ACK осуществляется запись кода подтверждения, соответствующий бит регистра SPW\_INTPACK устанавливается в 0. Необходимость данного регистра связана с тем, что коды распределенных прерываний и коды подтверждения могут приходиться из сети очень часто, быстрее, чем процессор может среагировать на очередное прерывание и прочесть код. Если даже в регистре SPW\_INT\_ACK код распределенного прерывания или код подтверждения будет перезаписан следующим, информация о нем не будет утрачена – она сохранится в регистре SPW\_INTPACK. Для того чтобы его сбросить, необходимо записать в этот разряд регистра SPW\_INTPACK 1. (При записи в бит значения 0, его значение не меняется).

### Игнорирование пакетов прерываний

Пакеты прерываний будут отброшены при приеме в случае, если прерывание с данным номером замаскировано в регистре SPW\_MSK\_INT.

Пакет прерывания будет отброшен при передаче в случае, если была попытка отправки при выставленном сигнале INT\_BLOCK регистра SPW\_INT\_ACK. Если пакет прерывания был отправлен с номером прерывания, на который не пришел пакет подтверждения прерывания, что можно увидеть в регистре SPW\_ACK\_NOT. Также в случае попытки отправки прерывания до состояния «RUN» системы прерывание будет отброшено. Не стоит пытаться отправлять прерывания в любом состоянии помимо «RUN». В случае отмены отправки пакета прерывания в регистре SPW\_ST будет выставлен бит NOTWRINT.

### Работа с пакетами подтверждений

Если в регистре SPW\_INTPACK регистрируется код распределенного прерывания и работа с пакетами подтверждений для данного прерывания разрешена в SPW\_MSK\_ACK, то для него запускается счет таймута (каждому разряду SPW\_INTPACK соответствует отдельный счетчик). В зависимости от того, был ли код распределенного прерывания принят из сети или отправлен процессором, начальное значение счетчика устанавливается в LOC\_CNT1 или LOC\_CNT2. (значение счетчика декрементируется каждый раз, когда глобальный счетчик досчитывает до определенного для него максимального значения). Если за время счета из сети не поступает соответствующий код подтверждения, то соответствующий разряд регистра SPW\_ACK\_NOT устанавливается в 1. Для того чтобы его сбросить, необходимо записать в этот разряд регистра SPW\_ACK\_NOT 1. (При записи в бит значения 0, его значение не меняется).

### Игнорирование пакетов подтверждений прерываний

Пакеты прерываний будут отброшены при приеме в случае, если прерывание с данным номером замаскировано в регистре SPW\_MSK\_ACK.

Пакет подтверждения прерывания будет отброшен при передаче в случае, если была попытка отправки при выставленном сигнале ACK\_BLOCK регистра SPW\_INT\_ACK. Также в случае попытки отправки прерывания до состояния «RUN» системы подтверждение прерывания будет отброшено. Не стоит пытаться отправлять пакет в любом состоянии помимо «RUN». В случае попытки ручной отправки пакета подтверждения прерывания на прерывание, на которое настроена автоматическая отправка подтверждения, пакет также будет проигнорирован.

Прямой доступ к памяти (DMA)

Принятые и передаваемые SpaceWire-пакеты хранятся не в самом IP-ядре SpaceWire (по причине их большого размера), а в основной памяти системы. В IP-ядре имеется два канала прямого доступа к памяти (DMA): TX и RX. Каналы работают независимо друг от друга и могут функционировать одновременно. Каждый канал является мастером на системной шине.

В ходе приема пакета приемник помещает принятые байты в RX FIFO буфер (количество элементов в FIFO задается параметром RX\_FIFO\_ADDR\_W), откуда их вычитывает RX канал DMA и записывает в основную память.

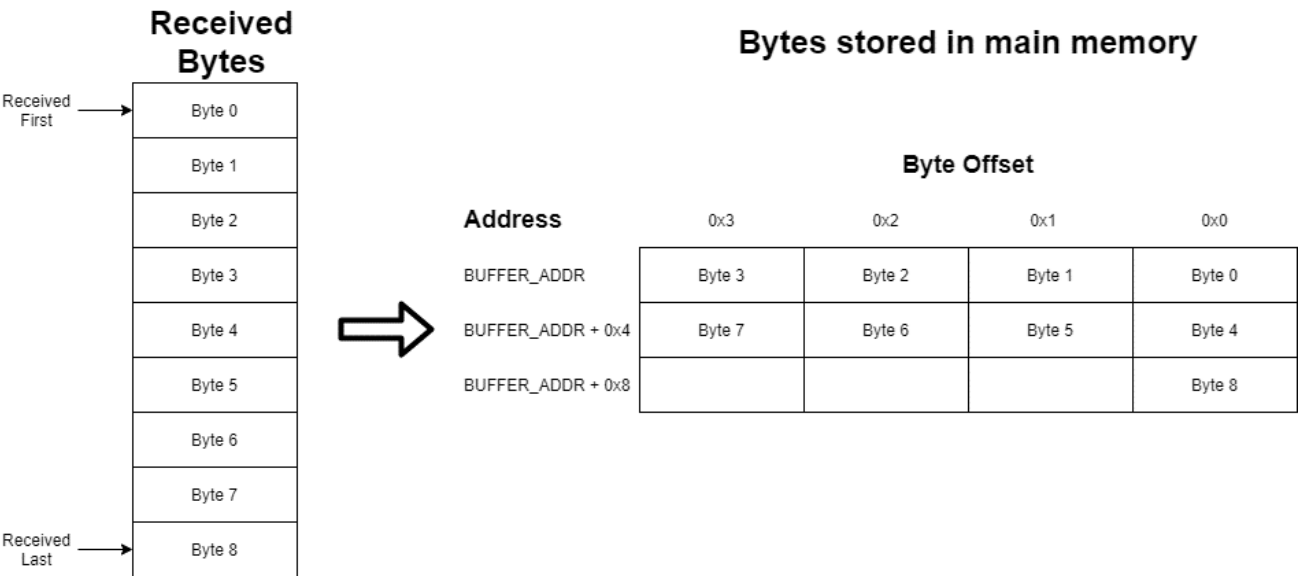
В ходе передачи пакета TX канал DMA вычитывает байты пакета из буфера в основной памяти через системную шину и помещает их в TX FIFO (количество элементов в FIFO задается параметром TX\_FIFO\_ADDR\_W). Байты извлекаются из FIFO передатчиком и передаются в канал SpaceWire.

Работа каналов DMA управляется дескрипторами - специальными структурами данных, которые также находятся в основной памяти. Каждый дескриптор содержит адрес и размер буфера данных для пакета, а также набор управляющих сигналов и флагов. Дескрипторы сгруппированы в две таблицы. DMA RX канал управляется таблицей RX дескрипторов, DMA канал TX управляется таблицей TX дескрипторов. Количество дескрипторов в каждой из таблиц задается параметром IP-ядра NUM\_DESCR (равен 16). Каждый дескриптор занимает 12 байт. Таким образом, каждая таблица дескрипторов занимает 12\*NUM\_DESCR байт основной памяти. Таблицы занимают непрерывные участки памяти, то есть дескрипторы в каждой таблице должны идти один за другим, без незанятых промежутков.

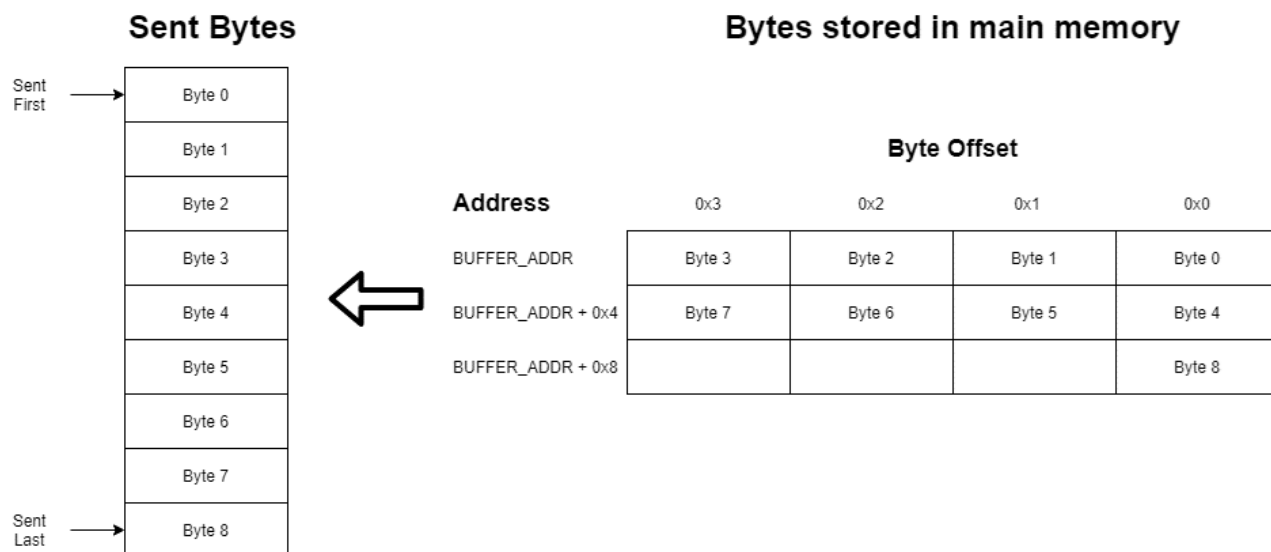
Перед началом работы интерфейса SpaceWire необходимо выделить место в основной памяти под таблицы дескрипторов и буферы данных и инициализировать дескрипторы. Также необходимо сообщить IP-ядру SpaceWire адреса таблиц, записав адрес таблицы TX дескрипторов в регистр SPW\_TX\_DESCR\_ADDR, а адрес таблицы RX дескрипторов в регистр SPW\_RX\_DESCR\_ADDR. Запрещено менять SPW\_TX\_DESCR\_ADDR во время работы TX канала DMA и запрещено менять SPW\_RX\_DESCR\_ADDR во время работы RX канала DMA.

Порядок байтов в памяти

Принятые байты сохраняются в 32-битной основной памяти в порядке Little-endian. Это означает, что первый принятый байт сохраняется по адресу BUFFER\_ADDR, второй - по адресу BUFFER\_ADDR + 0x1 и так далее:



Аналогичный порядок байтов используется и при передаче: первым на линию SPW передается байт, находящийся по адресу BUFFER\_ADDR, затем байт, находящийся по адресу BUFFER\_ADDR + 0x1 и так далее:



### Работа DMA каналов

Каждый канал DMA активируется записью в соответствующий бит регистра управления (START\_TX\_DMA / START\_RX\_DMA). При этом бит EN в регистре управления должен быть выставлен в 1, иначе каналы нельзя активировать (можно включить блок SPW в целом и активировать канал(ы) DMA за одну запись в регистр управления). При выключении блока записью 0 в бит EN каналы прекращают работу и сбрасываются в свое начальное состояние.

Сразу после записи в START\_TX\_DMA / START\_RX\_DMA выставляется в 1 бит статусного регистра TXDMA\_ACTIVE / RXDMA\_ACTIVE. TX канал DMA после включения немедленно обрабатывает нулевой дескриптор. RX канал DMA сначала ждет появления данных в RX FIFO, и затем обрабатывает нулевой дескриптор. Обработка дескриптора происходит следующим образом:

1. Канал DMA вычитывает дескриптор из памяти.

2. TX канал DMA вычитывает заданное в дескрипторе количество байтов из указанного в дескрипторе буфера данных и записывает их в TX\_FIFO. RX канал DMA вычитывает байты из RX\_FIFO и помещает их в указанный в дескрипторе буфер данных, пока либо не вычитает из FIFO EOP, либо количество байтов не превысит размер буфера, указанный в дескрипторе.

3. Канал DMA обновляет второе 32-битное слово дескриптора, выставляя бит EN в 0 и бит DONE в 1, чтобы показать, что этот дескриптор обработан.

Закончив обработку нулевого дескриптора (отправив/получив пакет в его буфер данных), канал DMA переходит к первому дескриптору и так далее, пока не дойдет до конца таблицы или не закончит обрабатывать дескриптор с битом WRAP выставленным в 1. После этого канал снова перейдет к нулевому дескриптору. Если при вычитке дескриптора оказывается, что у бит EN в нем не выставлен в 1, то канал DMA останавливает работу и выставляется бит TXDMA\_NO\_DESCR / RXDMA\_NO\_DESCR в статусном регистре. При этом бит статусного регистра TXDMA\_ACTIVE / RXDMA\_ACTIVE переходит в 0. После этого программа может инициализировать дескрипторы и запустить канал снова записью в START\_TX\_DMA / START\_RX\_DMA. В этом случае канал продолжит работу с того дескриптора, на котором остановился (у которого бит EN не был выставлен в 1).

### Дескрипторы

#### Первое 32-битное слово дескриптора

Биты	Описание
31:2	BUFFER_ADDR - старшие 30 битов адреса буфера данных. Младшие 2 бита адреса зарезервированы и всегда равны 0. Для TX дескрипторов именно из этого буфера DMA вычитает пакет для передачи в канал. Для RX дескрипторов именно в этот буфер будет записан принятый пакет.



Биты	Описание
1:0	Резерв

**Второе 32-битное слово дескриптора**

Биты	Описание
31	TRUNC - этот бит актуален только для RX дескрипторов. Если бит выставлен в 1, то при приеме пакета в этот дескриптор произошло переполнение буфера в основной памяти. Окончание пакета будет принято в следующий дескриптор.
30	RMAP - значение этого бита отличается для RX и TX дескрипторов. Для RX дескрипторов этот бит выставляется в 1 каналом DMA, если этот пакет - ответ на команду RMAP. Для TX дескрипторов этот бит необходимо выставить программно в 1 если этот пакет - команда RMAP.
29	EOP - бит актуален только для RX дескрипторов. По окончании работы с RX дескриптором канал DMA выставляет этот бит в 1, если после данных в этом пакете идет EOP.
28	EOP - бит актуален только для RX дескрипторов. По окончании работы с RX дескриптором канал DMA выставляет этот бит в 1, если после данных в этом пакете идет EOP.
27	DONE - должен быть выставлен в 0 программой при инициализации дескриптора. Выставляется в 1 каналом DMA после окончания работы с этим дескриптором.
26	IE - если выставлен в 1 при инициализации дескриптора, то после обработки этого дескриптора IP-ядро выставит в 1 бит TXDMA_DESCR_DONE/RXDMA_DESCR_DONE регистра ST и сформирует прерывание (прерывание также должно быть разрешено в регистре маски прерываний).
25	WRAP - если выставлен в 1 при инициализации дескриптора, то после обработки этого дескриптора канал DMA вернется в начало таблицы дескрипторов и продолжит работать с дескриптором номер 0
24	EN - должен быть выставлен в 1 при инициализации дескриптора, чтобы показать, что этот дескриптор инициализирован. Бит EN выставляется в 0 каналом DMA когда работа с этим дескриптором закончена (пакет отправлен/принят). До этого момента данный дескриптор «принадлежит» DMA и не должен быть изменен программно.
23:0	PACKET_SIZE - размер пакета. Это поле заполняется при инициализации дескриптора, однако его значение отличается для RX и TX дескрипторов. Для TX дескрипторов PACKET_SIZE - это количество байт, которое канал DMA должен вычитать из буфера данных и отправить в канал SpaceWire. Для RX дескрипторов PACKET_SIZE - это максимальный размер буфера данных (максимальный размер сообщения, которое может быть принято в этот RX дескриптор). После завершения приема канал DMA обновляет это поле, записывая сюда количество байт, принятых в этот дескриптор.

**Третье 32-битное слово дескриптора****RX**

Биты	Описание
31:24	Transaction Identifie (MS) - старшая часть номера транзакции принятого ответного пакета RMAP, полученная при обработке пакета. Это поле актуально только для RX дескрипторов и только в том случае, когда бит REPLY выставлен в 1 (то есть этот пакет является ответом на команду RMAP).
23:16	Transaction Identifie (LS) - младшая часть номера транзакции принятого ответного пакета RMAP, полученная при обработке пакета. Это поле актуально только для RX дескрипторов и только в том случае, когда бит REPLY выставлен в 1 (то есть этот пакет является ответом на команду RMAP).
15	ERR_DATA - ошибка данных принятого ответного пакета RMAP, полученная при обработке пакета. Это поле актуально только для RX дескрипторов и только в том случае, когда бит REPLY выставлен в 1 (то есть этот пакет является ответом на команду RMAP).
14	ERR_HR - ошибка заголовка принятого ответного пакета RMAP, полученная при обработке пакета. При получении данной ошибки поле STATUS_REPLY может быть ошибочно,



Биты	Описание
	правильность поля STATUS_REPLY зависит от значения STATUS_CODES_RX, которое определено стандартом. Это поле актуально только для RX дескрипторов и только в том случае, когда бит REPLY выставлен в 1 (то есть этот пакет является ответом на команду RMAP).
13:8	STATUS_CODES_RX - статус принятого ответного пакета на команду RMAP, полученный при обработке пакета. Это поле актуально только для RX дескрипторов и только в том случае, когда бит REPLY выставлен в 1 (то есть этот пакет является ответом на команду RMAP).
7:0	STATUS_REPLY - статус ответа на команду RMAP, содержащийся непосредственно в принятом пакете. Это поле актуально только для RX дескрипторов и только в том случае, когда бит REPLY выставлен в 1 (то есть этот пакет является ответом на команду RMAP).

## TX

Биты	Описание
31:6	Резерв
5:0	"Target SpaceWire Address" SIZE - размер (в байтах) адреса назначения команды RMAP. Это поле актуально только для TX дескрипторов и только в том случае, когда бит RMAP выставлен в 1 (то есть этот пакет является командой RMAP). Заполняется при инициализации дескриптора.



### Особенности реализации

При приёме ответного слова на операцию «Чтение» может быть переполнен RX FIFO (47 байт). В данном случае пакет будет отброшен, а в поле STATUS\_CODES\_RX будет выставлен код ошибки 9.

Если при приеме RMAP REPLY происходит переполнения буфера в основной памяти (при этом во втором слове дескриптора выставляется бит TRUNC), то третье слово дескриптора будет содержать нули. Третье слово дескриптора будет содержать валидные данные в следующем дескрипторе (куда будет принят остаток пакета).

## Переполнение буфера данных при приеме

В каждом RX-дескрипторе жестко задан размер буфера для принятых данных (PACKET\_SIZE), тогда как стандарт SpaceWire не ограничивает размер пакета данных. Таким образом, возможна ситуация, когда размер принимаемого пакета окажется больше PACKET\_SIZE. В этом случае DMA перейдет к следующему на очереди RX дескриптору и продолжит принимать пакет в его буфер данных. При этом в текущем дескрипторе будет выставлен бит TRUNC - именно так можно определить, что продолжение пакета находится в следующем RX дескрипторе. Таким образом, размер пакета, который способен принять контроллер SpaceWire, не ограничен размером одного буфера данных.

## Ошибки на системной шине

Если в ходе чтения/записи основной памяти каналом DMA происходит ошибка, то этот канал DMA останавливается и выставляется бит статусного регистра TXDMA\_BUS\_ERROR для TX канала DMA или RXDMA\_BUS\_ERROR для RX канала DMA. Бит статусного регистра TXDMA\_ACTIVE / RXDMA\_ACTIVE при этом сбрасывается в 0. Канал сбрасывается в свое начальное состояние, при последующем запуске он начнет работы с дескриптора номер 0.

## **RMAP**

### **Пакеты RMAP**

Устройство поддерживает все операции RMAP. Ответ на команды RMAP формируется автоматически в соответствии со стандартом.

Прием служебной информации и данных ответных пакетов при отсутствии ошибок, а также при ошибках типа ERR\_SEND\_HR осуществляется в дескрипторы приемника. Служебная информация записывается в регистры дескрипторов в первые 3 слова, в соответствии с описанием, представленным ранее.

Биты в регистре статуса RMAP формируются после отправки ответного слова на команду. Если ответное слово не требовалось, то статусные биты формируются сразу после обработки команды. Статусные биты формируются сразу после обработки ответного пакета.

### **Доступ к основной памяти**

RMAP использует RX канал DMA для записи данных в основную память и TX канал DMA для чтения данных из основной памяти. В ходе конфигурации SpaceWire можно задать до 4х диапазонов адресов, в которые RMAP разрешен доступ.

Диапазоны адресов, в которые RMAP разрешен доступ, задаются записью в регистры SPW\_RMAP\_REGION\_BASE\_ADDR и SPW\_RMAP\_REGION\_SIZE. Всего есть 4 пары этих регистров, каждая пара задает свой диапазон адресов, куда RMAP разрешен доступ: от SPW\_RMAP\_REGION\_BASE\_ADDR (включительно) до SPW\_RMAP\_REGION\_BASE\_ADDR + SPW\_RMAP\_REGION\_SIZE (не включительно). Для каждого диапазона можно разрешить чтение (бит RD\_EN) и запись (WR\_EN) отдельно. По-умолчанию все диапазоны адресов выключены (оба бита RD\_EN и WR\_EN выставлены в 0), а значит RMAP не имеет доступа в основную память системы. Диапазоны адресов, в которые RMAP разрешен доступ, задаются записью в регистры SPW\_RMAP\_REGION\_BASE\_ADDR и SPW\_RMAP\_REGION\_SIZE. Всего есть 4 пары этих регистров, каждая пара задает свой диапазон адресов, куда RMAP разрешен доступ: от SPW\_RMAP\_REGION\_BASE\_ADDR (включительно) до SPW\_RMAP\_REGION\_BASE\_ADDR + SPW\_RMAP\_REGION\_SIZE (не включительно). Для каждого диапазона можно разрешить чтение (бит RD\_EN) и запись (WR\_EN) отдельно. По-умолчанию все диапазоны адресов выключены (оба бита RD\_EN и WR\_EN выставлены в 0), а значит RMAP не имеет доступа в основную память системы.

Запись в регистры, управляющие доступом DMA к памяти, должна происходить перед началом работы RMAP. Во время работы RMAP менять значения этих регистров запрещено.

### **Ограничение на адреса, указанные в командах RMAP**

Адрес, указанный в команде RMAP, должен быть выровнен на границу 32-битного слова (то есть младшие 2 бита должны быть равны 0).

### **CRC принятого пакета**

Размер поля DATA принимаемого пакета, в котором требуется проверка данных (crc), не должен превышать 47 байт.

### **Ошибки в пакетах RMAP**

При приеме пакета RMAP определяются ошибки по header и data. Ошибки по header делятся на 2 категории: ERR\_NSEND\_HR и ERR\_SEND\_HR.

К ошибкам ERR\_NSEND\_HR относятся ошибки, на которые по стандарту не нужно отправлять ответные слова. Это ошибки, связанные с отсутствием или несовпадением поля crc, а также ошибка по неизвестному типу пакета. Такие пакеты будут отброшены устройством, так как считаются полностью не валидными. В регистре статуса RMAP будут корректно выставлены биты ERR\_NSEND\_HR, ERR\_HR и CODES, остальные биты значения не имеют.

К ошибкам ERR\_SEND\_HR относятся ошибки, по которым допустимо отправлять ответный пакет в случае, если был на него запрос. Все данные в регистре статуса RMAP будут валидны. Если пакет

представляет собой ответный пакет на команду, то для него также возможно выставление данного бита. Это будет значить, что все биты регистра статуса валидны. Вся информация по данному пакету зафиксирована в дескрипторе приемника в соответствии с описанием.

Фиксация ошибок по данным реализована для всех типов пакетов (команды и ответные пакеты) в соответствии со стандартом.



#### Особенности

При недопустимом значении в поле «Data Length» операции read-modify-write в ответном слове будет отправлено то же значение «Data Length», что и в пакете команды. В остальных случаях значение будет поделено на 2.

### Порядок работы со SPW

- записать необходимые коэффициенты в регистр **SPW\_DIV**;
- записать необходимые коэффициенты в регистр **SPW\_PAUSE**;
- разрешить необходимые прерывания в регистре **SPW\_MSK**;
- настроить модуль на нужный тип работы, путём записи в регистр **SPW\_CFG**;
- настроить базовый адрес таблицы дескрипторов для принимаемых пакетов путём записи значения в регистр **SPW\_RX\_DESCR\_ADDR**;
- настроить первое и второе слово дескриптора в соответствии с требованиями;
- настроить базовый адрес таблицы дескрипторов для передаваемых пакетов путём записи значения в регистр **SPW\_TX\_DESCR\_ADDR**;
- настроить первое, второе и третье (для RMAP) слово дескриптора в соответствии с требованиями;
- разрешить работу модуля и DMA путем записи в регистр **SPW\_CTRL**.

### Формат пакета RMAP для передачи

Для отправки команды гтар пользователь должен сформировать пакет требуемого формата в области памяти, на которую указывает соответствующий дескриптор.

Формат команды Write RMAP:

1. Target SpaceWire Address(при необходимости)
2. Target Logical Address
3. Protocol ID
4. Write command options
5. Key
6. Reply Address (при необходимости)
7. Initiator Logical Address
8. Transaction Identifier
9. Extended Address
10. Memory address

11. Data Length

12. Data

В слове «Data Length» указывается количество записанных данных «Data». CRC для «Header» и для «DATA» будет посчитан и вставлен автоматически при отправке команды.

Формат команды Read RMAP:

1. Target SpaceWire Address(при необходимости)

2. Target Logical Address

3. Protocol ID

4. Read command options

5. Key

6. Reply Address

7. Initiator Logical Address

8. Transaction Identifier

9. Extended Address

10. Memory address

11. Data Length

CRC для «Header» будет посчитан и вставлен автоматически при отправке команды.

Формат команды Read-Modify-Write RMAP:

1. Target SpaceWire Address(при необходимости)

2. Target Logical Address

3. Protocol ID

4. Read-modify-write command options

5. Key

6. Reply Address

7. Initiator Logical Address

8. Transaction Identifier

9. Extended Address

10. Memory address

11. Data Length

12. Data

13. Mask

В слове "Data Length" указывается количество записанных данных "Data" и "Mask". CRC для "Header" и для "Data"+ "Mask" будет посчитан и вставлен автоматически при отправке команды.

#### **Тестовый режим работы**

Работа модуля может быть проверена в тестовом режиме. Для этого пользователю необходимо записать «1» в бит DSLOOPBACK. После этого выходы DS модуля SPW будут соединены с входами DS этого же модуля. Таким образом модуль будет взаимодействовать сам с собой.

### Частотные ограничения

Ряд модулей SPW работает на частоте  $F_{spw}$ . Данная частота определяет максимальную скорость передачи данных. Для того, чтобы осуществлять обмен на скорости 200 Мб/с,  $F_{spw}$  должна быть 200 МГц. При уменьшении  $F_{spw}$  максимальная скорость передачи будет уменьшаться прямо пропорционально.

На основе  $F_{spw}$  формируются частоты с коэффициентами деления из  $DIV\_TX$  и  $DIV\_10MGz$ . Скорость обмена в 10 Мб/с обязательна для стандарта SpaceWire, соответственно  $F_{spw}$  не может быть ниже 10 МГц. В случае 10 МГц на  $F_{spw}$  в регистр  $SPW\_DIV$  поля  $DIV\_10MGz$  должен быть записан коэффициент «0», что будет соответствовать скорости передачи данных в 10 Мб/с.

Обработка маркеров времени производится на  $F_{apb}$ . Это вызывает частотные ограничения. Один маркер времени при непрерывном потоке должен быть передан медленнее одного такта  $F_{apb}$ . Иначе маркеры времени не будут успевать вычитываться системой. Между приемником, работающим на  $F_{spw}$ , и доменом  $F_{apb}$  стоит буфер на 8 слов, чтобы компенсировать начальную задержку приема маркеров в домен  $F_{apb}$ .

Обработка пакетов прерываний и подтверждений прерываний производится на  $F_{apb}$ . Это вызывает частотные ограничения. Один пакет при непрерывном потоке должен быть передан медленнее одного такта  $F_{apb}$ . Иначе прерывания/подтверждения прерываний не будут успевать вычитываться системой. Между приемником, работающим на  $F_{spw}$ , и доменом  $F_{apb}$  стоит буфер на 8 слов, чтобы компенсировать начальную задержку приема прерываний/подтверждений прерываний в домен  $F_{apb}$ .

С учетом того, что broadcast пакет (прерывание/подтверждение/маркер времени) длится 14 битов, то при скорости 200 Мб/с **Fapb** должна быть не ниже  $200/14=14.29$  МГц



#### Предупреждение

Для корректной работы контроллера не рекомендуется тактировать шины AHB и APB0 различными по частоте синхросигналами.

### Карта регистров

Смещение от базового адреса блока	Аббревиатура	Доступ	Описание
0x00	SPW_CTRL	RW	Регистр управления
0x04	SPW_CFG	RW	Регистр конфигурации
0x08	SPW_MSK	RW	Регистр маски
0x0C	SPW_ST	R	Регистр статуса
0x10	SPW_DIV	RW	Регистр делителя
0x14	SPW_PAUSE	RW	Регистр задержки
0x18	SPW_MSK_INT	RW	Регистр маски прерываний
0x1C	SPW_MSK_ACK	RW	Регистр маски подтверждений
0x20	SPW_WAITACK_CNT	RW	Регистр задержки подтверждений
0x24	SPW_INT_ACK	RW	Регистр прерываний и подтверждений

Смещение от базового адреса блока	Аббревиатура	Доступ	Описание
0x28	SPW_INTPACK	RW	Регистр зафиксированных прерываний
0x2C	SPW_ACK_NOT	RW	Регистр отсутствующих подтверждений
0x30	SPW_TIME_CNT	RW	Регистр времени отправки временных меток
0x34	SPW_TIME	RW	Регистр времени
0x38	SPW_TX_DESCR_ADDR	RW	Базовый адрес таблицы TX дескрипторов
0x3C	SPW_RX_DESCR_ADDR	RW	Базовый адрес таблицы RX дескрипторов
RMAP			
0x40	SPW_LADDR	R/W	Логический адрес устройства
0x44	SPW_KEY	R/W	Ключ авторизации устройства
0x48	SPW_ST_RMAP	R/W	Регистр статусов протокола RMAP
0x4C	SPW_MSK_INT_RMAP	R/W	Ключ авторизации устройства
0x50	SPW_RMAP_REGION_BASE_ADDR_0	R/W	Базовый адрес разрешенного региона памяти для RMAP
0x54	SPW_RMAP_REGION_BASE_ADDR_1	R/W	Базовый адрес разрешенного региона памяти для RMAP
0x58	SPW_RMAP_REGION_BASE_ADDR_2	R/W	Базовый адрес разрешенного региона памяти для RMAP
0x5C	SPW_RMAP_REGION_BASE_ADDR_3	R/W	Базовый адрес разрешенного региона памяти для RMAP
0x60	SPW_RMAP_REGION_EXTENDED_ADDR	R/W	Расширенный адрес разрешенных регионов памяти для RMAP
0x64	SPW_RMAP_REGION_SIZE_0	R/W	Размер разрешенного региона памяти для RMAP
0x68	SPW_RMAP_REGION_SIZE_1	R/W	Размер разрешенного региона памяти для RMAP
0x6C	SPW_RMAP_REGION_SIZE_2	R/W	Размер разрешенного региона памяти для RMAP
0x70	SPW_RMAP_REGION_SIZE_3	R/W	Размер разрешенного региона памяти для RMAP

**SPW\_CTRL [0x00].**

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3	START_TX_DMA_A	Запустить работу TX DMA(автоматически сбрасывается в 0): • 1 - запустить TX DMA	R/W	0
2	START_RX_DMA_A	Запустить работу RX DMA(автоматически сбрасывается в 0): • 1 - запустить RX DMA	R/W	0

Биты	Название	Описание	Доступ	Сброс
1	LINK_DIS	Остановка работы SPW: • 1 - работа SPW остановлена; • 0 - SPW в рабочем режиме.	R/W	0
0	EN	Включение модуля SPW: • 1 - включить модуль; • 0 - отключить модуль.	R/W	0

**SPW\_CFG [0x04].**

Биты	Название	Описание	Доступ	Сброс
31:9	-	Резерв	R	0
8	DSLOOPBACK	Соединение выходных линий DS со входными линиями DS SPW: • 1 - Соединить линии DS; • 0 - Стандартный режим.	R/W	0
7	RMAP_EN	Поддержка SPW_RMAP: • 1 - SPW с поддержкой протокола RMAP; • 0 - SPW без поддержки протокола RMAP	R/W	0
6	AUTOTIME	Разрешение автоматической отправки символов времени при инкрементации счетчика времени: • 1 - Разрешить автоматическую отправку; • 0 - Запретить автоматическую отправку.	R/W	0
5	EN_TIME	Разрешение работы с символами времени: • 1 - Разрешить работу; • 0 - Запретить работу.	R/W	0
4	EN_ACK	Разрешение отправки символов подтверждения прерывания(ack) при получении выбранных прерываний: • 1 - Разрешить отправку; • 0 - Запретить отправку.	R/W	0
3	EN_INT	Разрешение работы с символами прерываний: • 1 - Разрешить работу; • 0 - Запретить работу.	R/W	0
2	OLD_SPEC	Включение режима работы с устройствами, основанными на спецификации ECSS-E-ST-50-12C(31July2008): • 1 - работа по спецификации ECSS-E-ST-50-12C(31July2008); • 0 - работа по спецификации ECSS-E-ST-50-12C-Rev.1(15May2019)	R/W	0
1	AUTOSTART	Разрешение перехода в состояние «Started» при приеме символа «NULLS»: • 1 - Разрешить переход • 0 - Запретить переход.	R/W	0
0	LINK_START	Разрешение перехода в состояние «Started» без ожидания приема символа «NULLS»: • 1 - Разрешить переход; • 0 - Запретить переход.	R/W	0



## SPW\_MSK [0x08].

Биты	Название	Описание	Доступ	Сброс
31:29	-	Резерв	R	0
28	TXDMA_ACTIVE_MSK	Разрешить прерывание по статусу TXDMA_ACTIVE: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
27	RXDMA_ACTIVE_MSK	Разрешить прерывание по статусу RXDMA_ACTIVE: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
26	TXDMA_PACKET_ERROR_MSK	Разрешить прерывание по статусу TXDMA_PACKET_ERROR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
25	RXDMA_BUF_OVERFLOW_MSK	Разрешить прерывание по статусу RXDMA_BUF_OVERFLOW: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
24	TXDMA_NO_DESCR_MSK	Разрешить прерывание по статусу RXDMA_NO_DESCR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
23	RXDMA_NO_DESCR_MSK	Разрешить прерывание по статусу TXDMA_NO_DESCR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
22	TXDMA_BUS_ERROR_MSK	Разрешить прерывание по статусу TXDMA_BUS_ERROR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
21	RXDMA_BUS_ERROR_MSK	Разрешить прерывание по статусу RXDMA_BUS_ERROR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
20	TXDMA_DESCR_DONE_MSK	Разрешить прерывание по статусу TXDMA_DESCR_DONE: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
19	RXDMA_DESCR_DONE_MSK	Разрешить прерывание по статусу RXDMA_DESCR_DONE: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
18	TXDMA_PACKET_DONE_MSK	Разрешить прерывание по статусу TXDMA_PACKET_DONE: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
17	RXDMA_PACKET_DONE_MSK	Разрешить прерывание по статусу RXDMA_PACKET_DONE: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
16	NOTWRACK_MSK	Разрешить прерывание по статусу NOTWRACK: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0



Биты	Название	Описание	Доступ	Сброс
15	NOTWRINT_MSK	Разрешить прерывание по статусу NOTWRINT: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
14	NOTACK_MSK	Разрешить прерывание по статусу NOTACK: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
13	GOTACK_MSK	Разрешить прерывание по статусу GOTACK: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
12	GOTINT_MSK	Разрешить прерывание по статусу GOTINT: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
11	ACK_SENT_MSK	Разрешить прерывание по статусу ACK_SENT: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
10	INT_SENT_MSK	Разрешить прерывание по статусу INT_SENT: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
9	GOTTIME_BAD_MSK	Разрешить прерывание по статусу GOTTIME_BAD: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
8	GOTTIME_GOOD_MSK	Разрешить прерывание по статусу GOTTIME_GOOD: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
7	TIME_SENT_MSK	Разрешить прерывание по статусу TIME_SENT: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
6	CREDERR_MSK	Разрешить прерывание по статусу CREDERR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
5	ESCERR_MSK	Разрешить прерывание по статусу ESCERR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
4	DSCERR_MSK	Разрешить прерывание по статусу DSCERR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
3	PERR_MSK	Разрешить прерывание по статусу PERR: • 1 - прерывание разрешено • 0 - прерывание запрещено	R/W	0
2:0	-	Резерв	R	0

## SPW\_ST [0x0C].

Биты	Название	Описание	Доступ	Сброс
31:17	-	Резерв	R	0
28	TXDMA_ACTIV E	Признак активности TX канала DMA: <ul style="list-style-type: none"> <li>1 - канал работает;</li> <li>0 - канал не работает. Бит выставляется в 1, когда пользователь записывает 1 в бит START_TX_DMA и сбрасывается в 0, когда TX канал DMA вычитывает дескриптор с битом EN равным 0, когда происходит ошибка доступа к основной памяти TXDMA_BUS_ERROR, или когда весь блок SPW выключается записью 0 в бит EN регистра SPW_CTRL.</li> </ul>	R	0
27	RXDMA_ACTIV E	Признак активности RX канала DMA: <ul style="list-style-type: none"> <li>1 - канал работает;</li> <li>0 - канал не работает. Бит выставляется в 1, когда пользователь записывает 1 в бит START_RX_DMA и сбрасывается в 0, когда TX канал DMA вычитывает дескриптор с битом EN равным 0, когда происходит ошибка доступа к основной памяти RXDMA_BUS_ERROR, или когда весь блок SPW выключается записью 0 в бит EN регистра SPW_CTRL.</li> </ul>	R	0
26	TXDMA_PACKE T_ERROR	Выставляется в 1, когда в ходе передачи пакета в канал SpaceWire происходит ошибка и часть передаваемого на данный момент пакета не была отправлена в сеть	RC	0
25	RXDMA_BUF_ OVERFLOW	Выставляется в 1, когда в ходе приема пакета буфер данных переполняется (продолжение пакета будет записано в следующий дескриптор)	RC	0
24	TXDMA_NO_D ESCR	Выставляется в 1, когда TX канал DMA останавливает работу, так у следующего на очереди TX дескриптора бит EN выставлен в 0	RC	0
23	RXDMA_NO_D ESCR	Выставляется в 1, когда RX канал DMA останавливает работу, так у следующего на очереди RX дескриптора бит EN выставлен в 0	RC	0
22	TXDMA_BUS_E RROR	Выставляется в 1, когда происходит ошибка доступа к основной памяти каналом TX DMA (когда адрес, куда пытается обратиться DMA не существует в системе или запрещен для чтения/записи)	RC	0
21	RXDMA_BUS_E RROR	Выставляется в 1, когда происходит ошибка доступа к основной памяти каналом RX DMA (когда адрес, куда пытается обратиться DMA не существует в системе или запрещен для чтения/записи)	RC	0
20	TXDMA_DESC R_DONE	Выставляется в 1, когда TX канал DMA заканчивает обработку дескриптора. При этом бит IE в этом дескрипторе должен быть выставлен в 1. В момент выставления этого бита в 1 все данные из буфера в основной памяти переданы в TXFIFO, но еще не все данные переданы в канал SpaceWire.	RC	0
19	RXDMA_DESC R_DONE	Выставляется в 1, когда RX канал DMA заканчивает обработку дескриптора. При этом бит IE в этом дескрипторе должен быть выставлен в 1. В ходе приема пакета может быть обработано несколько дескрипторов.	RC	0
18	TXDMA_PACKE T_DONE	Выставляется в 1, когда заканчивается передача пакета. В этот момент все данные из буфера в основной памяти переданы в канал SpaceWire.	RC	0

Биты	Название	Описание	Доступ	Сброс
17	RXDMA_PACKET_DONE	Выставляется в 1, когда RX канал DMA заканчивает прием пакета. В ходе приема пакета может быть обработано несколько дескрипторов.	RC	0
16	NOTWRACK	Символ подтверждения прерывания не был отправлен на передачу: • 1 - символ подтверждения прерывания не отправлен на передачу • 0 - символ подтверждения прерывания корректно отправлен на передачу	RC	0
15	NOTWRINT	Символ прерывания не был отправлен на передачу: • 1 - символ прерывания не отправлен на передачу • 0 - символ прерывания корректно отправлен на передачу	RC	0
14	NOTACK	В заданное время не пришел пакет ACK на отправленное прерывание: • 1 - нет принятого символа подтверждения на отправленное прерывание • 0 - символы подтверждения прерывания приходят корректно	RC	0
13	GOTACK	Признак приема символа подтверждения прерывания из канала связи SpaceWire: • 1 - принят символ подтверждения прерывания • 0 - нет принятого символа подтверждения прерывания	RC	0
12	GOTINT	Признак приема символа прерывания из канала связи SpaceWire: • 1 - принят символ прерывания • 0 - нет принятого символа прерывания	RC	0
11	ACK_SENT	Признак передачи символа подтверждения прерывания по каналу связи SpaceWire: • 1 - символ подтверждения прерывания передан корректно • 0 - нет переданного символа подтверждения прерывания	RC	0
10	INT_SENT	Признак передачи символа прерывания по каналу связи SpaceWire: • 1 - символ прерывания передан корректно • 0 - нет переданного символа прерывания	RC	0
9	GOTTIME_BAD	Признак приема неверного маркера времени из канала связи SpaceWire: • 1 - маркер времени принят с ошибкой • 0 - нет принятого маркера времени	RC	0
8	GOTTIME_GOOD	Признак приема корректного маркера времени из канала связи SpaceWire: • 1 - маркер времени принят корректно • 0 - нет принятого маркера времени	RC	0
7	TIME_SENT	Признак отправки маркера времени по каналу связи SpaceWire: • 1 - маркер времени отправлен корректно • 0 - нет отправленного маркера времени	RC	0
6	CREDERR	Признак ошибки кредитования: • 1 - зафиксирована ошибка • 0 - нет ошибки	RC	0
5	ESCERR	Признак ошибки в ESC последовательности:	RC	0

Биты	Название	Описание	Доступ	Сброс
		<ul style="list-style-type: none"> <li>1 - зафиксирована ошибка</li> <li>0 - нет ошибки</li> </ul>		
4	DSCERR	Признак рассоединения с каналом связи SpaceWire: <ul style="list-style-type: none"> <li>1 - произошло рассоединение</li> <li>0 - нет рассоединения</li> </ul>	RC	0
3	PERR	Признак ошибки четности: <ul style="list-style-type: none"> <li>1 - зафиксирована ошибка</li> <li>0 - нет ошибки</li> </ul>	RC	0
2:0	STATE	Текущее состояние блока SpaceWire: <ul style="list-style-type: none"> <li>000 - Error Reset</li> <li>001 - Error Wait;</li> <li>010 - Ready</li> <li>011 - Started</li> <li>100 - Connecting</li> <li>101 - Run</li> </ul>	R	0

**SPW\_DIV [0x10].**

Биты	Название	Описание	Доступ	Сброс
31:14	-	Резерв	R	0
13:8	DIV_TX	Коэффициент деления $F_{spw}$ , для получения требуемой скорости передачи	R/W	0
7:6	-	Резерв	R	0
5:0	DIV_10MGz	Коэффициент деления $F_{spw}$ , для получения 10 МГц	R/W	0

Поле DIV\_TX определяет скорость работы передающего тракта SPW.

Расчет скорости передачи происходит по формуле:  $F_{tx} = F_{spw}/(DIV\_TX+1)$ ,

где  $F_{tx}$  - конечная частота, с которой должны быть переданы данные,  $F_{spw}$  - частота работы SPW, DIV\_TX - коэффициент поля DIV\_TX регистра SPW\_DIV

В поле DIV\_10MGz необходимо указать коэффициент, на который будет поделена частота работы SPW. В результате деления необходимо получить частоту 10 МГц.

Расчет происходит по формуле:  $F_{10MGz} = F_{spw}/(DIV\_10MGz+1)$ ,

где  $F_{10MGz}$  - конечная частота, которая должна быть 10 МГц,  $F_{spw}$  - частота работы SPW, DIV\_10MGz - коэффициент поля DIV\_10MGz регистра SPW\_DIV

**SPW\_PAUSE [0x14].**

Биты	Название	Описание	Доступ	Сброс
31:9	-	Резерв	R	0
8:0	DSCTIME	Количество тактов $F_{spw}$ необходимое на интервал работы счетчика 850 нс	R/W	0

Время, через которое SPW зафиксирует ошибку «disconnect» рассчитывается следующим образом:

$$T_{disc} = (DSCTIME+1)/F_{spw}$$

где  $F_{spw}$  - частота работы SPW, DSCTIME - значение поля DSCTIME регистра SPW\_PAUSE

Для соответствия стандарту SPW  $T_{disc}$  должно быть максимально близко к 850 нс

## SPW\_MSK\_INT [0x18].

Биты	Название	Описание	Доступ	Сброс
31:0	MSK_INT	Регистр определяет номера пакетов прерываний, которые будут обработаны системой. Порядковый номер бита соответствует номеру пакета	R/W	0

## SPW\_MSK\_ACK [0x1C].

Биты	Название	Описание	Доступ	Сброс
31:0	MSK_ACK	регистр определяет номера пакетов прерываний, на который будут автоматический отсылаться пакет подтверждения, и по отправке которых будет ожидать пакет подтверждения прерывания. Порядковый номер бита соответствует номеру пакета	RC	0

## SPW\_WAITACK\_CNT [0x20].

Биты	Название	Описание	Доступ	Сброс
31:24	LOC_CNT2	Время до отправки кода подтверждения на код прерывания, принятого из сети, в тактах Fapb	R/W	0
23:16	LOC_CNT1	Значение периода ожидания кода подтверждения на код прерывания, отправленного из SPW, в тактах Fapb	R/W	0
15:0	GLOB_CNT	Значение периода глобального счетчика в тактах Fapb	R/W	0

В этот регистр записываются значение периода для глобального счетчика (в количестве тактов Fapb) и максимальные значения локальных счетчиков ожидания кодов подтверждения распределенных прерываний. Отдельный локальный счетчик таймаутов соответствует каждому разряду IRQ. Если в SPW поступает код распределенного прерывания, то запускается соответствующий ему счетчик локальных таймаутов. Он декрементируется каждый раз при завершении очередного периода счета глобального счетчика таймаутов

## SPW\_INT\_ACK [0x24].

Биты	Название	Описание	Доступ	Сброс
31:24	-	Резерв	R	0
23:19	ACK_RX	Регистр последнего принятого подтверждения прерывания	R	0
18	ACK_BLOCK	Отправка пакета ACK заблокирована	R	0
17	TX_ACK	Отправить подтверждение прерывания, записанное в ACK	W	0
16:12	ACK	Номер подтверждения для отправки	R/W	0
11:7	INT_RX	Регистр последнего принятого прерывания	R	0
6	INT_BLOCK	Отправка прерывания заблокирована	R	0
5	TX_INT	Отправить прерывание, записанное в INT	W	0
4:0	INT	Номер прерывания для отправки	R/W	0

Биты ACK\_BLOCK и INT\_BLOCK встают в «1», когда пользователь в ручном режиме отправляет соответствующий пакет. Данные биты оповещают пользователя, что пакет не отправлен, поэтому не стоит пытаться отправлять новый пакет пока данный бит в «1». Когда нужный бит перейдет в «0» пользователь может отправить новый пакет.

Пользователю запрещено отправлять в ручном режиме коды подтверждения прерывания, совпадающие с автоматический отправляемыми кодами.

**SPW\_INTPACK [0x28].**

Биты	Название	Описание	Доступ	Сброс
31:17	INTPACK	Регистр фиксирует номера пакетов принятых и отправленных прерываний. Порядковый номер бита соответствует номеру пакета	RW1C	0

**SPW\_ACK\_NOT [0x2C].**

Биты	Название	Описание	Доступ	Сброс
31:0	SPW_ACK_NOT	Регистр фиксирует номера пакетов прерываний, на которые не был получен пакет подтверждения. Порядковый номер бита соответствует номеру пакета	RW1C	0

**SPW\_TIME\_CNT [0x30].**

Биты	Название	Описание	Доступ	Сброс
31:0	TIME_CNT	регистр определяет, через какое количество тактов Fapb будет отправлена инкрементированная временная метка	R/W	0

**SPW\_TIME [0x34]**

Биты	Название	Описание	Доступ	Сброс
31:19	-	Резерв	R	0
18:13	TIME_RX	Значение маркера времени, принятого из сети последним	R	0
12	TIME_TX_EN	При установке этого бита инициируется процесс передачи маркера времени. После этого TIME_TX_EN автоматически сбрасывается в ноль	W	0
11:6	TIME_TX	Значение маркера времени для передачи при установке бита TIME_TX_EN	R/W	0
5:0	TIME_NOW	Текущее время сети	R	0

**SPW\_TX\_DESCR\_ADDR [0x38].**

Биты	Название	Описание	Доступ	Сброс
31:2	TX_DESCR_ADDR	Базовый адрес таблицы TX дескрипторов (старшие 30 разрядов адреса). Младшие 2 разряда адреса зарезервированы и всегда равны 0.	R	0
1:0	-	Резерв	R	0

**SPW\_RX\_DESCR\_ADDR [0x3C].**

Биты	Название	Описание	Доступ	Сброс
31:2	RX_DESCR_ADDR	Базовый адрес таблицы RX дескрипторов (старшие 30 разрядов адреса). Младшие 2 разряда адреса зарезервированы и всегда равны 0.	RC	0
1:0	-	Резерв	R	0

**SPW\_LADDR [0x40].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0

Биты	Название	Описание	Доступ	Сброс
7:0	LADDR	Логический адрес устройства	R/W	0

**SPW\_KEY [0x44].**

Биты	Название	Описание	Доступ	Сброс
31:8	-	Резерв	R	0
7:0	KEY	Ключ авторизации устройства	R/W	0

**SPW\_ST\_RMAP [0x48].**

Биты	Название	Описание	Доступ	Сброс
31:18	-	Резерв	R	0
17	ERR_NSEN D_HR	Произошла ошибка при работе с заголовком пакета, ответное слово не отправлено • 1 - Зафиксирована ошибка • 0 - Ошибки отсутствуют	RC	0
16	ERR_SEND _HR	Произошла ошибка при работе с заголовком пакета, ответное слово отправлено • 1 - Зафиксирована ошибка • 0 - Ошибки отсутствуют	RC	0
15	REPLY	Пакет является ответом, на запрошенную операцию • 1 - Пакет является ответом • 0 - Ответных пакетов не зафиксировано	RC	0
14	RDMW_OP	Происходило взаимодействие в рамках операции «Чтение- Модификация-Запись» • 1 - Зафиксировано взаимодействие • 0 - Взаимодействие не обнаружено	RC	0
13	RD_OP	Происходило взаимодействие в рамках операции «Чтение» • 1 - Зафиксировано взаимодействие • 0 - Взаимодействие не обнаружено	RC	0
12	WR_OP	Происходило взаимодействие в рамках операции «Запись» • 1 - Зафиксировано взаимодействие • 0 - Взаимодействие не обнаружено	RC	0
11	EEP_END	При формировании ответного слова произошла ошибка, пакет закончен символом EEP • 1 - Зафиксирована ошибка • 0 - Ошибки отсутствуют	RC	0
10	ERR_DATA	Произошла ошибка при работе с данными пакета • 1 - Зафиксирована ошибка • 0 - Ошибки отсутствуют	RC	0
9	ERR_HR	Произошла ошибка при работе с заголовком пакета • 1 - Зафиксирована ошибка • 0 - Ошибки отсутствуют	RC	0
8	GOOD	Успешное окончание операции • 1 - Успешное окончание • 0 - Операции отсутствовали или возникли какие- либо проблемы При возникновении проблемы один из битов EEP_END, ERR_DATA, ERR_HR будет выставлен в «1»	RC	0



Биты	Название	Описание	Доступ	Сброс
7:0	ST_CODES	Коды статусов в соответствии со стандартом RMAP. Отображается код последней операции по протоколу RMAP. К обрабатываемым операциям относятся: запрос на операцию, ответ на запрос на операцию, результат операции.	RC	0

При возникновении ошибки с заголовком пакета(ERR\_HR) и выставлении бита ERR\_NSEND\_HR значение остальных полей, кроме CODES значения не имеют

#### SPW\_MSK\_INT\_RMAP [0x4C].

Биты	Название	Описание	Доступ	Сброс
31:18	-	Резерв	R	0
17	ERR_NSEND_HR_MSK	Разрешение прерывания по биту ERR_NSEND_HR регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
16	ERR_SEND_HR_MSK	Разрешение прерывания по биту ERR_SEND_HR регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
15	REPLY_MSK	Разрешение прерывания по биту REPLY регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
14	RDMW_OP_MSK	Разрешение прерывания по биту RDMW_OP регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
13	RD_OP_MSK	Разрешение прерывания по биту RD_OP регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
12	WR_OP_MSK	Разрешение прерывания по биту WR_OP регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
11	EEP_END_MSK	Разрешение прерывания по биту EEP_END регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
10	ERR_DATA_MSK	Разрешение прерывания по биту ERR_DATA регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
9	ERR_HR_MSK	Разрешение прерывания по биту ERR_HR регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0
8	GOOD_MSK	Разрешение прерывания по биту GOOD регистра SPW_ST_RMAP • 1 - Прерывание разрешено; • 0 - Прерывание запрещено.	R/W	0



Биты	Название	Описание	Доступ	Сброс
7:0	-	Резерв	R	0

**SPW\_RMAP\_REGION\_BASE\_ADDR\_x [0x50 / 0x54 / 0x58 / 0x5C].**

Биты	Название	Описание	Доступ	Сброс
31:2	REGION_BASE_ADDR	Базовый адрес региона памяти, разрешенного для доступа RMAP. Младшие 2 бита адреса зарезервированы и всегда равны 0.	R/W	0
1	WR_EN	Запись в этот регион памяти разрешена для RMAP	R/W	0
0	RD_EN	Чтение этого региона памяти разрешено для RMAP	R/W	0

**SPW\_RMAP\_REGION\_EXTENDED\_ADDR [0x60].**

Биты	Название	Описание	Доступ	Сброс
31:24	REGION_ADDR_EXTENDED_3	Расширенный адрес региона памяти 3, разрешенного для доступа RMAP	R/W	0
23:16	REGION_ADDR_EXTENDED_2	Расширенный адрес региона памяти 2, разрешенного для доступа RMAP	R/W	0
15:8	REGION_ADDR_EXTENDED_1	Расширенный адрес региона памяти 1, разрешенного для доступа RMAP	R/W	0
7:0	REGION_ADDR_EXTENDED_0	Расширенный адрес региона памяти 0, разрешенного для доступа RMAP	R/W	0

**SPW\_RMAP\_REGION\_SIZE [0x64 / 0x68 / 0x6C / 0x70].**

Биты	Название	Описание	Доступ	Сброс
31:2	REGION_SIZE	Размер (в байтах) региона памяти, разрешенного для доступа RMAP. Младшие 2 бита адреса зарезервированы и всегда равны 0.	R/W	0
1:0	-	Резерв	R	0

## Таймер сна (SLEEP TIMER)

Таймер сна предназначен для вывода системы из режима глубокого сна через заданное пользователем время. STIMER нельзя использовать для вывода системы из обычного режима сна - для этой цели можно воспользоваться одним из обычных таймеров в системе.

Таймер сна представляет собой 24-разрядный таймер, считающий вверх (от нуля к заданному периоду) на частоте  $Ip\_clk$ . Эта частота отличается от остальных частот в системе тем, что не отключается в режиме глубокого сна. После выхода из глубокого сна, таймер сна сформирует прерывание.

### Общее описание

Таймер сна предназначен для вывода системы из режима глубокого сна через заданное пользователем время. STIMER нельзя использовать для вывода системы из обычного режима сна - для этой цели можно воспользоваться одним из обычных таймеров в системе.

Таймер сна представляет собой 24-разрядный таймер, считающий вверх (от нуля к заданному периоду) на частоте  $Ip\_clk$ . Эта частота отличается от остальных частот в системе тем, что не отключается в режиме глубокого сна. Пользователь должен включить таймер записью в регистр STIMER\_CTRL перед переходом в режим глубокого сна. В момент перехода в режим глубокого сна (когда процессор выполнит инструкцию WFI) таймер начнет счет. По окончании счета будет сформирован сигнал WAKEUP для PMM, который выведет систему из режима глубокого сна. После выхода из глубокого сна, таймер сна сформирует прерывание. Если система выйдет из режима глубокого сна по другой причине (сигнал WAKEUP от GPIO), то таймер будет сброшен в 0, но прерывание и сигнал WAKEUP сформированы не будут.

Значение, до которого будет считать таймер определяется как:

$$N_t = \text{Период счета таймера} + 1$$

где  $N_t$  - количество тактов частоты  $Ip\_clk$ .

Актуальное значение периода счета обязательно должно быть записано перед включением модуля «Таймер сна». Перед запуском данного модуля для снижения энергопотребления рекомендуется переключить систему на тактирование от RCL-генератора и выключить все остальные источники тактирования (см. документацию на CMM). Пользователю необходимо удостовериться, что процесс переключения источника частоты закончен, и только после этого включать таймер сна и уходить в режим глубокого сна.

### Регистры модуля STIMER

Смещение от базового адреса блока	Аббревиатура	Описание
0x0	STIMER_CTRL	Регистр управления
0x4	STIMER_PERIOD	Регистр периода счета

### Sleep Timer Control (STIMER\_CTRL) [0x00].

Биты	Название	Описание	Доступ	Сброс
31:1	-	Резерв	R	0
0	TIMER_EN	Разрешение работы таймера сна: <ul style="list-style-type: none"> <li>0 - запрещена работа таймера сна</li> <li>1 - разрешена работа таймера сна. Таймер начнет счет в момент перехода системы в режим глубокого сна.</li> </ul> Бит не сбрасывается по окончании счета, может быть только очищен пользователем	R/W	0

**Sleep Timer Period (STIMER\_PERIOD) [0x04].**

Биты	Название	Описание	Доступ	Сброс
31:24	-	Резерв	R	0
23:0	PERIOD	Период счета таймера сна	R/W	0

## Таймеры

Каждый таймер в системе поддерживает 3 различных независимых режима работы: «Простой таймер» - программный старт счета, окончание счета по достижению заданного периода; «Таймер с внешней остановкой» - программный старт счета, окончание счета по внешнему событию; «Межсобытийный таймер» - старт и окончание счета по внешнему событию.

Кроме того, таймеры 0 и 1 (но не таймер 2) дополнительно поддерживают 4-ый режим работы - «Таймер-счетчик». В этом режиме Таймеры 0 и 1 работают вместе, один в качестве таймера, второй в качестве счетчика внешних событий. Этот режим можно использовать, чтобы определить, за какое время произошло заданное количество внешних событий.

В режиме «Простой таймер» модуль представляет собой 24-х разрядный таймер с инкрементацией каждый такт рабочей частоты. Таймер считает до значения, записанного в регистр. По достижению заданного значения таймер либо останавливается, либо начинает счет с нуля. Также во время работы допускается перезапись текущего периода счета таймера. В результате перезаписи текущее значение таймера будет сброшено в 0 и счет начнется заново.

В режиме «Таймер с внешней остановкой» модуль представляет собой 24-разрядный таймер с инкрементацией каждый такт рабочей частоты. При запуске таймер считает с нуля до момента возникновения заданного события на внешнем выводе (GPIOA [24:26] альтернативная функция). Сторожевой таймер (WatchDog Timer)

Сторожевой таймер предназначен для принудительной перезагрузки системы в случае её зависания. В основе WDT лежит счетчик разрядностью 32 бита. Модуль формирует регулярное прерывание в зависимости от запрограммированного значения. Каждый такт синхросигнала значение счетчика уменьшается на единицу. Когда значение счетчика достигает 0, формируется сигнал прерывания. Затем счетчик перезагружается и заново начинает отсчет к нулю. Если к моменту, когда счетчик достиг заново значения 0, прерывание не очищено, то в систему формируется сигнал сброса. Таким образом, сторожевой таймер предоставляет возможность восстановления системы после сбоя программного обеспечения. При необходимости модуль может быть выключен.

Каждый таймер в системе поддерживает 3 различных независимых режима работы:

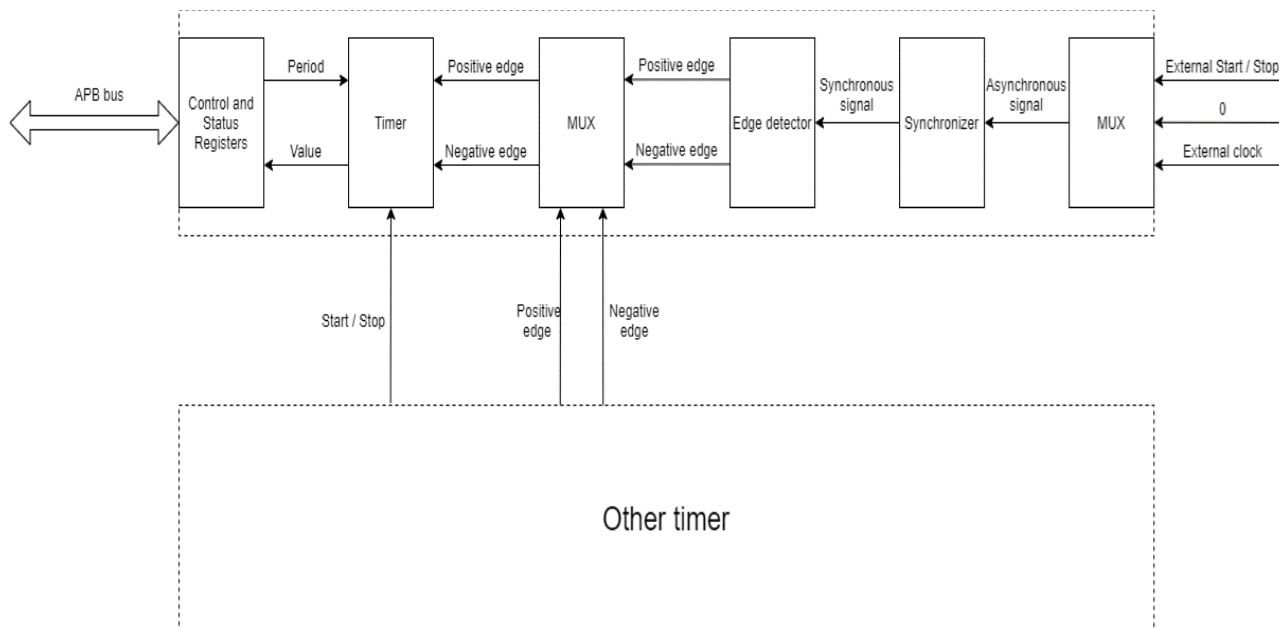
«Простой таймер» - программный старт счета, окончание счета по достижению заданного периода

«Таймер с внешней остановкой» - программный старт счета, окончание счета по внешнему событию

«Межсобытийный таймер» - старт и окончание счета по внешнему событию

Кроме того, Таймеры 0 и 1 (но не Таймер 2) дополнительно поддерживают 4-ый режим работы - «Таймер-счетчик». В этом режиме Таймеры 0 и 1 работают вместе, один в качестве таймера, второй в качестве счетчика внешних событий. Этот режим можно использовать, чтобы определить, за какое время произошло заданное количество внешних событий.

## Структурная схема



- Control and status Registers (Управляющие и статусные регистры) - хранят управляющие данные и выдают на шину статусные данные.
- Timer (Таймер).
- MUX (Переключатель) - передает на выход выбранный входной сигнал.
- Edge Detector (Выделитель) - выделяет фронт и спад внешнего сигнала.
- Synchronizer (Синхронизатор) - синхронизирует внешний сигнал к системной частоте.

## Простой таймер

## Принцип работы

В режиме «Простой таймер» модуль представляет собой 24-разрядный таймер с инкрементацией каждый такт рабочей частоты. Таймер считает до значения, записанного в регистр **TMR\_PERIOD**. По достижению заданного значения таймер либо останавливается, либо начинает счет с нуля. Данная функция определяется битом **TMR\_CTRL.CYCLES**. Также во время работы допускается перезапись текущего периода счета таймера. В результате перезаписи текущее значение таймера будет сброшено в 0 и счет начнется заново. Текущее значение таймера находится в регистре **TMR\_VALUE**.

Значение, до которого будет считать модуль определяется как:

$$N_t = \text{Период счета таймера} + 1$$

где  $N_t$  - количество тактов системной частоты.

## Статусы и прерывания

Статусы содержатся в регистре **TMR\_ST**. В данном режиме вырабатывается статус окончания счёта периода — бит **END\_PRD**. На основании данного статуса возможно возникновение прерывания. Прерывание разрешается путем записи 1 в соответствующий бит регистра маски прерываний

**TMR\_MSK.****Алгоритм работы**

- 1) Пользователь записывает требуемое значение периода счета таймера в регистр **TMR\_PERIOD**.
- 2) Пользователь разрешает нужные прерывания путем записи единицы в соответствующие биты регистра **TMR\_MSK**.
- 3) Пользователь настраивает **TMR\_CTRL**:

Выбирает требуемое значение бита **CYCLES**.

В бит **T/C** необходимо записать значение 0.

В биты **MODE** необходимо записать значение 00.

В бит **EXT\_EN** необходимо записать значение 0.

В бит **EN** записывается значение 1.

Текущее значения таймера может быть считано из регистра **TMR\_VALUE**.

**Таймер с внешней остановкой****Принцип работы**

В режиме «Таймер с внешней остановкой» модуль представляет собой 24-разрядный таймер с инкрементацией каждый такт рабочей частоты. При запуске таймер считает с нуля до момента возникновения заданного события на внешнем пине (**GPIOA[24:26]** альтернативная функция). Тип события остановки определяется битом **TMR\_CFG.STOP\_TYPE**. Текущее значение таймера находится в регистре **TMR\_VALUE**.

**Статусы и прерывания**

Статусы содержатся в регистре **TMR\_ST**. В данном режиме вырабатывается статус переполнения таймера — бит **OVW** и статус остановки таймера по внешнему событию — бит **STOP\_EVENT**. На основании данных статусов возможно возникновение прерываний. Прерывание разрешается путем записи 1 в соответствующий бит регистра маски прерываний **TMR\_MSK**.

**Алгоритм работы**

- 1) Пользователь задает тип события остановки счета таймера, путем записи требуемого значения в бит **TMR\_CFG.STOP\_TYPE**.

- 2) Пользователь разрешает нужные прерывания путем записи единицы в соответствующие биты регистра **TMR\_MSK**.

- 3) Пользователь настраивает **TMRx\_CTRL**:

В бит **CYCLES** необходимо записать значение 0.

В бит **T/C** необходимо записать значение 0.

В биты **MODE** необходимо записать значение 01.

В бит **EXT\_EN** необходимо записать значение 0.

В бит **EN** записывается значение 1.

Текущее значения таймера может быть считано из регистра **TMR\_VALUE**.

**Межсобытийный таймер****Принцип работы**

В режиме «Межсобытийный таймер» модуль представляет собой 24-разрядный таймер с

инкрементацией каждый такт рабочей частоты. После разрешения работы таймер ожидает событие старта на внешнем пине. Тип события старта определяется битом **TMR\_CFG.START\_TYPE**. Окончанием счета является событие остановки счета таймера на внешнем пине. Тип события остановки определяется битом **TMR\_CFG.STOP\_TYPE**. Текущее значение таймера находится в регистре **TMR\_VALUE**.

### Статусы и прерывания

Статусы содержатся в регистре **TMR\_ST**. В данном режиме вырабатывается статус переполнения таймера — бит **OVW**, статус запуска таймера по внешнему событию — бит **START\_EVENT** и статус остановки таймера по внешнему событию — бит **STOP\_EVENT**. На основании данных статусов возможно возникновение прерываний. Прерывание разрешается путем записи 1 в соответствующий бит регистра маски прерываний **TMR\_MSK**.

### Алгоритм работы

1) Пользователь задает тип события запуска счета и остановки счета таймера, путем записи требуемого значения в биты **START\_TYPE** и **STOP\_TYPE** регистра **TMR\_CFG**.

2) Пользователь разрешает нужные прерывания путем записи единицы в соответствующие биты регистра **TMR\_MSK**.

3) Пользователь настраивает **TMR\_CTRL**:

В бит **CYCLES** необходимо записать значение 0.

В бит **T/C** необходимо записать значение 0.

В биты **MODE** необходимо записать значение 10.

В бит **EXT\_EN** необходимо записать значение 0.

В бит **EN** записывается значение 1.

Текущее значения таймера может быть вычитано из регистра **TMR\_VALUE**.

### Таймер-счётчик

Примечание: данный режим работы поддерживают только Таймер 0 и 1. Таймер 2 данный режим работы не поддерживает. При запуске Таймера 2 в режиме «таймер-счетчик» модуль уйдёт в состояние ожидания, выход из которого возможен только по отключению модуля.

### Принцип работы

В режиме «Таймер-счетчик» модули взаимодействуют между собой. Один из модулей необходимо настроить в режим таймера, а второй в режим счетчика. На внешний вход модуля, который работает в режиме счетчика могут быть поданы тактовая частота (Для таймера 0 - RCL, для таймера 1 - RCH), либо сигнал с внешнего пина. Вид события определяется битом **TMR\_CFG.EVENT\_TYPE**. После настройки модуль в режиме таймера и модуль в режиме счетчика ожидают события старта, тип которого определяется битом **TMR\_CFG.START\_TYPE**. Затем модуль, работающий в режиме таймера, считает до момента остановки его модулем в режиме счетчика. Модуль в режиме счетчика считает до значения, записанного в регистр **TMR\_PERIOD**. Счетчик увеличивается на 1 каждый раз, когда фиксирует заданное событие. Тип события определяется битом **TMR\_CFG.FIX\_TYPE**. По завершению счета модуль в режиме счетчика формирует событие окончания счета для модуля, работающего в режиме таймера. В результате оба модуля прекращают счет. Перезаписывать значение регистра **TMR\_PERIOD** для модуля в режиме счетчика в ходе работы запрещено.

Значение, до которого будет считать счетчик определяется как:

$N_e = \text{Период счета таймера} + 1$

где  $N_e$  - количество событий.

## Статусы и прерывания

Статусы содержатся в регистре **TMR\_ST**.

В данном режиме, для модуля, работающего в режиме таймера, вырабатывается статус переполнения таймера — бит **OVW**, статус запуска таймера по внешнему событию — бит **START\_EVENT** и статус остановки таймера по внешнему событию — бит **STOP\_EVENT**.

Для модуля, работающего в режиме счетчика, вырабатывается только статус запуска таймера по внешнему событию — бит **START\_EVENT**.

На основании данных статусов возможно возникновение прерываний. Прерывание разрешается путем записи 1 в соответствующий бит регистра маски прерываний **TMR\_MSK**.

## Алгоритм работы

### Для модуля в режиме таймера:

1) Пользователь задает тип события запуска счета, путем записи требуемого значения в биты **TMR\_CFG.START\_TYPE**.

2) Пользователь разрешает нужные прерывания путем записи единицы в соответствующие биты регистра **TMR\_MSK**.

3) Пользователь настраивает **TMR\_CTRL**:

В бит **CYCLES** необходимо записать значение 0.

В бит **T/C** необходимо записать значение 0.

В биты **MODE** необходимо записать значение 11.

В бит **EXT\_EN** необходимо записать значение 1.

В бит **EN** записывается значение 1.

Текущее значения таймера может быть вычитано из регистра **TMR\_VALUE**.

Бит **EXT\_EN** разрешает соседнему модулю начать работу, таким образом синхронизируя запуск обоих модулей.

### Для модуля в режиме счетчика:

1) Пользователь задает тип события запуска счета, путем записи требуемого значения в биты **TMR\_CFG.START\_TYPE**.

2) Пользователь записывает требуемое значение периода счета в регистр **TMR\_PERIOD**.

3) Пользователь разрешает нужные прерывания путем записи единицы в соответствующие биты регистра **TMR\_MSK**.

4) Пользователь настраивает **TMR\_CTRL**:

В бит **CYCLES** необходимо записать значение 0.

В бит **T/C** необходимо записать значение 1.

В биты **MODE** необходимо записать значение 11.

В бит **EXT\_EN** необходимо записать значение 1.

В бит **EN** записывается значение 1.

Текущее значения счетчика может быть вычитано из регистра **TMR\_VALUE**.

Бит **EXT\_EN** разрешает соседнему модулю начать работу, таким образом синхронизируя запуск обоих модулей.



## Карта регистров

Смещение от базового адреса блока	Аббревиатура	Доступ	Описание
0x0	TMR_CTRL	RW	Регистр управления
0x4	TMR_CFG	RW	Регистр конфигурации
0x8	TMR_PERIOD	RW	Период счета таймера
0xC	TMR_VALUE	R	Текущее значение таймера
0x10	TMR_MSK	RW	Регистр маски прерываний
0x14	TMR_ST	RC	Регистр статуса

## TMR\_CTRL [0x0C].

Биты	Название	Описание	Доступ	Сброс
31:6	-	Резерв	R	0
5	CYCLES	тип работы таймера по достижению значения периода: • 1 - повторный счет с нуля; • 0 - остановка счета.	R/W	0
4	T/C	работа в режиме таймера или счетчика, имеет значение только при MODE «Таймер-счетчик»: • 1 - режим счетчика; • 0 - режим таймера.	R/W	0
3:2	MODE	режимы работы модуля: • 11 - «Таймер-счетчик»; • 10 - «Межсобытийный таймер»; • 01 - «Таймер с внешней остановкой»; • 00 - «Простой таймер».	R/W	0
1	EXT_EN	разрешение работы соседнего модуля «Таймер», имеет значение только при MODE «Таймер-счетчик»: • 1 - работа соседнего таймера разрешена; • 0 - работа соседнего таймера запрещена.	R/W	0
0	EN	разрешение работы таймера: • 1 - таймер включен; • 0 - таймер выключен.	R/W	0

При работе с регистрами модуля «Таймер 0» бит EXT\_EN управляет модулем «Таймер 1». При работе с регистрами модуля «Таймер 1» бит EXT\_EN управляет модулем «Таймер 0».

## TMR\_CFG [0x04].

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3	EVENT_TYPE	вид события, которое считается счетчиком в режиме «Таймер-счетчик»: • 1 - внешнее событие; • 0 - тактовый сигнал (Для таймера 0 - RCL, для таймера 1 - RCH, для таймера 2 - внешний CLK);	R/W	0
2	FIX_TYPE	тип события, которое считается счетчиком в режиме «Таймер-счетчик»: • 1 - событие заднего фронта; • 0 - событие переднего фронта.	R/W	0

Биты	Название	Описание	Доступ	Сброс
1	STOP_TYPE	тип события остановки счета таймера: • 1 - остановка по заднему фронту; • 0 - остановка по переднему фронту.	R/W	0
0	START_TYP E	тип события старта счета таймера: • 1 - старт по заднему фронту; • 0 - старт по переднему фронту.	R/W	0

**TMR\_PERIOD [0x08].**

Биты	Название	Описание	Доступ	Сброс
31:24	-	Резерв	R	0
23:0	PERIOD	Период счета таймера.	R/W	0

**TMR\_VALUE [0x0C].**

Биты	Название	Описание	Доступ	Сброс
31:24	-	Резерв	R	0
23:0	VALUE	текущее значение таймера.	R	0

**TMR\_MSK [0x10].**

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3	STOP_EVENT_ M	разрешение прерывания при остановке таймера по внешнему событию: • 1 - разрешено; • 0 - запрещено.	R/W	0
2	START_EVENT_ M	разрешение прерывания при запуске таймера по внешнему событию: • 1 - разрешено; • 0 - запрещено.	R/W	0
1	OVW_M	разрешение прерывания по переполнению таймера: • 1 - разрешено; • 0 - запрещено.	R/W	0
0	END_PRD_M	разрешение прерывания по окончанию счёта периода: • 1 - разрешено; • 0 - запрещено.	R/W	0

**TMR\_ST [0x14].**

Биты	Название	Описание	Доступ	Сброс
31:4	-	Резерв	R	0
3	STOP_EVENT	остановка таймера по внешнему событию: • 1 - зафиксировано событие остановки; • 0 - событие остановки не зафиксировано.	RC	0

Биты	Название	Описание	Доступ	Сброс
2	START_EVENT	запуск таймера по внешнему событию: <ul style="list-style-type: none"><li>1 - зафиксировано событие старта;</li><li>0 - событие старта не зафиксировано.</li></ul>	RC	0
1	OVW	переполнение таймера: <ul style="list-style-type: none"><li>1 - зафиксировано переполнение;</li><li>0 - переполнение не зафиксировано.</li></ul>	RC	0
0	END_PRD	конец счёта периода: <ul style="list-style-type: none"><li>1 - зафиксирован конец счёта периода;</li><li>0 - конец периода не зафиксирован.</li></ul>	RC	0

## Сторожевой таймер (WDT)

### Общее описание

Сторожевой таймер предназначен для принудительной перезагрузки системы в случае её зависания. Для большей защиты от случайного доступа к регистрам, операции записи могут быть программно заблокированы с помощью регистра WDT\_LOCK.

В основе WDT лежит счетчик разрядностью 32 бита. Модуль формирует регулярное прерывание INT\_WDT в зависимости от запрограммированного значения. Каждый такт синхросигнала значение счетчика уменьшается на единицу. Когда значение счетчика достигает 0, формируется сигнал прерывания. Затем счетчик перезагружается и заново начинает отсчет к нулю. Если к моменту, когда счетчик заново достиг значения 0, прерывание не очищено, то в систему формируется сигнал сброса. Таким образом, сторожевой таймер предоставляет возможность восстановления системы после сбоя программного обеспечения. При необходимости модуль может быть выключен.

Примечание: сброс WDT не сбрасывает CMM и DM.

### Регистры сторожевого таймера

Смещение от базового адреса блока	Аббревиатура	Доступ	Описание
0x0	WDT_LOAD	RW	Период счета сторожевого таймера
0x4	WDT_VAL	RO	Текущее значение сторожевого таймера
0x8	WDT_CTRL	RW	Регистр управления
0xC	WDT_CLR	WO	Регистр сброса прерывания
0x10	WDT_INTRAW	RO	Регистр исходного прерывания
0x14	WDT_INT	RO	Регистр маскируемого прерывания
0x18	WDT_LOCK	RW	Регистр блокировки доступа к WDT
0x1C	WDT_TCR	RW	Регистр перехода в тестовый режим
0x20	WDT_TOP	WO	Регистр управления в тестовом режиме

### WDT\_LOAD [0x00].

Биты	Название	Описание	Доступ	Сброс
31:0	VALUE	Период счета сторожевого таймера. Регистр содержит значение, с которого счетчик начнет уменьшаться. При перезаписи данного регистра таймер немедленно стартует с записанного значения. Минимальное допустимое значение для WDT_LOAD равно 1.	R/W	0

**WDT\_VAL [0x04]**

Биты	Название	Описание	Доступ	Сброс
31:0	CURRENT_VALUE	Текущее значение таймера.	RO	0

**WDT\_CTRL [0x08]**

Биты	Название	Описание	Доступ	Сброс
31:2	-	Резерв	R	0
1	RST_EN	Reset Enable. Разрешение на формирование сигнала сброса по завершению счета таймера: • 1 – сброс разрешен; • 0 – сброс запрещен.	R/W	0
0	INT_EN	Interrupt Enable. Разрешение на формирование прерывания по окончании счета таймера: • 1 – прерывание разрешено; • 0 – прерывание запрещено.	R/W	0

Сторожевой таймер начинает счет, если установлен бит **WDT\_CTRL.INT\_EN**. Сторожевой таймер перестает считать, если бит **WDT\_CTRL.INT\_EN** сброшен в ноль.

После разрешения прерывания, если пользователь до этого его запрещал, счетчик автоматически перезагрузится со значения из регистра **WDT\_LOAD**.

**WDT\_CLR [0x0C]**

Биты	Название	Описание	Доступ	Сброс
31:0	CLEAR	Запись любого значения в данный регистр очищает прерывание сторожевого таймера и перезагружает счетчик значением из регистра <b>WDT_LOAD</b> .	WO	0

**WDT\_INTRAW [0x10]**

Биты	Название	Описание	Доступ	Сброс
31:1	-	Резерв	R	0
0	RAW_WDT	Исходное прерывание таймера: • 1 – прерывание произошло; • 0 – прерывание отсутствует.	RO	0

Данный регистр указывает на необработанное прерывание от счетчика. На основании данного сигнала формируется маскируемое прерывание.

Тестовое прерывание, сформированное записью 1 в **WDT\_TOP.TST\_INT**, не отображается в этом регистре.

**WDT\_INT [0x14]**

Биты	Название	Описание	Доступ	Сброс
31:1	-	Резерв	R	0
0	INT_WDT	Маскируемое прерывание таймера: • 1 – прерывание произошло; • 0 – прерывание отсутствует.	RO	0

Данное прерывание формируется на основании битов **WDT\_INTRAW.RAW\_WDT** и **WDT\_CTRL.INT\_EN** и передаётся в систему.

**WDT\_LOCK [0x18]**

Биты	Название	Описание	Доступ	Сброс
31:1	-	Резерв	R	0
0	LOCK_WDT	Блокировка записи в регистры: <ul style="list-style-type: none"> <li>1 – запись во все регистры сторожевого таймера заблокирована;</li> <li>0 – запись во все регистры сторожевого таймера разрешена.</li> </ul>	R/W	0

Данный регистр блокирует доступ на запись во все остальные регистры сторожевого таймера. Запись значения 0x55 обеспечивает доступ на запись ко всем регистрам. Запись любого другого значения блокирует доступ.

По умолчанию запись во все регистры сторожевого таймера заблокирована.

**WDT\_TCR [0x1C]**

Биты	Название	Описание	Доступ	Сброс
31:1	-	Резерв	R	0
0	TEST_EN	Тестовый режим: <ul style="list-style-type: none"> <li>1 – сторожевой таймер в тестовом режиме;</li> <li>0 – сторожевой таймер в рабочем режиме.</li> </ul>	R/W	0

В тестовом режиме пользователь непосредственно управляет маскируемым прерыванием и сбросом от сторожевого таймера через регистр **WDT\_TOP**.

**WDT\_TOP [0x20]**

Биты	Название	Описание	Доступ	Сброс
31:2	-	Резерв	R	0
1	TST_INT	Формирование сигнала прерывания от таймера в тестовом режиме: <ul style="list-style-type: none"> <li>1 – прерывание активно;</li> <li>0 – прерывание неактивно.</li> </ul>	RO	0
0	TST_RES	Формирование сигнала сброса от таймера в тестовом режиме: <ul style="list-style-type: none"> <li>1 – сброс активен;</li> <li>0 – сброс неактивен.</li> </ul>	RO	0

## Система управления сбросом и питанием (PMM)

Блок PMM выполняет две основные функции: управляет режимами сна и формирует сигналы сброса для всех остальных компонентов системы.

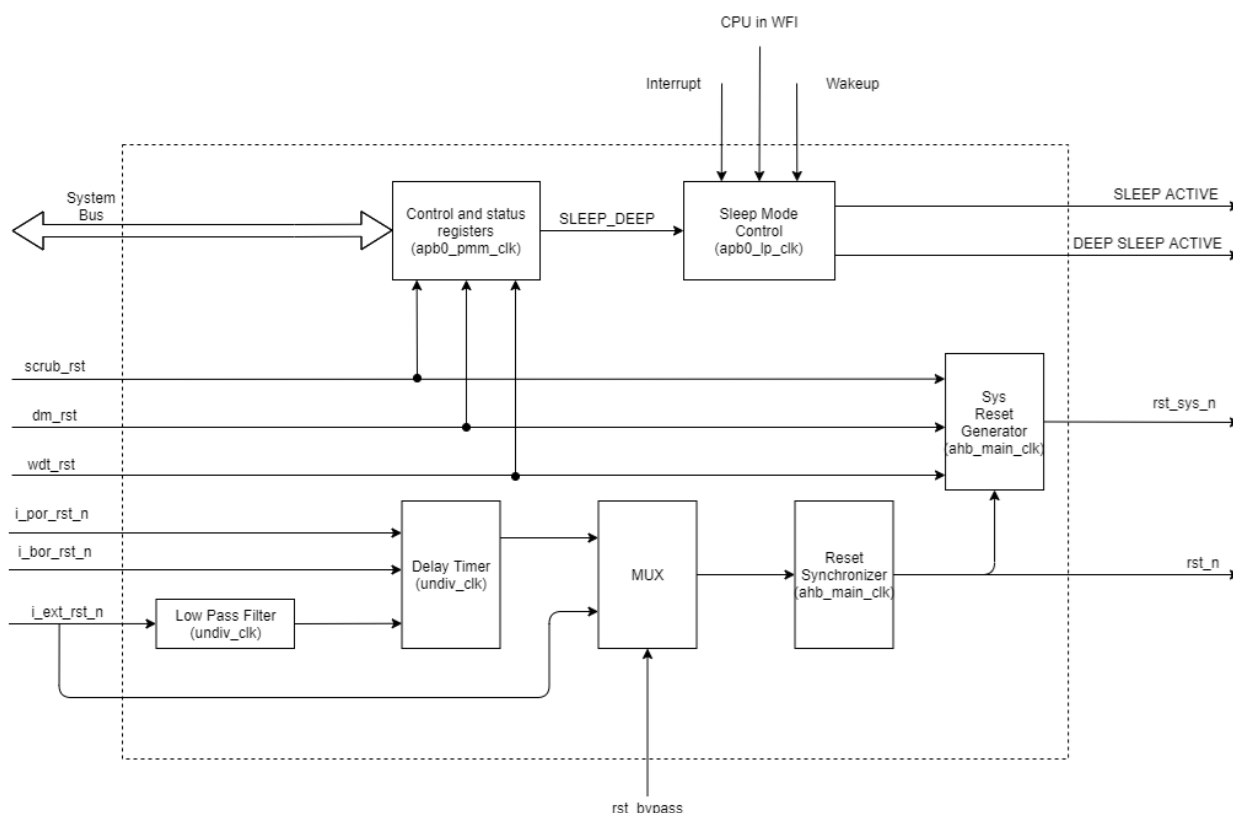
### Основные функции

Блок PMM выполняет две основные функции:

Управляет режимами сна;

Формирует сигналы сброса для всех остальных компонентов системы.

### Структурная схема



### Управление режимами сна

Управляющие и статусные регистры доступны для чтения и записи по системной шине (System Bus). Бит PMM\_CTRL.SLEEP\_DEEP определяет, в какой именно режим сна перейдет система, когда процессор выполнит инструкцию WFI. Если бит выставлен в 0, то система перейдет в режим «Сон процессора», а если бит выставлен в 1, то в режим «Глубокий сон».

После появления сигнала CPU in WFI система переходит в режим сна - PMM выдает в CMM сигнал SLEEP\_ACTIVE или DEEP\_SLEEP\_ACTIVE и CMM отключает тактовые сигналы, соответствующие данному режиму сна. Блок управления режимами сна выходит из режима «Сон процессора» при регистрации любого прерывания в системе (Interrupt), и выходит из режима «Глубокий сон» по сигналу пробуждения (Wakeup).

### Формирование сигналов сброса

Первоначальный сброс системы после подачи питания проводят блоки BOR и POR — сигналы i\_por\_rst\_n и i\_bor\_rst\_n. Когда любой из этих сбросов активен (активный уровень у этих сигналов - 0),

то таймер задержки (Delay Timer) сброшен в 0. Когда таймер сброшен в 0, то в систему подается 0 (активный уровень сброса). Значение внешнего сигнала сброса (*i\_ext\_rst\_n*) при этом игнорируется.

После того, как напряжение микросхемы установилось на рабочем уровне, сигналы сброса от блоков BOR и POR переходят в неактивное состояние. Таймер задержки более не сброшен. Он отсчитывает 1000 тактов частоты *undiv\_clk* (при этом сброс все еще подан на систему) и останавливается. С этого момента в систему подается значение внешнего сигнала сброса (*i\_ext\_rst*). Внешний сигнал сброса предварительно проходит через простейший фильтр нижних частот (Filter). Фильтр не пропускает на выход изменения сигнала *i\_ext\_rst\_n* короче 5 тактов *undiv\_clk*.

Время работы фильтра - 5 тактов *undiv\_clk*, то есть для того, чтобы сформировать сброс в систему, необходимо продержать уровень сигнала на пине *i\_ext\_rst\_n* в нуле не менее чем 5 тактов *undiv\_clk*. Для того, чтобы сигнал сброса системы перешёл из активного уровня в неактивный, значение на пине *rst\_n\_free* также должно держаться в единице не менее 5 тактов *undiv\_clk*.

Если подать на пин *i\_rst\_bypass* единицу, то внешний сброс *i\_ext\_rst\_n* через мультиплексор MUX минует фильтр и таймер задержки и попадает непосредственно в систему.

Далее сигнал сброса попадает на синхронизатор сброса, который синхронизирует сброс на частоту ANB/APB, на которой работают остальные блоки в системе.

PMM формирует два основных сигнала сброса.

Сброс *rst\_n* - сброс для DM и CMM. Активен, когда активен внешний сброс, либо сброс от POR, либо сброс от BOR. Формирование этого сигнала подробно рассмотрено выше.

Сброс *sys\_rst\_n* - сброс для всей системы, кроме DM и CMM. Активен, когда активен *rst\_n* ИЛИ сброс от WDT ИЛИ сброс от DM ИЛИ сброс от Memory Scrubber.

## Регистры PMM

Смещение от базового адреса блока	Аббревиатура	Описание
0x0	PMM_CTRL	Регистр управления
0x4	PMM_ST	Регистр статуса

### Предупреждение

Регистры PMM сбрасываются по *rst\_n*, то есть не сбрасываются при сбросе от DM, WDT или Memory Scrubber. Это нужно для того, чтобы регистр PMM\_ST мог запоминать, что сброс от WDT, DM или Memory Scrubber произошел.

### PMM\_CTRL [0x00].

Биты	Название	Описание	Доступ	Сброс
31:1	-	Резерв	R	0
0	SLEEP_DEEP	значение этого бита определяет, в какой режим сна перейдет система при выполнении процессором инструкции WFI: <ul style="list-style-type: none"> <li>0 - в режим «Сон процессора»;</li> <li>1 - в режим «Глубокий сон».</li> </ul>	R/W	0

### PMM\_ST [0x04]. Регистр очищается при чтении.

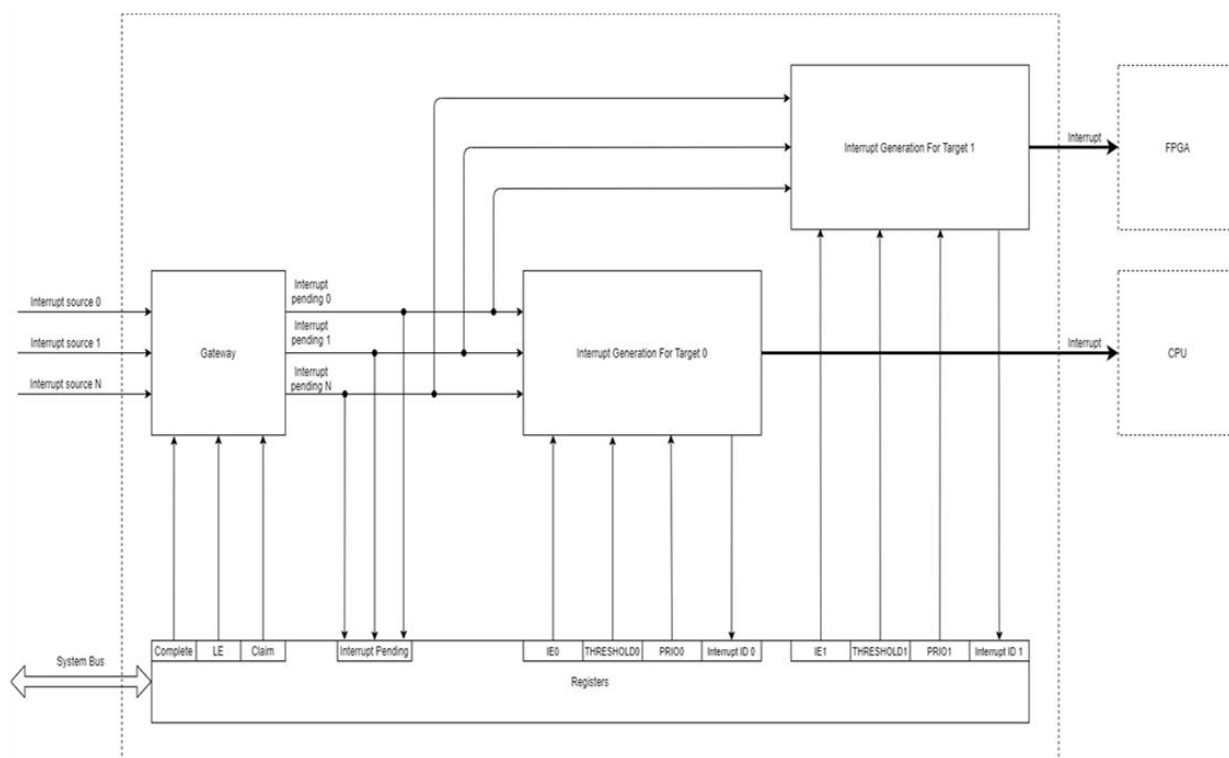
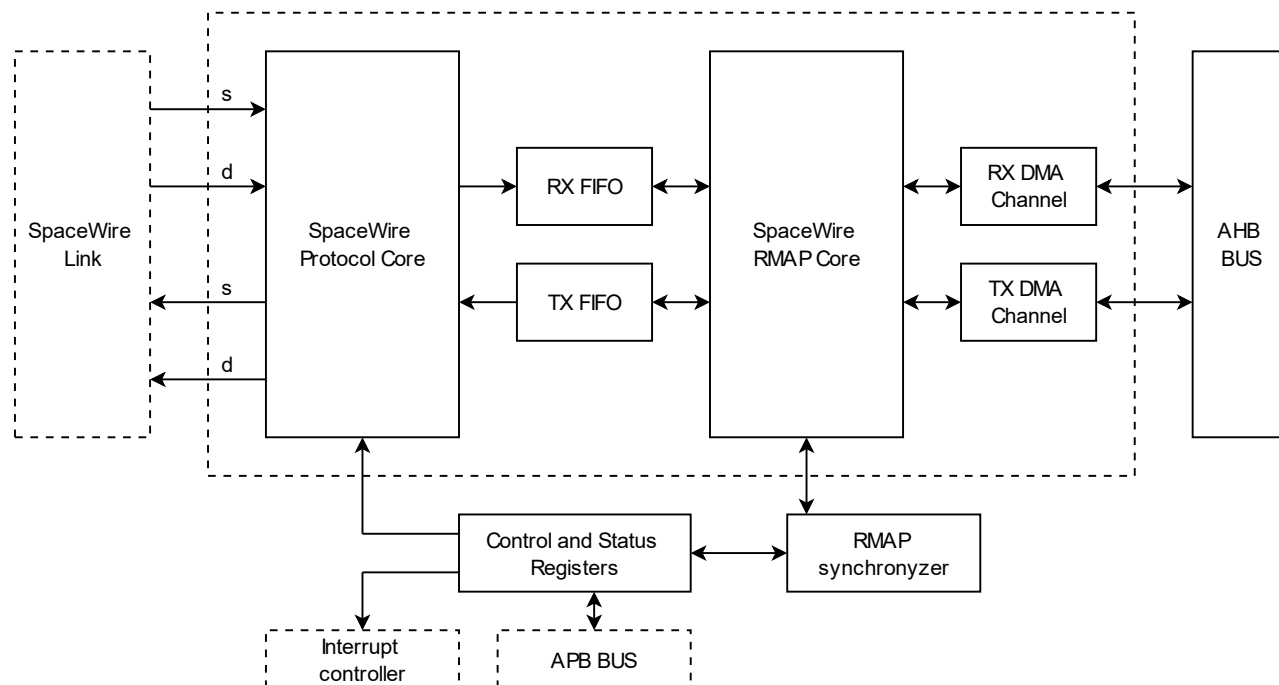
Биты	Название	Описание	Доступ	Сброс
31:3	-	Резерв	R	0
2	SCRUB_RST	сброс от Memory Scrubber: <ul style="list-style-type: none"> <li>1 - зафиксирован сброс от Memory Scrubber.</li> </ul>	RC	0



Биты	Название	Описание	Доступ	Сброс
1	DM_RST	сброс от модуля отладки: • 1 - зафиксирован сброс от модуля отладки.	RC	0
0	WDT_RST	сброс по сторожевому таймеру: • 1 - зафиксирован сброс по сторожевому таймеру.	RC	0

## Модуль PLIC

Модуль PLIC (Platform-level Interrupt Controller) – это контроллер прерываний системы. PLIC принимает сигналы прерывания от всех периферийных устройств, обрабатывает их (приоритет, разрешение прерываний) и посылает сигнал прерывания одной из целей прерывания - процессору или блоку FPGA.



В микросхему интегрирован приемо-передатчик с LVDS уровнями. Для реализации обмена внешних микросхем не требуется.

Всеми функциями аналоговой части микросхемы можно управлять по интерфейсу SpaceWire без задействования CPU.

### ID прерываний

Каждый источник прерывания в системе имеет уникальный ID. ID 0 зарезервирован и означает отсутствие активных прерываний. ID прерываний используются, чтобы цель прерывания (процессорное ядро или блок FPGA) могла определить, какое прерывание активно (прочитав регистр CC), так и чтобы подтвердить окончание обработки данного прерывания (записав в регистр CC).

ID прерывания	Устройство	Тип прерывания
0	Активные прерывания отсутствуют	
1	GPIOA	Зависит от настройки GPIO (регистр INTTYPE)
2	GPIOB	Зависит от настройки GPIO (регистр INTTYPE)
3	SCRUBBER	По уровню
4	CMM	По фронту
5	FPGA	Зависит от логики, реализованной в FPGA
6	WDT	По уровню
7	STIMER	По фронту
8	TIMER0	По фронту
9	TIMER1	По фронту
10	TIMER2	По фронту
11	UART0	По фронту
12	UART1	По фронту
13	SPI0	По фронту
14	SPI1	По фронту
15	I2C0	По фронту
16	I2C1	По фронту
17	SPW0	По фронту
18	SPW1	По фронту
19	MILSTD0	По уровню
20	MILSTD1	По уровню
21	CAN0	По уровню
22	CAN1	По уровню
23	OWI0	По фронту
24	OWI1	По фронту

### Прерывания

#### Разрешение прерываний

Каждый бит в регистре IE управляет соответствующим источником прерываний: 0 - прерывание от этого источника запрещено, 1 - прерывание от этого источника разрешено. Если прерывание от данного источника запрещено, прерывание от этого источника не вызовет прерывание цели. При этом прерывание все же будет фиксироваться в регистре IP.

Если прерывание разрешено в регистре IE, то при его возникновении при чтении регистра CC будет выдано ID этого прерывания (если нет других активных прерываний с более высоким приоритетом). Но это не обязательно означает, что возникновение этого прерывания вызовет прерывание цели. Для того, чтобы прерывание вызвало прерывание цели, должно быть выполнено еще одно условие: его приоритет должен быть выше текущего порога приоритета этой цели.

### Приоритет и порог приоритета

Каждый из источников прерываний имеет настраиваемый приоритет от 0 до 7. Приоритет для каждого из источников настраивается записью в соответствующий регистр PRIO.

Записью в регистр THRESHOLD можно задать текущий порог приоритета прерываний. Когда PLIC фиксирует сигнал прерывания от одного из источников, он посылает сигнал прерывания к цели только если приоритет этого прерывания строго больше порога приоритета этой цели. Стоит отметить, что, таким образом, прерывания с приоритетом 0 никогда не вызовут прерывания цели. Также стоит отметить, что если порог прерываний цели установлен на максимальное значение (7), то никакие источники не будут вызывать прерывание этой цели.

### Входной блок(Gateway)

Входной блок PLIC (Gateway) принимает сигналы прерываний от источников прерываний и переводит их в стандартный формат, использующийся внутри PLIC. Сигналы прерываний делятся на 2 типа: по уровню (прерывание фиксируется, когда сигнал держится в 1) и по фронту (прерывание фиксируется, когда сигнал переходит из 0 в 1). Для каждого из источников можно задать тип прерывания записью в регистр LE.

Различные блоки в системе имеют различный тип сигнала прерывания.

Когда входной блок фиксирует прерывание, в регистре IP соответствующий этому источнику бит выставляется в 1. Когда цель подтверждает, что начала обработку этого прерывания (вычитав ID этого прерывания из регистра CC), бит в регистре IP сбрасывается в 0. Бит в регистре IP не выставится в 1 снова, пока цель не подтвердит завершение обработки прерывания (записав ID этого прерывания в регистр CC). До этого момента любые прерывания от данного источника игнорируются. В частности, это означает, что для сигналов прерываний по фронту после первого фронта (который приводит к выставлению бита в регистре IP) все последующие фронты будут проигнорированы. Подсчет проигнорированных фронтов (как описано в спецификации RISC-V) не поддерживается.

Следует отметить, что для прерываний по уровню если сигнал прерывания перейдет в 0 после выставления бита в регистре IP в 1, то бит в регистре IP будет оставаться в 1. Сбросить его в 0 может только цель, подтвердив, что начал обработку этого прерывания (вычитав ID этого прерывания из регистра CC). Если возможно, цель прерывания должна проверять, по-прежнему ли нуждается прерывание в обработке (прочитав статусный регистр периферийного устройства, которое сгенерировало прерывание).

### Подтверждение начала/окончания обработки прерывания

#### Подтверждение начала обработки прерывания

Цель подтверждает начало обработки прерывания чтением регистра CC. У каждой цели свой регистр CC. Значение, которое возвращается при чтении этого регистра - это ID активного прерывания с наибольшим приоритетом. Если два активных прерывания имеют один и тот же приоритет, PLIC возвращает прерывание с наименьшим ID. Значение порога приоритета при этом не играет роли (порог влияет только на то, вызывает ли прерывание от данного источника прерывание цели), поэтому при чтении регистра CC могут быть прочитаны ID с приоритетом меньше THRESHOLD. Если активных прерываний нет, чтение регистра CC возвращает 0.

Если один источник прерывания включен в регистрах IE сразу двух целей и приоритет этого источника выше THRESHOLD обеих целей, то сигнал прерывания будет послан обоим целям. В этом случае происходит «гонка» между двумя целями - первая цель, которая читает свой регистр CC, вычитает ID прерывания, а вторая вычитает 0, показывающий отсутствие прерываний. Не рекомендуется настраивать PLIC таким образом! Рекомендуется включать каждый источник

прерывания только в одном из регистров IE.

Подтверждение окончания обработки прерывания

После того, как цель подтверждает начало обработки прерывания, соответствующий бит в регистре IP сбрасывается в 0, вне зависимости от значения на линии прерывания. До того момента, как цель подтвердит окончание обработки прерывания, бит в регистре IP не выставится в 1 снова. Цель подтверждает окончание обработки записью ID прерывания в регистр CC. При этом PLIC снова становится восприимчив к прерыванию от устройства с данным ID.

### С-функции для работы с PLIC

При работе с PLIC используйте файлы `pllc.c` и `pllc.h` из примеров программ. Эти файлы содержат функции для инициализации PLIC, разрешения прерываний, задания порога приоритета и пример функции-обработчика прерывания.

Также при работе с прерываниями используйте файлы `isr.c`, `isr.h`, `irq.c`, `irq.h`.

### Карта регистров

Смещение от базового адреса блока	Аббревиатура	Название регистра	Доступ	Описание
Регистры управления источниками прерываний				
0x00	IP	Interrupt Pending	RO	Активные на данный момент прерывания
0x04	LE	Interrupt Source Type (Level/Edge)	RW	Тип сигнала (фронт/уровень) для каждого из прерываний
0x0C	PRI01	Interrupt Priority	RW	Приоритет прерывания GPIOA
0x10	PRI02	Interrupt Priority	RW	Приоритет прерывания GPIOB
0x14	PRI03	Interrupt Priority	RW	Приоритет прерывания SCRUBBER
0x18	PRI04	Interrupt Priority	RW	Приоритет прерывания CMM
0x1C	PRI05	Interrupt Priority	RW	Приоритет прерывания FPGA
0x20	PRI06	Interrupt Priority	RW	Приоритет прерывания WDT
0x24	PRI07	Interrupt Priority	RW	Приоритет прерывания STIMER
0x28	PRI08	Interrupt Priority	RW	Приоритет прерывания TIMER0
0x2C	PRI09	Interrupt Priority	RW	Приоритет прерывания TIMER1
0x30	PRI010	Interrupt Priority	RW	Приоритет прерывания TIMER2
0x34	PRI011	Interrupt Priority	RW	Приоритет прерывания UART0
0x38	PRI012	Interrupt Priority	RW	Приоритет прерывания UART1
0x3C	PRI013	Interrupt Priority	RW	Приоритет прерывания SPI0
0x40	PRI014	Interrupt Priority	RW	Приоритет прерывания SPI1
0x44	PRI015	Interrupt Priority	RW	Приоритет прерывания I2C0
0x48	PRI016	Interrupt Priority	RW	Приоритет прерывания I2C1
0x4C	PRI017	Interrupt Priority	RW	Приоритет прерывания SPW0
0x50	PRI018	Interrupt Priority	RW	Приоритет прерывания SPW1
0x54	PRI019	Interrupt Priority	RW	Приоритет прерывания MILSTD0
0x58	PRI020	Interrupt Priority	RW	Приоритет прерывания MILSTD1
0x5C	PRI021	Interrupt Priority	RW	Приоритет прерывания CAN0

Смещение от базового адреса блока	Аббревиатура	Название регистра	Доступ	Описание
0x60	PRIO22	Interrupt Priority	RW	Приоритет прерывания CAN1
0x64	PRIO23	Interrupt Priority	RW	Приоритет прерывания OWI0
0x68	PRIO24	Interrupt Priority	RW	Приоритет прерывания OWI1
Регистры управление генерацией прерывания для CPU				
0x100	CPU_IE	Interrupt Enable (CPU)	RW	Разрешение прерываний для CPU
0x104	CPU_THRESHOLD	Threshold of Priority (CPU)	RW	Порог приоритета прерываний для CPU
0x108	CPU_CC	Claim / Complete (CPU)	RW	Чтение выдает ID активного прерывания CPU, запись подтверждает окончание CPU обработки прерывания
0x10C	CPU_MSIP	Machine-mode Software Interrupt Pending (CPU)	RW	Генерация программного прерывания для CPU
Регистры управления генерацией прерывания для FPGA				
0x200	FPGA_IE	Interrupt Enable (FPGA)	RW	Разрешение прерываний для FPGA
0x204	FPGA_THRESHOLD	Threshold of Priority (FPGA)	RW	Порог приоритета прерываний для FPGA
0x208	FPGA_CC	Claim / Complete (FPGA)	RW	Чтение выдает ID активного прерывания FPGA, запись подтверждает окончание FPGA обработки прерывания
0x20C	FPGA_MSIP	Machine-mode Software Interrupt Pending (FPGA)	RW	Генерация программного прерывания для FPGA

**Interrupt Pending (IP)**

Биты	Название	Описание	Доступ	Сброс
31:25			R	0
24	OWI1_P		R	0
23	OWI0_P		R	0
22	CAN1_P		R	0
21	CAN0_P		R	0
20	MILSTD1_P		R	0
19	MILSTD0_P		R	0
18	SPW1_P		R	0
17	SPW0_P		R	0
16	I2C1_P		R	0
15	I2C0_P		R	0
14	SPI1_P		R	0
13	SPI0_P		R	0
12	UART1_P		R	0
11	UART0_P		R	0

Биты	Название	Описание	Доступ	Сброс
10	TIMER2_P		R	0
9	TIMER1_P		R	0
8	TIMER0_P		R	0
7	STIMER_P		R	0
6	WDT_P		R	0
5	FPGA_P		R	0
4	CMM_P		R	0
3	SCRUB_P		R	0
2	GPIOB_P		R	0
1	GPIOA_P		R	0
0			R	0

Если бит в этом регистре выставлен в 1, это означает, что зафиксировано прерывание от соответствующего блока.

Бит в регистре IP сбрасывается в 0, когда цель прерывания вычитывает соответствующее прерыванию ID из регистра CC.

#### Interrupt Source Type Level / Edge (LE).

Биты	Название	Описание	Доступ	Сброс
31:25			R	0
24	OWI1_LE		R/W	0
23	OWI0_LE		R/W	0
22	CAN1_LE		R/W	0
21	CAN0_LE		R/W	0
20	MILSTD1_LE		R/W	0
19	MILSTD0_LE		R/W	0
18	SPW1_LE		R/W	0
17	SPW0_LE		R/W	0
16	I2C1_LE		R/W	0
15	I2C0_LE		R/W	0
14	SPI1_LE		R/W	0
13	SPI0_LE		R/W	0
12	UART1_LE		R/W	0
11	UART0_LE		R/W	0
10	TIMER2_LE		R/W	0
9	TIMER1_LE		R/W	0
8	TIMER0_LE		R/W	0
7	STIMER_LE		R/W	0
6	WDT_LE		R/W	0
5	FPGA_LE		R/W	0
4	CMM_LE		R/W	0
3	SCRUB_LE		R/W	0
2	GPIOB_LE		R/W	0
1	GPIOA_LE		R/W	0
0			R	0

Каждый бит регистра задает тип сигнала прерывания от соответствующего блока:

0 - прерывание по уровню;

1 - прерывание по фронту.

**Interrupt Priority (PRIO).**

В PLIC существует отдельный регистр PRIO для каждого из источников прерываний.

Биты	Название	Описание	Доступ	Сброс
31:3	Резерв		R	0
2:0	-	PRIO - определяет приоритет соответствующего прерывания, от 0 до 7.	R/W	0

**Interrupt Enable (CPU\_IE, FPGA\_IE).**

В PLIC присутствуют два регистра IE:

CPU\_IE для цели прерывания номер 0 - процессорного ядра;

FPGA\_IE для цели прерывания номер 1 — FPGA.

Биты	Название	Описание	Доступ	Сброс
31:25			R	0
24	OWI1_IE		R/W	0
23	OWI0_IE		R/W	0
22	CAN1_IE		R/W	0
21	CAN0_IE		R/W	0
20	MILSTD1_IE		R/W	0
19	MILSTD0_IE		R/W	0
18	SPW1_IE		R/W	0
17	SPW0_IE		R/W	0
16	I2C1_IE		R/W	0
15	I2C0_IE		R/W	0
14	SPI1_IE		R/W	0
13	SPI0_IE		R/W	0
12	UART1_IE		R/W	0
11	UART0_IE		R/W	0
10	TIMER2_IE		R/W	0
9	TIMER1_IE		R/W	0
8	TIMER0_IE		R/W	0
7	STIMER_IE		R/W	0
6	WDT_IE		R/W	0
5	FPGA_IE		R/W	0
4	CMM_IE		R/W	0
3	SCRUB_IE		R/W	0
2	GPIOB_IE		R/W	0
1	GPIOA_IE		R/W	0
0			R	0

Каждый бит регистра разрешает или запрещает прерывание от соответствующего источника:

0 - прерывание запрещено;

1 - прерывание разрешено.



**Threshold (CPU\_THRESHOLD, FPGA\_THRESHOLD).**

В PLIC присутствуют два регистра THRESHOLD:

- CPU\_THRESHOLD для цели прерывания номер 0 - процессорного ядра;
- FPGA\_THRESHOLD для цели прерывания номер 1 — FPGA.

Биты	Название	Описание	Доступ	Сброс
31:3	Резерв		R	0
2:0	-	THRESHOLD - определяет текущий порог приоритета прерываний, от 0 до 7. Если приоритет прерывания ниже или равен значению THRESHOLD, то это прерывание не вызовет прерывание цели. Однако, это прерывание по-прежнему будет зарегистрировано в регистрах IP и CC, если оно разрешено в регистре IE.	R/W	0

**Claim / Complete (CPU\_CC, FPGA\_CC).**

В PLIC присутствуют два регистра CC:

- CPU\_CC для цели прерывания номер 0 - процессорного ядра;
- FPGA\_CC для цели прерывания номер 1 — FPGA.

Биты	Название	Описание	Доступ	Сброс
31:5	Резерв		R	0
4:0	-	ID - при чтении выдает ID активного на данный момент прерывания с самым высоким приоритетом. Читая этот регистр, цель прерывания подтверждает, что начала обработку этого прерывания, и соответствующий бит в регистре IP очищается. Если прочитан 0, то это означает, что активных прерываний нет. В самом конце обработки прерывания следует записать в этот регистр ID только что обработанного прерывания. Этим цель прерывания подтверждает, что завершила обработку прерывания. При этом PLIC снова становится восприимчив к прерыванию от устройства с данным ID.	R/W	0

**Machine-Mode Software Interrupt Pending (CPU\_MSIP, FPGA\_MSIP):**

В PLIC присутствуют два регистра MSIP:

CPU\_MSIP для цели прерывания номер 0 - процессорного ядра;

FPGA\_MSIP для цели прерывания номер 1 — FPGA.

Биты	Название	Описание	Доступ	Сброс
31:1	Резерв		R	0
0	MSIP	MSIP - запись 1 в этот бит выставляет в 1 сигнал msip (сигнал программного прерывания) для цели. Запись 0 в этот бит сбрасывает сигнал msip в 0.	R/W	0

## FPGA

FPGA -модуль сопряжения программируемой логической матрицей. Конфигурирование матрицы производится через встроенное ОЗУ, не имеющее интерфейса чтения. Запись в ОЗУ осуществляется по системным адресам. Матрица поддерживает режим автоматической загрузки конфигурации. Загрузка может осуществляться как из встроенного ОППЗУ емкостью 97344 байта, так и из внешней памяти.

Сконфигурированный программируемый логический блок может обращаться как к выводам GPIO, так и к внутренней системной шине. При этом возможно управление всеми функциями аналоговой части с помощью FPGA, не задействуя CPU.

Микросхема содержит программируемый логический блок (ПЛИС), интегрированный в цифровое ядро. ПЛИС позволяет осуществлять либо вспомогательные функции, реализуя нестандартные протоколы обмена и обработки данных в помощь CPU, либо контроль аналоговой части без задействования CPU.

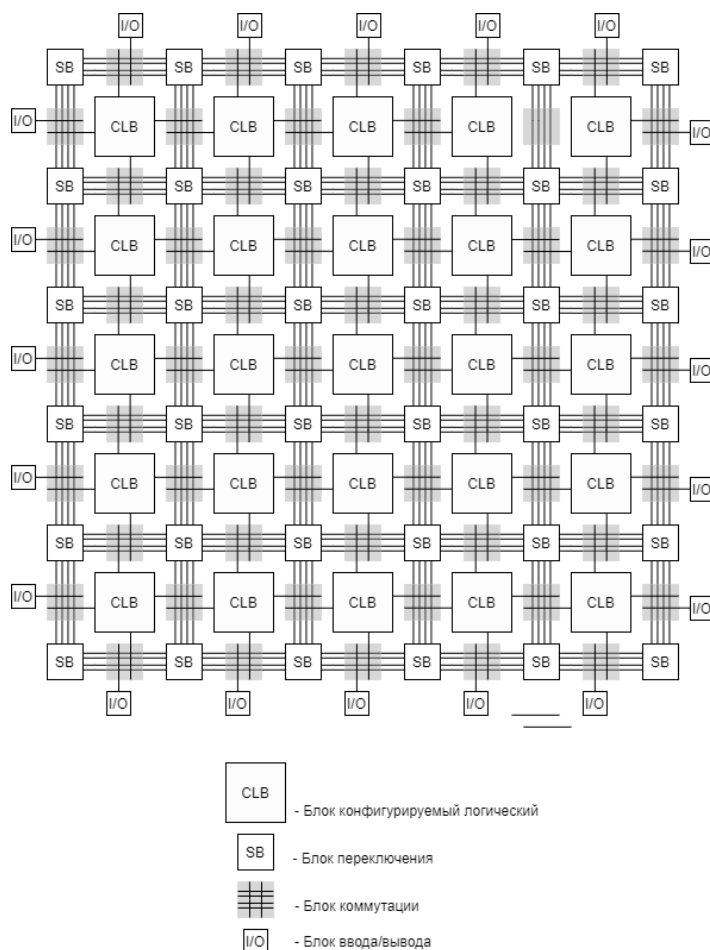


Рисунок 1. Структурная схема программируемого логического блока

Матрица содержит 900 конфигурируемых логических блоков (CLB), по 2 блока LE в каждом. Каждый элемент LE содержит в себе 3-х входовой LUT и D-триггер.

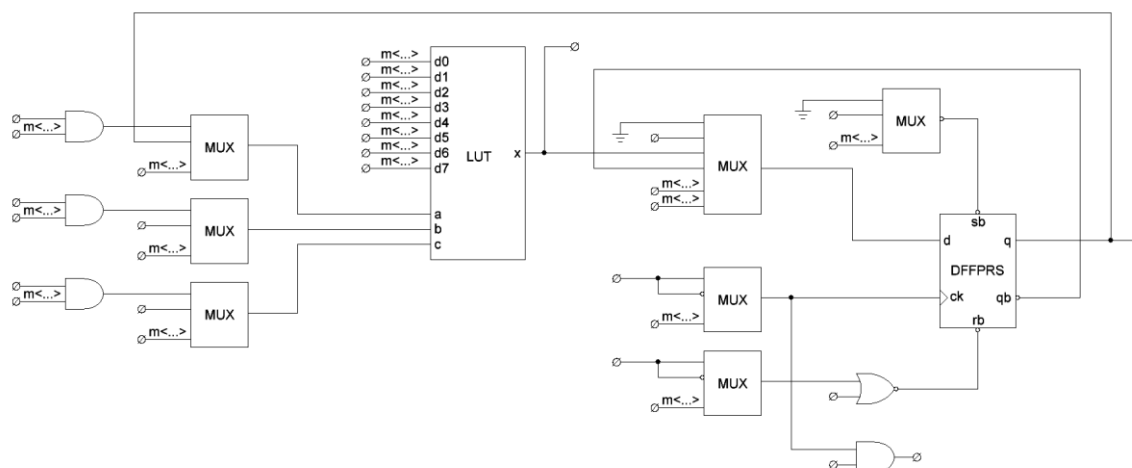


Рисунок 2. Структурная схема элемента LE

Эквивалентная емкость встроенного ПЛИС составляет 10 тысяч 2-х входных вентиляей.

### Функционирование

Конфигурирование ПЛИС производится через встроенное ОЗУ, не имеющее интерфейс чтения. Запись в ОЗУ осуществляется по системным адресам. Однако, физическая запись производится только при записи каждого 36-го двойного слова. Буфер записи содержит регистр разрядностью 1126 бит. Адреса DWORD от 0 до 35 соответствуют адресам внутри данного регистра. Старший адрес записывает 6 младших разрядов. Запись по адресу 35 приводит к перезаписи текущего содержимого регистра в битовую линию внутри ПЛИС - ОЗУ. Строка декодируется старшими битами 35-го адреса.

ПЛИС поддерживает режим автоматической загрузки конфигурации. Загрузка начинается, если в момент выхода из сброса установлен вывод BS\_ENA. Загрузка производится из ПЗУ (0x10008000), если вывод GPIOA[15] равен 0. В противном случае загрузка производится из внешней памяти (0x70018000).

### Режим загрузки из внутренней памяти

В данном режиме FPGA автоматически копирует 97344 байта из ПЗУ, расположенного по адресу 0x10008000, в конфигурационное ОЗУ ПЛИС, расположенное по адресу 0x50198000. Процесс загрузки сигнализируется битом DMA.DA. Окончание процесса сигнализируется битом CFG.CD.

### Режим загрузки из внешней памяти

Перед началом загрузки FPGA производит модификацию регистра CMM.AHBCLKEN устанавливая бит GPIOB. Далее, выждав паузу 32 такта для включения частоты тактирования GPIO, производится запись значений 0x7FFFFFFF и 0x0 в регистры GPIOB.ALTF0 и GPIOB.ALTF1 соответственно. После этого GPIOB сконфигурирован в режим чтения конфигурации из внешней памяти. FPGA автоматически копирует 97344 байта из внешнего ПЗУ, расположенного по адресу 0x70018000 (начальный адрес смещен относительно начального адреса ПЗУ), в конфигурационное ОЗУ ПЛИС, расположенное по адресу 0x50180000.

## Регистры FPGA

Регистр	Адрес	Описание
RAM	0x50180000-0x50197FEF	Конфигурационная ОЗУ
CSR0	0x50197FF0	Регистр подключения к ПЛИС 0
CSR1	0x50197FF4	Регистр подключения к ПЛИС 1
DMA	0x50197FF8	Регистр состояния DMA
CFG	0x50197FFC	Регистр конфигурирования ПЛИС

## CSR0 [0x50197FF0] / CSR1 [0x50197FF4]

Регистры общего назначения, подключенные к выводам ПЛИС. CSR0 содержит 16 бит, CSR1 - 8 бит.

Биты	Название	Описание	Доступ	Сброс
CSR0				
31:16	-	Резерв	R	0
15:0	CSR	Выводы регистра, подключенные к ПЛИС. Значение регистра может обновляться ПЛИС. При одновременной записи и обновлении со стороны ПЛИС, ПЛИС имеет более высокий приоритет. Запись в регистр производится при 1 на выводе csr_wr[0] ПЛИС. Данные для записи на выводах csr_out[15:0]. Биты регистра подключены ко входам csr_in [15:0] ПЛИС.	R/W	0
CSR1				
31:8	-	Резерв	R	0
7:0	CSR	Выводы регистра, подключенные к ПЛИС. Значение регистра может обновляться ПЛИС. При одновременной записи и обновлении со стороны ПЛИС, ПЛИС имеет более высокий приоритет. Запись в регистр производится при 1 на выводе csr_wr[1] ПЛИС. Данные для записи на выводах csr_out[23:16]. Биты регистра подключены ко входам csr_in [23:16] ПЛИС.	R/W	0

## DMA [0x50197FF8]

Регистр состояния DMA

Биты	Название	Описание	Доступ	Сброс
31:2	-	Резерв	R	0
1	ID	Initialization Done. Признак завершения инициализации ОЗУ ПЛИС. После подачи питания ОЗУ ПЛИС находится в неопределенном состоянии. Внутренняя схема с большей вероятностью инициализирует память в 0. После выхода из сброса DMA заполняет всю ОЗУ ПЛИС значением 0. Далее происходит либо инициализация ПЛИС, или ПЛИС остается в начальном состоянии (в зависимости от вывода BS_ENA).	R	0
0	DA	DMA Active. Признак активности DMA. DMA запускается после снятия сброса с системы, если выводы BOOTSTRAP требуют аппаратной загрузки прошивки ПЛИС. Устанавливается по окончании загрузки прошивки и не может быть переопределен программно (только сбросом системы). Если прошивка не требуется, бит остается сброшенным.	R	0

## CFG [0x50197FFC]

Регистр конфигурирования ПЛИС

Биты	Название	Описание	Доступ	Сброс
31	SE	Scrubbing Enable. Управление режимом периодического обновления содержимого ОЗУ содержимым ПЗУ. Обновление начинается, если DMA находится в неактивном состоянии (DMA.DA = 0) и ПЛИС проинициализирована (DMA.ID = 1). Не рекомендуется применять режим периодического обновления, если ПЛИС не была предварительно сконфигурирована (DMA.CD = 0). • 0 - режим обновления не активен; • 1 - активировать режим обновления.	R/W	0
30	SS	Scrubbing Source. Источник данных для периодической перезаписи конфигурационной ОЗУ ПЛИС. • 0 - внутреннее ПЗУ (0x10008000); • 1 - внешнее ПЗУ (0x50198000).	R/W	0
29:24	-	Резерв.	R	0
23:16	INTRVL	Интервал (в количестве тактов АНВ) между последовательными запросами копирования DWORD из ПЗУ в ОЗУ ПЛИС. Позволяет настроить интервал скрабирования, при начальной загрузке равен 0 для максимальной скорости начальной загрузки.	R/W	0
15:1	-	Резерв.	R	0
0	CD	Configuration Done. Признак окончания конфигурирования ПЛИС. Устанавливается аппаратно по окончании автоматической загрузки прошивки. В случае ручной прошивки, следует установить программно. При сброшенном бите тактирование на ПЛИС не подается и сброс в активном состоянии.	R/W	0

## Пользовательские выводы FPGA

ПЛИС имеет следующие выводы, зарезервированные для использования совместно с аппаратными ресурсами микроконтроллера. В процессе конфигурации (CFG.CD = 0) на входы ПЛИС FROM\_PADx поступает логический 0; выходы TO\_PADx выдают в систему логический 0 (кроме io\_out\_oe, выдающего 1).

Название входа	Название выхода	Назначение входа	Назначение выхода
FROM_PAD_CLK1		hclk (AHB clock)	
FROM_PAD_CLK2		pclk (APB clock)	
FROM_PAD_CLK3		FPGA0	
FROM_PAD_CLK4		FPGA1	
FROM_PAD1	TO_PAD1	irq_in	dac_ctrl[0]
FROM_PAD2	TO_PAD2	irq_vec[0]	dac_ctrl[1]
FROM_PAD3	TO_PAD3	irq_vec[1]	dac_ctrl[2]
FROM_PAD4	TO_PAD4	irq_vec[2]	dac_ctrl[3]
FROM_PAD5	TO_PAD5	irq_vec[3]	dac_ctrl[4]
FROM_PAD6	TO_PAD6	irq_vec[4]	dac_ctrl[5]
FROM_PAD7	TO_PAD7		o_irq

Название входа	Название выхода	Назначение входа	Назначение выхода
FROM_PAD8	TO_PAD8		csr_wr[0]
FROM_PAD9	TO_PAD9		csr_wr[1]
FROM_PAD10	TO_PAD10	csr_in[0]	dac_ctrl[6]
FROM_PAD11	TO_PAD11		csr_out[0]
FROM_PAD12	TO_PAD12	csr_in[1]	dac_ctrl[7]
FROM_PAD13	TO_PAD13		csr_out[1]
FROM_PAD14	TO_PAD14	csr_in[2]	dac_ctrl[8]
FROM_PAD15	TO_PAD15		csr_out[2]
FROM_PAD16	TO_PAD16	csr_in[3]	dac_ctrl[9]
FROM_PAD17	TO_PAD17		csr_out[3]
FROM_PAD18	TO_PAD18	csr_in[4]	dac_ctrl[10]
FROM_PAD19	TO_PAD19		csr_out[4]
FROM_PAD20	TO_PAD20	csr_in[5]	dac_ctrl[11]
FROM_PAD21	TO_PAD21		csr_out[5]
FROM_PAD22	TO_PAD22	csr_in[6]	
FROM_PAD23	TO_PAD23		csr_out[6]
FROM_PAD24	TO_PAD24	csr_in[7]	
FROM_PAD25	TO_PAD25		csr_out[7]
FROM_PAD26	TO_PAD26	csr_in[8]	
FROM_PAD27	TO_PAD27		csr_out[8]
FROM_PAD28	TO_PAD28	csr_in[9]	
FROM_PAD29	TO_PAD29		csr_out[9]
FROM_PAD30	TO_PAD30	csr_in[10]	
FROM_PAD31	TO_PAD31		csr_out[10]
FROM_PAD32	TO_PAD32	csr_in[11]	
FROM_PAD33	TO_PAD33		csr_out[11]
FROM_PAD34	TO_PAD34	csr_in[12]	
FROM_PAD35	TO_PAD35		csr_out[12]
FROM_PAD36	TO_PAD36	csr_in[13]	
FROM_PAD37	TO_PAD37		csr_out[13]
FROM_PAD38	TO_PAD38	csr_in[14]	
FROM_PAD39	TO_PAD39		csr_out[14]
FROM_PAD40	TO_PAD40	csr_in[15]	
FROM_PAD41	TO_PAD41		csr_out[15]
FROM_PAD42	TO_PAD42		ahb_addr[0]
FROM_PAD43	TO_PAD43		ahb_addr[1]
FROM_PAD44	TO_PAD44		ahb_addr[2]
FROM_PAD45	TO_PAD45		ahb_addr[3]
FROM_PAD46	TO_PAD46		ahb_addr[4]

Название входа	Название выхода	Назначение входа	Назначение выхода
FROM_PAD47	TO_PAD47		ahb_addr[5]
FROM_PAD48	TO_PAD48		ahb_addr[6]
FROM_PAD49	TO_PAD49		ahb_addr[7]
FROM_PAD50	TO_PAD50		ahb_wdata[0]
FROM_PAD51	TO_PAD51		ahb_wdata[1]
FROM_PAD52	TO_PAD52		ahb_wdata[2]
FROM_PAD53	TO_PAD53		ahb_wdata[3]
FROM_PAD54	TO_PAD54		ahb_wdata[4]
FROM_PAD55	TO_PAD55		ahb_wdata[5]
FROM_PAD56	TO_PAD56		ahb_wdata[6]
FROM_PAD57	TO_PAD57		ahb_wdata[7]
FROM_PAD58	TO_PAD58	ahb_rdata[0]	
FROM_PAD59	TO_PAD59	ahb_rdata[1]	
FROM_PAD60	TO_PAD60	ahb_rdata[2]	
FROM_PAD61	TO_PAD61	ahb_rdata[3]	
FROM_PAD62	TO_PAD62	ahb_rdata[4]	
FROM_PAD63	TO_PAD63	ahb_rdata[5]	
FROM_PAD64	TO_PAD64	ahb_rdata[6]	
FROM_PAD65	TO_PAD65	ahb_rdata[7]	
FROM_PAD66	TO_PAD66		ahb_wr
FROM_PAD67	TO_PAD67		ahb_rd
FROM_PAD68	TO_PAD68		ahb_addr[8]
FROM_PAD69	TO_PAD69		ahb_addr[9]
FROM_PAD70	TO_PAD70		ahb_addr[10]
FROM_PAD71	TO_PAD71		ahb_addr[11]
FROM_PAD72	TO_PAD72		ahb_addr[12]
FROM_PAD73	TO_PAD73		ahb_addr[13]
FROM_PAD74	TO_PAD74		ahb_addr[14]
FROM_PAD75	TO_PAD75		ahb_addr[15]
FROM_PAD76	TO_PAD76		ahb_wdata[8]
FROM_PAD77	TO_PAD77		ahb_wdata[9]
FROM_PAD78	TO_PAD78		ahb_wdata[10]
FROM_PAD79	TO_PAD79		ahb_wdata[11]
FROM_PAD80	TO_PAD80		ahb_wdata[12]
FROM_PAD81	TO_PAD81		ahb_wdata[13]
FROM_PAD82	TO_PAD82		ahb_wdata[14]
FROM_PAD83	TO_PAD83		ahb_wdata[15]
FROM_PAD84	TO_PAD84	ahb_rdata[8]	
FROM_PAD85	TO_PAD85	ahb_rdata[9]	

Название входа	Название выхода	Назначение входа	Назначение выхода
FROM_PAD86	TO_PAD86	ahb_rdata[10]	
FROM_PAD87	TO_PAD87	ahb_rdata[11]	
FROM_PAD88	TO_PAD88	ahb_rdata[12]	
FROM_PAD89	TO_PAD89	ahb_rdata[13]	
FROM_PAD90	TO_PAD90	ahb_rdata[14]	
FROM_PAD91	TO_PAD91	ahb_rdata[15]	
FROM_PAD92	TO_PAD92		ahb_addr[16]
FROM_PAD93	TO_PAD93		ahb_addr[17]
FROM_PAD94	TO_PAD94		ahb_addr[18]
FROM_PAD95	TO_PAD95		ahb_addr[19]
FROM_PAD96	TO_PAD96		ahb_addr[20]
FROM_PAD97	TO_PAD97		ahb_addr[21]
FROM_PAD98	TO_PAD98		ahb_addr[22]
FROM_PAD99	TO_PAD99		ahb_addr[23]
FROM_PAD100	TO_PAD100		ahb_wdata[16]
FROM_PAD101	TO_PAD101		ahb_wdata[17]
FROM_PAD102	TO_PAD102		ahb_wdata[18]
FROM_PAD103	TO_PAD103		ahb_wdata[19]
FROM_PAD104	TO_PAD104		ahb_wdata[20]
FROM_PAD105	TO_PAD105		ahb_wdata[21]
FROM_PAD106	TO_PAD106		ahb_wdata[22]
FROM_PAD107	TO_PAD107		ahb_wdata[23]
FROM_PAD108	TO_PAD108	ahb_rdata[16]	
FROM_PAD109	TO_PAD109	ahb_rdata[17]	
FROM_PAD110	TO_PAD110	ahb_rdata[18]	
FROM_PAD111	TO_PAD111	ahb_rdata[19]	
FROM_PAD112	TO_PAD112	ahb_rdata[20]	
FROM_PAD113	TO_PAD113	ahb_rdata[21]	
FROM_PAD114	TO_PAD114	ahb_rdata[22]	
FROM_PAD115	TO_PAD115	ahb_rdata[23]	
FROM_PAD116	TO_PAD116		ahb_addr[24]
FROM_PAD117	TO_PAD117		ahb_addr[25]
FROM_PAD118	TO_PAD118		ahb_addr[26]
FROM_PAD119	TO_PAD119		ahb_addr[27]
FROM_PAD120	TO_PAD120		ahb_addr[28]
FROM_PAD121	TO_PAD121		ahb_addr[29]
FROM_PAD122	TO_PAD122		ahb_addr[30]
FROM_PAD123	TO_PAD123		ahb_addr[31]
FROM_PAD124	TO_PAD124		ahb_wdata[24]



Название входа	Название выхода	Назначение входа	Назначение выхода
FROM_PAD125	TO_PAD125		ahb_wdata[25]
FROM_PAD126	TO_PAD126		ahb_wdata[26]
FROM_PAD127	TO_PAD127		ahb_wdata[27]
FROM_PAD128	TO_PAD128		ahb_wdata[28]
FROM_PAD129	TO_PAD129		ahb_wdata[29]
FROM_PAD130	TO_PAD130		ahb_wdata[30]
FROM_PAD131	TO_PAD131		ahb_wdata[31]
FROM_PAD132	TO_PAD132	ahb_rdata[24]	
FROM_PAD133	TO_PAD133	ahb_rdata[25]	
FROM_PAD134	TO_PAD134	ahb_rdata[26]	
FROM_PAD135	TO_PAD135	ahb_rdata[27]	
FROM_PAD136	TO_PAD136	ahb_rdata[28]	
FROM_PAD137	TO_PAD137	ahb_rdata[29]	
FROM_PAD138	TO_PAD138	ahb_rdata[30]	
FROM_PAD139	TO_PAD139	ahb_rdata[31]	
FROM_PAD140	TO_PAD140	csr_in[16]	
FROM_PAD141	TO_PAD141		csr_out[16]
FROM_PAD142	TO_PAD142	ahb_ready	
FROM_PAD143	TO_PAD143	ahb_error	
FROM_PAD144	TO_PAD144	csr_in[17]	
FROM_PAD145	TO_PAD145		csr_out[17]
FROM_PAD146	TO_PAD146	csr_in[18]	
FROM_PAD147	TO_PAD147		csr_out[18]
FROM_PAD148	TO_PAD148	csr_in[19]	
FROM_PAD149	TO_PAD149		csr_out[19]
FROM_PAD150	TO_PAD150	csr_in[20]	
FROM_PAD151	TO_PAD151		csr_out[20]
FROM_PAD152	TO_PAD152	csr_in[21]	
FROM_PAD153	TO_PAD153		csr_out[21]
FROM_PAD154	TO_PAD154	csr_in[22]	
FROM_PAD155	TO_PAD155		csr_out[22]
FROM_PAD156	TO_PAD156	csr_in[23]	
FROM_PAD157	TO_PAD157		csr_out[23]
FROM_PAD158	TO_PAD158		io_out_oe[15]
FROM_PAD159	TO_PAD159		FPGA15
FROM_PAD160	TO_PAD160	FPGA15	
FROM_PAD161	TO_PAD161		io_out_oe[14]
FROM_PAD162	TO_PAD162		FPGA14
FROM_PAD163	TO_PAD163	FPGA14	

Название входа	Название выхода	Назначение входа	Назначение выхода
FROM_PAD164	TO_PAD164		io_out_oe[13]
FROM_PAD165	TO_PAD165		FPGA13
FROM_PAD166	TO_PAD166	FPGA13	
FROM_PAD167	TO_PAD167		io_out_oe[12]
FROM_PAD168	TO_PAD168		FPGA12
FROM_PAD169	TO_PAD169	FPGA12	
FROM_PAD170	TO_PAD170		io_out_oe[11]
FROM_PAD171	TO_PAD171		FPGA11
FROM_PAD172	TO_PAD172	FPGA11	
FROM_PAD173	TO_PAD173		io_out_oe[10]
FROM_PAD174	TO_PAD174		FPGA10
FROM_PAD175	TO_PAD175	FPGA10	
FROM_PAD176	TO_PAD176		io_out_oe[9]
FROM_PAD177	TO_PAD177		FPGA9
FROM_PAD178	TO_PAD178	FPGA9	
FROM_PAD179	TO_PAD179		io_out_oe[8]
FROM_PAD180	TO_PAD180		FPGA8
FROM_PAD181	TO_PAD181	FPGA8	
FROM_PAD182	TO_PAD182		io_out_oe[7]
FROM_PAD183	TO_PAD183		FPGA7
FROM_PAD184	TO_PAD184	FPGA7	
FROM_PAD185	TO_PAD185		io_out_oe[6]
FROM_PAD186	TO_PAD186		FPGA6
FROM_PAD187	TO_PAD187	FPGA6	
FROM_PAD188	TO_PAD188		io_out_oe[5]
FROM_PAD189	TO_PAD189		FPGA5
FROM_PAD190	TO_PAD190	FPGA5	
FROM_PAD191	TO_PAD191		io_out_oe[4]
FROM_PAD192	TO_PAD192		FPGA4
FROM_PAD193	TO_PAD193	FPGA4	
FROM_PAD194	TO_PAD194		io_out_oe[3]
FROM_PAD195	TO_PAD195		FPGA3
FROM_PAD196	TO_PAD196	FPGA3	
FROM_PAD197	TO_PAD197		io_out_oe[2]
FROM_PAD198	TO_PAD198		FPGA2
FROM_PAD199	TO_PAD199	FPGA2	
FROM_PAD200	TO_PAD200		io_out_oe[1]
FROM_PAD201	TO_PAD201		FPGA1
FROM_PAD202	TO_PAD202		dac_ctrl[12]

Название входа	Название выхода	Назначение входа	Назначение выхода
FROM_PAD203	TO_PAD203		io_out_oe[0]
FROM_PAD204	TO_PAD204		FPGA0
FROM_PAD205	TO_PAD205		dac_ctrl [13]

#### Использование внешних выводов

Выводы io\_out\_oe определяют направление внешних выводов (0 - выход, 1 - вход) для соответствующих им выводов FPGAх.

При использовании ПЛИС без процессора (CPU\_EN = 0), ПЛИС должна быть сконфигурирована для задания альтернативных функций GPIO. Это накладывает ограничение в виде невозможности использования выводов FPGA0 и FPGA1 для управления асинхронным сбросом ПЛИС.