


УДОСТОВЕРЕН
ЮФКВ.30139-01-УД

УНИВЕРСАЛЬНЫЕ ЭЛЕКТРОННЫЕ МОДУЛИ УЭМ-МК, МВ98.03.
РАСШИРЕННАЯ БИБЛИОТЕКА ФУНКЦИЙ

Руководство программиста
ЮФКВ.30139-01 33 01
(ЮФКВ.30139-01 33 01-001ФЛ)

Листов 166

Литера

Инв.№ подл.	Подп. и дата	Взам.инв.№	Инв.№ дубл.	Подп. и дата
30875	 19.08.2016			

АННОТАЦИЯ

Универсальные электронные модули мультиплексного канала совмещают функции контроллера шины, оконечного устройства, монитора шины, тестера протокола и имитатора дополнительных оконечных устройств на шине ГОСТ Р 52070-2003 [1].

Расширенная библиотека функций предназначена для разработки прикладных программ, использующих УЭМ-МК или МВ98.03 для решения задач тестирования и отладки интерфейсов ГОСТ Р 52070-2003 и систем на их основе, мониторинга и анализа процессов передачи данных по шине ГОСТ Р 52070-2003. Расширенная библиотека функций обеспечивает использование прикладными программами всех возможностей аппаратуры УЭМ-МК, МВ98.03, при этом управление аппаратурой производится в терминах протокола ГОСТ Р 52070-2003.

Настоящий документ является руководством программиста по расширенной библиотеке функций.

СОДЕРЖАНИЕ

АННОТАЦИЯ.....	2
НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ.....	4
Указатель групп.....	5
Указатель структур данных.....	6
Список файлов.....	7
Группы.....	8
Вводные и дополнительные сведения.....	8
Коды завершения.....	8
Порядок действий при установлении связи с устройством.....	13
Адресная строка.....	14
Виртуальные устройства.....	14
Командные и ответные сегменты.....	14
Параметры передатчиков и характеристики выходных сигналов.....	15
Типы вносимых ошибок кодирования.....	17
Управление синхронизацией.....	21
Определения примитивных типов.....	23
Установление и разрыв связи с устройством.....	25
Действия с дескрипторами.....	28
Параметры конфигурации УЭМ.....	31
Описание параметров.....	32
Значения параметров.....	41
Параметры интервалов времени.....	55
Встроенный счетчик времени.....	58
Определения типов данных для КШ, ОУ, МШ.....	61
Функции КШ.....	68
Заполнение образа командного сегмента в ОЗУ ПЭВМ.....	68
Внесение ошибок состава сообщения.....	70
Создание и настройка командных сегментов.....	71
Создание и настройка кадров и программы КШ.....	81
Запуск и остановка КШ.....	87
Передача сообщений.....	88
Функции ОУ.....	90
Функции МШ.....	102
Запуск и остановка.....	104
Служебные функции.....	108
Формирование и разбор командных и ответных слов.....	111
Структуры данных.....	118
UEM_VM_MESSAGE.....	118
UEM_CMD_SEG.....	121
UEM_DATA.....	123
UEM_RAW_VM_MESSAGE.....	124
UEM_RESP_SEG.....	125
UEM_SEGMENT_DESCR.....	126
UEM_TIME_TAG.....	128
Файлы.....	130
uem.h.....	130
Алфавитный указатель.....	150
ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ.....	164
ПЕРЕЧЕНЬ ССЫЛОЧНЫХ ДОКУМЕНТОВ.....	165

НАЗНАЧЕНИЕ И УСЛОВИЯ ПРИМЕНЕНИЯ

Расширенная библиотека функций предназначена для разработки прикладных программ, использующих УЭМ-МК [2], МВ98.03 [3] для решения задач тестирования и отладки интерфейсов ГОСТ Р 52070-2003 [1] и систем на их основе, мониторинга и анализа процессов передачи данных по шине ГОСТ Р 52070-2003. Расширенная библиотека функций обеспечивает использование прикладными программами всех возможностей аппаратуры УЭМ-МК, МВ98.03, при этом управление аппаратурой производится в терминах протокола ГОСТ Р 52070-2003.

Разработка и эксплуатация прикладных программ, использующих расширенную библиотеку, должна выполняться на встроенной ПЭВМ в составе крейта VXI, в котором установлены модули УЭМ-МК или МВ98.03, не менее одного. На ПЭВМ должно быть установлено программное обеспечение VISA, unmbase [4] и unmuem [5, 6].

Далее по тексту под аббревиатурой УЭМ мы будем понимать любой из модулей УЭМ-МК или МВ98.03.

Указатель групп

Группы

Полный список групп.

Вводные и дополнительные сведения	8
Коды завершения	8
Порядок действий при установлении связи с устройством	13
Адресная строка	14
Виртуальные устройства	14
Командные и ответные сегменты	14
Параметры передатчиков и характеристики выходных сигналов	15
Типы вносимых ошибок кодирования	17
Управление синхронизацией	21
Определения примитивных типов	23
Установление и разрыв связи с устройством	25
Действия с дескрипторами	28
Параметры конфигурации УЭМ	31
Описание параметров	32
Значения параметров	41
Параметры интервалов времени	55
Встроенный счетчик времени	58
Определения типов данных для КШ, ОУ, МШ	61
Функции КШ	68
Заполнение образа командного сегмента в ОЗУ ПЭВМ	68
Внесение ошибок состава сообщения	70
Создание и настройка командных сегментов	71
Создание и настройка кадров и программы КШ	81
Запуск и остановка КШ	87
Передача сообщений	88
Функции ОУ	90
Функции МШ	102
Запуск и остановка	104
Служебные функции	108
Формирование и разбор командных и ответных слов	111

Указатель структур данных

Структуры данных

Структуры данных с их кратким описанием.

UEM_BM_MESSAGE (Разобранное сообщение МШ)	118
UEM_CMD_SEG (Образ командного сегмента)	121
UEM_DATA (Блок слов данных)	123
UEM_RAW_BM_MESSAGE (Принятое сообщение в аппаратном формате)	124
UEM_RESP_SEG (Ответный сегмент)	125
UEM_SEGMENT_DESCR (Описатель сегмента в мониторе шины)	126
UEM_TIME_TAG (Формат метки времени)	128

Список файлов

Файлы

Полный список файлов.

**uem.h (Универсальные электронные модули УЭМ-МК, МВ98.03. Расширенная библиотека функций.
Файл заголовков функций) 130**

Группы

Вводные и дополнительные сведения

Введение терминов, необходимые сведения об основных возможностях аппаратуры, типовые сценарии.

Группы

- **Коды завершения**
Результаты выполнения функций.
 - **Порядок действий при установлении связи с устройством**
Описание типового сценария
 - **Адресная строка**
Синтаксис идентификатора инструмента в VISA.
 - **Виртуальные устройства**
Введение термина: виртуальное устройство
 - **Командные и ответные сегменты**
Введение терминов: сегмент, командный сегмент, ответный сегмент
 - **Параметры передатчиков и характеристики выходных сигналов**
Сведения о возможностях аппаратного обеспечения. Соответствие значений параметров конфигурации передатчиков и электрических характеристик выходных сигналов.
 - **Типы вносимых ошибок кодирования**
Сведения о возможностях аппаратного обеспечения в части внесения ошибок кодирования и способы описания (программирования) вносимых ошибок.
 - **Управление синхронизацией**
Сведения об аппаратных средствах синхронизации с внешним оборудованием и об управлении ими.
-

Подробное описание

Введение терминов, необходимые сведения об основных возможностях аппаратуры, типовые сценарии.

Коды завершения

Результаты выполнения функций.

Макросы

- `#define UEM_WARN_OFFSET (0x3FFC0B00L)`
Начальный номер кодов предупреждений.
- `#define UEM_ERROR_OFFSET (_VI_ERROR + UEM_WARN_OFFSET)`
Начальный номер кодов ошибок.
- `#define UEM_ERROR_BAD_PARAM_VALUE (UEM_ERROR_OFFSET + 0)`
Недопустимое значение параметра.
- `#define UEM_ERROR_BAD_PARAM_VALUE_1 (UEM_ERROR_OFFSET + 1)`
Недопустимое значение в параметре 1.
- `#define UEM_ERROR_BAD_PARAM_VALUE_2 (UEM_ERROR_OFFSET + 2)`
Недопустимое значение в параметре 2.

- **#define UEM_ERROR_BAD_PARAM_VALUE_3 (UEM_ERROR_OFFSET + 3)**
Недопустимое значение в параметре 3.
- **#define UEM_ERROR_BAD_PARAM_VALUE_4 (UEM_ERROR_OFFSET + 4)**
Недопустимое значение в параметре 4.
- **#define UEM_ERROR_BAD_PARAM_VALUE_5 (UEM_ERROR_OFFSET + 5)**
Недопустимое значение в параметре 5.
- **#define UEM_ERROR_BAD_PARAM_VALUE_6 (UEM_ERROR_OFFSET + 6)**
Недопустимое значение в параметре 6.
- **#define UEM_ERROR_BAD_PARAM_VALUE_7 (UEM_ERROR_OFFSET + 7)**
Недопустимое значение в параметре 7.
- **#define UEM_ERROR_BAD_PARAM_VALUE_8 (UEM_ERROR_OFFSET + 8)**
Недопустимое значение в параметре 8.
- **#define UEM_ERROR_BAD_PARAM_VALUE_9 (UEM_ERROR_OFFSET + 9)**
Недопустимое значение в параметре 9.
- **#define UEM_ERROR_BAD_PARAM_VALUE_10 (UEM_ERROR_OFFSET + 10)**
Недопустимое значение в параметре 10.
- **#define UEM_ERROR_INV_HANDLE (UEM_ERROR_OFFSET + 11)**
Недействительный дескриптор.
- **#define UEM_ERROR_INV_HANDLE_TYPE (UEM_ERROR_OFFSET + 12)**
Неподходящий тип дескриптора.
- **#define UEM_ERROR_NO_FREE_RAM (UEM_ERROR_OFFSET + 13)**
Недостаточно ОЗУ УЭМ.
- **#define UEM_ERROR_NO_HOST_MEM (UEM_ERROR_OFFSET + 14)**
Недостаточно ОЗУ управляющей ПЭВМ.
- **#define UEM_ERROR_NOT_CONNECTED (UEM_ERROR_OFFSET + 15)**
Нет связи с устройством.
- **#define UEM_ERROR_INPOOL (UEM_ERROR_OFFSET + 16)**
Внутренняя ошибка менеджера памяти.
- **#define UEM_ERROR_BCP_NINST (UEM_ERROR_OFFSET + 17)**
Не установлена программа КШ.
- **#define UEM_ERROR_FORMAT_DISABLED (UEM_ERROR_OFFSET + 18)**
Формат сообщения запрещен конфигурацией УЭМ.
- **#define UEM_ERROR_FORMAT_X_MCODE (UEM_ERROR_OFFSET + 19)**
Формат сообщения несовместим с командой управления.
- **#define UEM_ERROR_NOT_APPLICABLE (UEM_ERROR_OFFSET + 20)**
Действие не применимо к объекту.
- **#define UEM_ERROR_ADDRESS_OUT_OF_RANGE (UEM_ERROR_OFFSET + 21)**
Адрес вне допустимого диапазона.
- **#define UEM_ERROR_NUMBER_OUT_OF_RANGE (UEM_ERROR_OFFSET + 23)**
Номер вне допустимого диапазона.
- **#define UEM_ERROR_BAD_TIMEOUT (UEM_ERROR_OFFSET + 24)**
Недопустимое значение таймута
- **#define UEM_ERROR_BAD_OVERLAY_SOURCE (UEM_ERROR_OFFSET + 25)**
Недопустимые исходные сегменты для наложения.
- **#define UEM_ERROR_WRONG_LOCATION (UEM_ERROR_OFFSET + 26)**
Объект расположен не в том устройстве.

- **#define UEM_ERROR_TOO_MANY_DATAWORDS (UEM_ERROR_OFFSET + 27)**
Слишком много слов данных.
 - **#define UEM_ERROR_MAX_SIZE_EXCEED (UEM_ERROR_OFFSET + 28)**
Превышен максимальный размер.
 - **#define UEM_ERROR_NO_FRAME_APPEND (UEM_ERROR_OFFSET + 29)**
Не добавлен кадр.
 - **#define UEM_ERROR_IN_USE (UEM_ERROR_OFFSET + 30)**
Устройство или объект используются.
 - **#define UEM_ERROR_THREAD_FAULT (UEM_ERROR_OFFSET + 31)**
Ошибки в работе служебной нити.
 - **#define UEM_ERROR_BM_INTERNAL_BUFFER_OVERFLOW (UEM_ERROR_OFFSET + 32)**
Переполнение внутреннего буфера МШ.
 - **#define UEM_ERROR_INC_RESP (UEM_ERROR_OFFSET + 33)**
Несовместимый ответный сегмент.
 - **#define UEM_WARN_NO_NEXT_MESSAGE (UEM_WARN_OFFSET + 0)**
Нет следующего сообщения (в буфере МШ).
 - **#define UEM_WARN_JUST_IN_STATE (UEM_WARN_OFFSET + 1)**
Устройство уже в нужном состоянии; никаких действий не производится.
-

Подробное описание

Результаты выполнения функций.

Нулевой код завершения соответствует нормальному завершению. Коды, меньшие 0, являются кодами ошибок. Коды, большие 0, являются кодами предупреждений и особых ситуаций, не являющихся ошибками.

Следует иметь в виду, что код завершения может быть сформирован не только настоящей расширенной библиотекой, но и нижележащем ПО: драйвером УЭМ, драйвером носителя мезонинов, ПО VISA. По поводу описания смысла этих кодов завершения отсылаем к документации по указанному ПО [4,5,6].

Макросы

#define UEM_WARN_OFFSET (0x3FFC0B00L)

Начальный номер кодов предупреждений.

#define UEM_ERROR_OFFSET (_VI_ERROR + UEM_WARN_OFFSET)

Начальный номер кодов ошибок.

#define UEM_ERROR_BAD_PARAM_VALUE (UEM_ERROR_OFFSET + 0)

Недопустимое значение параметра.

#define UEM_ERROR_BAD_PARAM_VALUE_1 (UEM_ERROR_OFFSET + 1)

Недопустимое значение в параметре 1.

#define UEM_ERROR_BAD_PARAM_VALUE_2 (UEM_ERROR_OFFSET + 2)

Недопустимое значение в параметре 2.

#define UEM_ERROR_BAD_PARAM_VALUE_3 (UEM_ERROR_OFFSET + 3)

Недопустимое значение в параметре 3.

#define UEM_ERROR_BAD_PARAM_VALUE_4 (UEM_ERROR_OFFSET + 4)

Недопустимое значение в параметре 4.

#define UEM_ERROR_BAD_PARAM_VALUE_5 (UEM_ERROR_OFFSET + 5)

Недопустимое значение в параметре 5.

#define UEM_ERROR_BAD_PARAM_VALUE_6 (UEM_ERROR_OFFSET + 6)

Недопустимое значение в параметре 6.

#define UEM_ERROR_BAD_PARAM_VALUE_7 (UEM_ERROR_OFFSET + 7)

Недопустимое значение в параметре 7.

#define UEM_ERROR_BAD_PARAM_VALUE_8 (UEM_ERROR_OFFSET + 8)

Недопустимое значение в параметре 8.

#define UEM_ERROR_BAD_PARAM_VALUE_9 (UEM_ERROR_OFFSET + 9)

Недопустимое значение в параметре 9.

#define UEM_ERROR_BAD_PARAM_VALUE_10 (UEM_ERROR_OFFSET + 10)

Недопустимое значение в параметре 10.

#define UEM_ERROR_INV_HANDLE (UEM_ERROR_OFFSET + 11)

Недействительный дескриптор.

#define UEM_ERROR_INV_HANDLE_TYPE (UEM_ERROR_OFFSET + 12)

Неподходящий тип дескриптора.

#define UEM_ERROR_NO_FREE_RAM (UEM_ERROR_OFFSET + 13)

Недостаточно ОЗУ УЭМ.

#define UEM_ERROR_NO_HOST_MEM (UEM_ERROR_OFFSET + 14)

Недостаточно ОЗУ управляющей ПЭВМ.

#define UEM_ERROR_NOT_CONNECTED (UEM_ERROR_OFFSET + 15)

Нет связи с устройством.

#define UEM_ERROR_INPOOL (UEM_ERROR_OFFSET + 16)

Внутренняя ошибка менеджера памяти.

#define UEM_ERROR_BCP_NINST (UEM_ERROR_OFFSET + 17)

Не установлена программа КШ.

#define UEM_ERROR_FORMAT_DISABLED (UEM_ERROR_OFFSET + 18)

Формат сообщения запрещен конфигурацией УЭМ.

#define UEM_ERROR_FORMAT_X_MCODE (UEM_ERROR_OFFSET + 19)

Формат сообщения несовместим с командой управления.

#define UEM_ERROR_NOT_APPLICABLE (UEM_ERROR_OFFSET + 20)

Действие не применимо к объекту.

В связи с особенностями данного объекта.

#define UEM_ERROR_ADDRESS_OUT_OF_RANGE (UEM_ERROR_OFFSET + 21)

Адрес вне допустимого диапазона.

#define UEM_ERROR_NUMBER_OUT_OF_RANGE (UEM_ERROR_OFFSET + 23)

Номер вне допустимого диапазона.

#define UEM_ERROR_BAD_TIMEOUT (UEM_ERROR_OFFSET + 24)

Недопустимое значение таймаута.

#define UEM_ERROR_BAD_OVERLAY_SOURCE (UEM_ERROR_OFFSET + 25)

Недопустимые исходные сегменты для наложения.

Сегменты не являются обычными командными сегментами или привязаны к одной и той же шине.

#define UEM_ERROR_WRONG_LOCATION (UEM_ERROR_OFFSET + 26)

Объект расположен не в том устройстве.

#define UEM_ERROR_TOO_MANY_DATAWORDS (UEM_ERROR_OFFSET + 27)

Слишком много слов данных.

#define UEM_ERROR_MAX_SIZE_EXCEED (UEM_ERROR_OFFSET + 28)

Превышен максимальный размер.

#define UEM_ERROR_NO_FRAME_APPEND (UEM_ERROR_OFFSET + 29)

Не добавлен кадр.

#define UEM_ERROR_IN_USE (UEM_ERROR_OFFSET + 30)

Устройство или объект используются.

#define UEM_ERROR_THREAD_FAULT (UEM_ERROR_OFFSET + 31)

Ошибки в работе служебной нити.

#define UEM_ERROR_BM_INTERNAL_BUFFER_OVERFLOW (UEM_ERROR_OFFSET + 32)

Переполнение внутреннего буфера МШ.

#define UEM_ERROR_INC_RESP (UEM_ERROR_OFFSET + 33)

Несовместимый ответный сегмент.

#define UEM_WARN_NO_NEXT_MESSAGE (UEM_WARN_OFFSET + 0)

Нет следующего сообщения (в буфере МШ).

#define UEM_WARN_JUST_IN_STATE (UEM_WARN_OFFSET + 1)

Устройство уже в нужном состоянии; никаких действий не производится.

Порядок действий при установлении связи с устройством

Описание типового сценария.

Для установления связи с устройством необходимо:

1. Открыть сеанс связи с носителем мезонинов при помощи функции **unmbase_init()** [4].
2. Открыть сеанс связи с устройством УЭМ при помощи функции **uem_init()**.
3. Связать эти сеансы при помощи функции **uem_connect()**.

Например:

```
unmbase_init ("VXI0::1::INSTR", VI_TRUE, VI_TRUE, &vi);  
uem_init ("VXI0::1::INSTR", VI_TRUE, VI_TRUE, &uem);  
uem_connect (uem, vi, num, VI_TRUE, VI_TRUE);
```

Адресная строка

Синтаксис идентификатора инструмента в VISA.

Адресная строка идентифицирует инструмент в системе VISA.

Для шины VXI строка имеет вид:

VXI [board]::VXI-la[::INSTR]

Где **board** - номер интерфейса VXI (по умолчанию 0), **VXI-la** - логический адрес VXI.

Необязательные параметры указаны в квадратных скобках.

Виртуальные устройства

Введение термина: виртуальное устройство.

В составе УЭМ имеется аппаратура, выполняющая функции контроллера шины (КШ), оконечных устройств (ОУ), монитора шины (МШ). При конфигурировании и управлении УЭМ удобно рассматривать эти компоненты аппаратуры УЭМ как отдельные устройства. В настоящей библиотеке и настоящем руководстве такие устройства называются *виртуальными устройствами* в составе УЭМ.

Для управления виртуальными устройствами их сначала надо открыть. Это выполняется при помощи функций **uem_bc_init()**, **uem_rt_init()**, **uem_bm_init()**. Открывать виртуальные устройства возможно только после того, как будет открыт УЭМ (см. **Порядок действий при установлении связи с устройством**).

Допускается открыть один виртуальный КШ, один виртуальный МШ, до 31 виртуальных ОУ (до 32 виртуальных ОУ при запрете групповых сообщений).

Командные и ответные сегменты

Введение терминов: сегмент, командный сегмент, ответный сегмент.

Под *сегментом* понимается последовательность соприкасающихся слов, то есть слов, передаваемых по МКПД без пауз.

Сегмент, передаваемый контроллером шины, содержит в начале командное слово (в форматах 3, 8 - два командных слова), за которым, в зависимости от формата сообщения МКПД, могут следовать слова данных. Сегмент, передаваемый контроллером шины, в настоящей библиотеке и настоящем документе называется *командным сегментом*.

Сегмент, передаваемый оконечным устройством, состоит из ответного слова, за которым, в зависимости от формата сообщения МКПД, могут следовать слова данных. Сегмент, передаваемый оконечным устройством, в настоящей библиотеке и настоящем документе называется *ответным сегментом*.

В терминах сегментов структура сообщений МКПД описывается следующим образом:

Сообщение состоит из командного сегмента, передаваемого контроллером шины, и ответных сегментов, передаваемых в ответ оконечными устройствами. В зависимости от формата сообщение может включать ноль, один или два ответных сегмента.

Более подробную информацию о форматах сообщений МКПД можно получить в [1].

Параметры передатчиков и характеристики выходных сигналов

Сведения о возможностях аппаратного обеспечения. Соответствие значений параметров конфигурации передатчиков и электрических характеристик выходных сигналов.

1. Приблизительное соответствие значений параметров UEM_TXA_RFT и UEM_TXB_RFT и длительностей фронта и среза сигнала МКПД.

2. Приблизительное соответствие значений параметров UEM_TXA_VPP и UEM_TXB_VPP и размаха сигнала МКПД.

Таблица 1. Приблизительное соответствие значений параметров UEM_TXA_RFT и UEM_TXB_RFT и длительностей фронта и среза сигнала МКПД

Разряды [15:6] доступны по записи и по чтению. Значение по умолчанию 0018h. Вводимое значение параметра имеет следующие особенности:

- два младших значения (0000h, 0001h) интерпретируются аппаратурой как 0002h;
- все нечетные значения из диапазона от 0003h до 0031h включительно интерпретируются аппаратурой как предшествующие четные значения (0003h как 0002h, 0005h как 0004h, и т.д.);
- максимальное значение составляет 0032h, все значения от 0033h и более интерпретируются аппаратурой как 0032h.

Значение параметра (hex)	Характеристика сигнала Длительность фронта (среза) импульсов сигнала на выходе устройства, нс (погрешность фактического значения не превышает 5 %)
0002	60
0004	65
0006	75
0008	85
000A	95
000C	110
000E	120
0010	135
0012	150
0014	165
0016	180
0018	195
001A	210
001C	225
001E	240
0020	255
0022	270
0024	285
285	300
0028	315
002A	330
002C	350

Значение параметра (hex)	Характеристика сигнала
	Длительность фронта (среза) импульсов сигнала на выходе устройства, нс (погрешность фактического значения не превышает 5 %)
002E	365
0030	370
0032	Отдельная настройка передатчика, при которой форма сигнала при длительности импульсов 500 нс аппроксимирует синусоидальный сигнал

Таблица 2. Приблизительное соответствие значений параметров UEM_TXA_VPP и UEM_TXB_VPP и размаха сигнала МКПД

Значение параметра (hex)	Характеристика сигнала
	(НП – непосредственное подключение, ТП – трансформаторное подключение по ГОСТ Р 52070) Размах сигнала на выходе устройства на эквивалентной нагрузке 35 Ом (НП) / 70 Ом (ТП), В:
0000 ... 000D	0 ... 0,28 (НП) / 0 ... 0,8 (ТП), при установке таких значений следует учитывать, что сигнал может находиться ниже границы порога срабатывания приемника, и информация может не быть принята МШ
000E / 000E	0,28 (НП) / 0,8 (ТП, соответствует 0,2 В в точке подключения ТМ по ГОСТ Р 51765)
003B / 003A	1,2 (НП) / 3,44 (ТП, соответствует 0,86 В в точке подключения ТМ по ГОСТ Р 51765)
0090 / 0090	3,0 (НП) / 8,4 (ТП, соответствует 2,1 В в точке подключения ТМ по ГОСТ Р 51765)
0136 / 0136	6,38 (НП) / 18,3 (ТП), значение по умолчанию
01B6 / 0195	9,0 (НП) / 24,0 (ТП, соответствует 6,0 В в точке подключения ТМ по ГОСТ Р 51765)
01FF	10,8 (НП) / 30,8 (ТП), максимальное значение

Типы вносимых ошибок кодирования

Сведения о возможностях аппаратного обеспечения в части внесения ошибок кодирования и способы описания (программирования) вносимых ошибок.

Перечисления

- enum UEM_ERROR_TYPE { UEM_ERRT_NONE = 0, UEM_ERRT_INV_PARITY = 1, UEM_ERRT_WRONG_BITCOUNT = 2, UEM_ERRT_BAD_SYNCHRO = 3, UEM_ERRT_BAD_BIPHASE_ZERO = 4, UEM_ERRT_BAD_BIPHASE_POS = 5, UEM_ERRT_BAD_BIPHASE_NEG = 6, UEM_ERRT_SHIFT_EDGE = 7 }

Тип вносимой ошибки кодирования.

- enum UEM_SYNCHRO_ERROR_POS { UEM_BAD_SYNCHRO_NONE = 0, UEM_BAD_SYNCHRO_IEEEEE = 1, UEM_BAD_SYNCHRO_EIEEEE = 2, UEM_BAD_SYNCHRO_EEIEEE = 3, UEM_BAD_SYNCHRO_NONE2 = 4, UEM_BAD_SYNCHRO_EEEIEE = 5, UEM_BAD_SYNCHRO_EEEEEIE = 6, UEM_BAD_SYNCHRO_EEEEEI = 7 }

Позиция ошибки кодирования синхримпульса.

Граничные значения изменения количества разрядов

Эти константы задают граничные значения аргумента **error_pos** в функциях **uem_cseg_error_set()**, **uem_response_error_set()**, когда в аргументе **error_type** указано **UEM_ERRT_WRONG_BITCOUNT**.

- #define UEM_BITCOUNT_CHANGE_MIN (-3)
Минимальное приращение количества разрядов.
- #define UEM_BITCOUNT_CHANGE_MAX (+3)
Максимальное приращение количества разрядов.

Граничные значения позиции ошибки

Эти константы задают граничные значения аргумента **error_pos** для некоторых значений аргумента **error_type** в функциях **uem_cseg_error_set()**, **uem_response_error_set()**.

- #define UEM_BIPHASE_POS_MIN 4
Минимальная позиция при error_type, равном UEM_ERRT_BAD_BIPHASE_ZERO, UEM_ERRT_BAD_BIPHASE_POS или UEM_ERRT_BAD_BIPHASE_NEG.
- #define UEM_BIPHASE_POS_MAX 20
Максимальная позиция при error_type, равном UEM_ERRT_BAD_BIPHASE_ZERO, UEM_ERRT_BAD_BIPHASE_POS или UEM_ERRT_BAD_BIPHASE_NEG.
- #define UEM_SHIFT_POS_MIN 0
Минимальная позиция при error_type, равном UEM_ERRT_SHIFT_EDGE.
- #define UEM_SHIFT_POS_MAX 40
Максимальная позиция при error_type, равном UEM_ERRT_SHIFT_EDGE.

Граничные значения величины сдвига

Эти константы задают граничные значения для аргумента **error_param** в функциях **uem_cseg_error_set()**, **uem_response_error_set()**, когда аргумент **error_type** равен **UEM_ERRT_SHIFT_EDGE**. Величина сдвига указывается в единицах по 10 нс. Отрицательные значения обозначают сдвиг влево, положительные - вправо.

- #define UEM_SHIFT_LENGTH_MIN (-25)
Максимальная величина сдвига влево.
- #define UEM_SHIFT_LENGTH_MAX (+25)
Максимальная величина сдвига вправо.

Значения аргументов по умолчанию

Эти константы задают значения аргументов функций `uem_cseg_error_set()`, `uem_response_error_set()` по умолчанию, соответствуют отсутствию внесения ошибок.

- `#define UEM_ERROR_TYPE_DEFAULT 0`
Значение по умолчанию для `error_type`.
 - `#define UEM_ERROR_POS_DEFAULT 0`
Значение по умолчанию для `error_pos`.
 - `#define UEM_ERROR_PARAM_DEFAULT 0`
Значение по умолчанию для `error_param`.
-

Подробное описание

Сведения о возможностях аппаратного обеспечения в части внесения ошибок кодирования и способы описания (программирования) вносимых ошибок.

Программирование ошибок кодирования выполняется при помощи функций `uem_cseg_error_set()`, `uem_response_error_set()`, имеющих для описания ошибок следующие параметры: `error_type` - основной параметр - тип вносимой ошибки, `error_pos` и `error_param` - дополнительные параметры, значения которых интерпретируются в зависимости от типа ошибки. Параметр `error_type` должен принимать значения из перечисления `UEM_ERROR_TYPE`. Описания типов ошибок и интерпретации дополнительных параметров содержится в описании этого перечисления. См. `UEM_ERROR_TYPE`.

В данном разделе также определены константы для задания параметров `error_pos` и `error_param` при различных `error_type`.

Макросы

`#define UEM_BITCOUNT_CHANGE_MIN (-3)`

Минимальное приращение количества разрядов.

`#define UEM_BITCOUNT_CHANGE_MAX (+3)`

Максимальное приращение количества разрядов.

`#define UEM_BIPHASE_POS_MIN 4`

Минимальная позиция при `error_type`, равном `UEM_ERRT_BAD_BIPHASE_ZERO`, `UEM_ERRT_BAD_BIPHASE_POS` или `UEM_ERRT_BAD_BIPHASE_NEG`.

`#define UEM_BIPHASE_POS_MAX 20`

Максимальная позиция при `error_type`, равном `UEM_ERRT_BAD_BIPHASE_ZERO`, `UEM_ERRT_BAD_BIPHASE_POS` или `UEM_ERRT_BAD_BIPHASE_NEG`.

`#define UEM_SHIFT_POS_MIN 0`

Минимальная позиция при `error_type`, равном `UEM_ERRT_SHIFT_EDGE`.

#define UEM_SHIFT_POS_MAX 40

Максимальная позиция при **error_type**, равном **UEM_ERRT_SHIFT_EDGE**.

#define UEM_SHIFT_LENGTH_MIN (-25)

Максимальная величина сдвига влево.

#define UEM_SHIFT_LENGTH_MAX (+25)

Максимальная величина сдвига вправо.

#define UEM_ERROR_TYPE_DEFAULT 0

Значение по умолчанию для **error_type**.

#define UEM_ERROR_POS_DEFAULT 0

Значение по умолчанию для **error_pos**.

#define UEM_ERROR_PARAM_DEFAULT 0

Значение по умолчанию для **error_param**.

Перечисления

enum UEM_ERROR_TYPE

Тип вносимой ошибки кодирования.

Элементы перечислений:

UEM_ERRT_NONE Ошибка не вносится.

UEM_ERRT_INV_PARITY Инверсия разряда четности (20-го) по отношению к его достоверному значению.

UEM_ERRT_WRONG_BITCOUNT Ошибка количества разрядов в слове.

Параметр **error_pos** интерпретируется как изменение числа разрядов. Допустимые значения: от -3 (**UEM_BITCOUNT_CHANGE_MIN**) до +3 (**UEM_BITCOUNT_CHANGE_MAX**).

При уменьшении количества разрядов последовательно исключаются 19-й, 18-й, 17-й разряды 20-ти разрядного слова по ГОСТ Р 52070, при увеличении – «избыточные» разряды добавляются после 19-го разряда, «сдвигая» разряд четности на последнюю позицию, при этом бит четности всегда подсчитывается с учетом заданных значений «избыточных» разрядов, либо с учетом только «оставшихся» разрядов при «укорочении», по правилу дополнения до нечетности.

Параметр **error_param** при **error_pos** > 0 содержит значения для дополнительных разрядов слова, бит 0 задает 20-й разряд слова, бит 1 - 21-й (если он есть), бит 2 - 22-й (если он есть).

UEM_ERRT_BAD_SYNCHRO Ошибка кодирования синхроимпульса слова.

Параметр **error_pos** должен содержать код позиции ошибки в синхроимпульсе, один из элементов перечисления **UEM_SYNCHRO_ERROR_POS**.

UEM_ERRT_BAD_BIPHASE_ZERO Пропуск перехода через 0, удерживается нулевой уровень.

Ошибка бифазного кодирования типа «проскок» в выбранном разряде слова, т.е. установка выходного сигнала МКПД в нулевой уровень напряжения в течение интервала передачи выбранного разряда.

Ошибка вносится в разряд с номером, указанным в параметре **error_pos**. Допустимый диапазон значений: от 4 (**UEM_BIPHASE_POS_MIN**) до 20 (**UEM_BIPHASE_POS_MAX**).

UEM_ERRT_BAD_BIPHASE_POS Пропуск перехода через 0, удерживается положительный уровень.

Ошибка бифазного кодирования, т.е. отсутствие обязательного перехода через 0 выходного сигнала МКПД в середине выбранного разряда, с установкой положительного уровня напряжения в течение интервала передачи выбранного разряда.

Ошибка вносится в разряд с номером, указанным в параметре **error_pos**. Допустимый диапазон значений: от 4 (**UEM_BIPHASE_POS_MIN**) до 20 (**UEM_BIPHASE_POS_MAX**).

UEM_ERRT_BAD_BIPHASE_NEG Пропуск перехода через 0, удерживается отрицательный уровень.

Ошибка бифазного кодирования, т.е. отсутствие обязательного перехода через 0 выходного сигнала МКПД в середине выбранного разряда, с установкой отрицательного уровня напряжения в течение интервала передачи выбранного разряда.

Ошибка вносится в разряд с номером, указанным в параметре **error_pos**. Допустимый диапазон значений: от 4 (**UEM_BIPHASE_POS_MIN**) до 20 (**UEM_BIPHASE_POS_MAX**).

UEM_ERRT_SHIFT_EDGE Сдвиг перехода через 0.

Ошибка смещения момента пересечения нулевого уровня напряжения для выбранного перехода через 0 на заданную величину.

Параметр **error_pos** задает позицию сдвигаемого перехода - номер полуразряда слова, допустимые значения от 0 (**UEM_SHIFT_POS_MIN**) до 40 (**UEM_SHIFT_POS_MAX**).

Параметр **error_param** задает направление («влево» или «вправо») и величину временного сдвига с разрешением 10 нс на МЗР в диапазоне от -25 (**UEM_SHIFT_LENGTH_MIN**) до +25 (**UEM_SHIFT_LENGTH_MAX**), т.е. от -250 до +250 нс.

Технический прием внесения данной ошибки заключается в инверсии (одновременной смене полярности сигналов управления передатчиком МКПД относительно «номинальных» логических уровней) выходного сигнала на следующих интервалах:

– при сдвиге «влево»: от точки ($((\mathbf{error_pos} - 1) \times 500) + (500 - (\mathbf{error_param} \times 10))$) нс до точки ($\mathbf{error_pos} \times 500$) нс;

– при сдвиге «вправо»: от точки ($\mathbf{error_pos} \times 500$) нс до точки ($(\mathbf{error_pos} \times 500) + (\mathbf{error_param} \times 10)$) нс.

Инверсия производится по описанным правилам вне зависимости от того, присутствует ли в точке, заданной значением параметра **error_pos**, пересечение нуля; если в «номинальном» коде пересечения нуля в данной точке нет, то при внесении ошибки будет сформирован «лишний» переход через ноль.

enum UEM_SYNCHRO_ERROR_POS

Позиция ошибки кодирования синхроимпульса.

Эти значения предназначены для использования в аргументе **error_pos** в функциях **uem_cseg_error_set()**, **uem_response_error_set()**, когда в аргументе **error_type** указано **UEM_ERRT_BAD_SYNCHRO**.

Под 1/6 частями синхроимпульса подразумеваются 6 последовательных интервалов по 500 нс, образующих этот синхроимпульс. Инверсия подразумевает одновременную смену полярности сигналов управления передатчиком МКПД на заданном интервале относительно «номинальных» уровней данных сигналов на этом интервале.

Элементы перечислений:

- UEM_BAD_SYNCHRO_NONE* Ошибка не вносится.
- UEM_BAD_SYNCHRO_IEEEEE* Инверсия первой 1/6.
- UEM_BAD_SYNCHRO_EIEEEE* Инверсия второй 1/6.
- UEM_BAD_SYNCHRO_EEIEEE* Инверсия третьей 1/6.
- UEM_BAD_SYNCHRO_NONE2* Ошибка не вносится.
- UEM_BAD_SYNCHRO_EEEIEE* Инверсия четвертой 1/6.
- UEM_BAD_SYNCHRO_EEEEIE* Инверсия пятой 1/6.
- UEM_BAD_SYNCHRO_EEEEEI* Инверсия шестой 1/6.

Управление синхронизацией

Сведения об аппаратных средствах синхронизации с внешним оборудованием и об управлении ими. В УЭМ существует четыре сигнала внешней синхронизации [2, 3]:

sync_in_1

Входной сигнал внешней синхронизации 1.

sync_in_2

Входной сигнал внешней синхронизации 2.

sync_out_1

Выходной сигнал внешней синхронизации 1.

sync_out_2

Выходной сигнал внешней синхронизации 2.

Сигналы **sync_in_1** и **sync_in_2** используются для синхронизации КШ с внешним оборудованием.

Сигнал **sync_in_1** служит для выдачи контроллером шины сообщений или слов в линию синхронно с внешним источником синхронизации. Такой синхронизации подвергаются сообщения или слова, для которых при программировании паузы функциями **uem_cseg_gap_get()**, **uem_cseg_word_gap_set()** в аргументе **flags** указан флаг **UEM_CSEG_GAP_ESYNC**. При функционировании КШ сигнал может использоваться как для задания периода всех сообщений (если привязка установлена в каждом КС), так и для задания периода блоков сообщений (кадров) (если привязка установлена только в первом КС блока).

Сигнал **sync_in_2** служит для отложенного или периодического запуска КШ, при запуске КШ функцией **uem_bc_start()** с аргументом **flags = UEM_BC_START_WAITING**.

Сигналы **sync_out_1** и **sync_out_2** формируются МШ и сообщают внешнему оборудованию о событиях в шине, детектируемых при помощи ряда условий.

Сигнал **sync_out_1** формируется после того, как обнаружена передача по шине МКПД слова, для которого выполняются каждое из следующих условий:

- Слово передано по шине, определяемой параметром конфигурации **UEM_SYNC1_CH**.
- Перед словом была пауза в передаче или не было такой паузы - в соответствии с заданным значением параметра **UEM_SYNC1_GAPB**.
- Передача слова выполнена с ошибкой или без ошибки в соответствии с заданным значением параметра **UEM_SYNC1_ERR**.

- Синхроимпульс слова соответствует заданному значению параметра конфигурации **UEM_SYNC1_C_D_**.
- Содержание (информационных разрядов) слова идентично значению параметра **UEM_BM_WORD_PATTERN** (при этом в сравнении участвуют только разряды, заданные параметром **UEM_BM_WORD_MASK**).

Сигнал **sync_out_2** формируется после обнаружения в шине достоверного командного или ответного слова с адресом ОУ, определяемым параметром конфигурации **UEM_SYNC2_VRTA**.

Обработка и формирование сигналов синхронизации разрешается параметрами конфигурации **UEM_SYNC_IN_1_ENA**, **UEM_SYNC_IN_2_ENA**, **UEM_SYNC_OUT_1_ENA**, **UEM_SYNC_OUT_2_ENA** соответственно. По умолчанию обработка и формирование запрещены.

Однократная программная имитация поступления или формирования сигналов выполняется путем записи значения 1 в параметры конфигурации **UEM_SYNC_IN_1_SET**, **UEM_SYNC_IN_2_SET**, **UEM_SYNC_OUT_1_SET**, **UEM_SYNC_OUT_2_SET** соответственно.

Для сигналов **sync_in_1** и **sync_in_2** имеются механизмы внутренней аппаратной генерации с заданной периодичностью. Разрешение генерации управляется параметрами **UEM_SYNC_IN_1_INTGEN** и **UEM_SYNC_IN_2_INTGEN** соответственно, а период повтора определяется параметрами **UEM_IST1** и **UEM_IST2** соответственно.

Определения примитивных типов

Дескрипторы устройства и объекта, идентификатор параметра, логическое, 32- и 16-битные значения.

Определения типов

- **typedef ViSession UEM_DEVHANDLE**
Дескриптор устройства УЭМ или виртуального устройства в составе УЭМ.
 - **typedef ViSession UEM_OBJHANDLE**
Дескриптор объекта в ОЗУ УЭМ.
 - **typedef ViBoolean UEM_BOOL**
Логическое значение.
 - **typedef ViUInt16 UEM_PARAMID**
Идентификатор параметра.
 - **typedef ViUInt32 UEM_DWORD**
32-битное целое без знака.
 - **typedef ViUInt16 UEM_WORD**
16-битное целое без знака.
-

Подробное описание

Дескрипторы устройства и объекта, идентификатор параметра, логическое, 32- и 16-битные значения.

Типы

typedef ViSession UEM_DEVHANDLE

Дескриптор устройства УЭМ или виртуального устройства в составе УЭМ.

typedef ViSession UEM_OBJHANDLE

Дескриптор объекта в ОЗУ УЭМ.

typedef ViBoolean UEM_BOOL

Логическое значение.

typedef ViUInt16 UEM_PARAMID

Идентификатор параметра.

Возможные значения представлены в разделе **Описание параметров**.

typedef ViUInt32 UEM_DWORD

32-битное целое без знака.

typedef ViUInt16 UEM_WORD

16-битное целое без знака.

Установление и разрыв связи с устройством

Описания функций установления и разрыва связи с устройством.

Функции

- ViStatus **uem_init** (ViRsrc idstr, ViBoolean idn, ViBoolean reset, ViSession *uem)
Инициализация объекта УЭМ.
- ViStatus **uem_connect** (ViSession uem, ViSession vi, ViUInt16 mezznum, ViBoolean idn, ViBoolean reset)
Привязка объекта УЭМ к сеансу носителя мезонина.
- ViStatus **uem_bc_init** (UEM_DEVHANDLE *bc, UEM_DEVHANDLE uem)
Открытие виртуального КШ в составе УЭМ.
- ViStatus **uem_rt_init** (UEM_DEVHANDLE *rt, UEM_DEVHANDLE uem, UEM_WORD rtaddr)
Открытие виртуального ОУ в составе УЭМ.
- ViStatus **uem_bm_init** (UEM_DEVHANDLE *bm, UEM_DEVHANDLE uem)
Открытие виртуального МШ в составе УЭМ.
- ViStatus **uem_close** (UEM_DEVHANDLE anydev)
Закрытие УЭМ или любого виртуального устройства в составе УЭМ.

Подробное описание

Описания функций установления и разрыва связи с устройством.

Приведенные здесь функции устанавливают связь с устройством УЭМ и с виртуальными устройствами (КШ/ОУ/МШ) в составе УЭМ. Рекомендуется также ознакомиться с разделами **Порядок действий при установлении связи с устройством** и **Виртуальные устройства**.

Функции

ViStatus **uem_init** (ViRsrc *idstr*, ViBoolean *idn*, ViBoolean *reset*, ViSession * *uem*)

Инициализация объекта УЭМ.

Рекомендуется ознакомиться с подразделом **Порядок действий при установлении связи с устройством**.

Аргументы:

in	<i>idstr</i>	В данном параметре передается Адресная строка VISA . Должна идентифицировать носитель мезонинов.
in	<i>idn</i>	Данный параметр определяет, производить ли идентификацию устройства в процедуре инициализации. Допустимые значения: VI_OFF (0) - Не производить идентификацию. VI_ON (1) - Производить идентификацию (по умолчанию). Примечание: Обычно не следует отключать идентификацию устройства, так как это дает дополнительную проверку на соответствие типа устройства тому, на который рассчитан драйвер.

in	<i>reset</i>	Данный параметр определяет, производить ли сброс устройства в процедуре инициализации. Допустимые значения: VI_OFF (0) - Не производить сброс. VI_ON (1) - Производить сброс (по умолчанию).
out	<i>uem</i>	В данной переменной функция возвращает дескриптор устройства УЭМ, который необходимо сохранить для всех последующих вызовов функций драйвера устройства. Примечания: 1) При каждом новом вызове функции инициализации открывается еще один сеанс связи с тем же устройством. 2) Обратите внимание, что типы UEM_DEVHANDLE и ViSession являются синонимами.

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_connect (ViSession *uem*, ViSession *vi*, ViUInt16 *meznum*, ViBoolean *idn*, ViBoolean *reset*)

Привязка объекта УЭМ к сеансу носителя мезонины.

Рекомендуется ознакомиться с подразделом **Порядок действий при установлении связи с устройством.**

Аргументы:

in	<i>uem</i>	Дескриптор устройства, возвращенный функцией uem_init() .
in	<i>vi</i>	Номер сеанса носителя мезонинов, возвращенный функцией unmbase_init() [4].
in	<i>meznum</i>	В данном параметре указывается номера позиции инструмента устройства УЭМ в носителе мезонинов. Допустимые значения 1-4.
in	<i>idn</i>	Данный параметр определяет, производить ли идентификацию устройства. Допустимые значения: VI_OFF (0) - Не производить идентификацию. VI_ON (1) - Производить идентификацию (по умолчанию). Примечание: Обычно не следует отключать идентификацию устройства, так как это дает дополнительную проверку на соответствие типа устройства тому, на который рассчитан драйвер.
in	<i>reset</i>	Данный параметр определяет, производить ли сброс устройства. Допустимые значения: VI_OFF (0) - Не производить сброс. VI_ON (1) - Производить сброс (по умолчанию).

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_bc_init (UEM_DEVHANDLE * *bc*, UEM_DEVHANDLE *uem*)

Открытие виртуального КШ в составе УЭМ.

Аргументы:

out	<i>bc</i>	Дескриптор КШ.
in	<i>uem</i>	Дескриптор УЭМ.

Возвращает:

Код завершения. См. **Коды завершения**.
При повторном открытии КШ будет возвращен дескриптор уже открытого КШ.

ViStatus uem_rt_init (UEM_DEVHANDLE * *rt*, UEM_DEVHANDLE *uem*, UEM_WORD *rtaddr*)

Открытие виртуального ОУ в составе УЭМ.

Аргументы:

out	<i>rt</i>	Дескриптор виртуального ОУ.
in	<i>uem</i>	Дескриптор УЭМ.
in	<i>rtaddr</i>	Адрес ОУ. Допустимые значения 0-30 (0-31 в сетях с запрещенными групповыми командами).

Возвращает:

Код завершения. См. **Коды завершения**.
При повторном открытии ОУ с указанным номером будет возвращен дескриптор уже открытого ОУ с этим номером.

ViStatus uem_bm_init (UEM_DEVHANDLE * *bm*, UEM_DEVHANDLE *uem*)

Открытие виртуального МШ в составе УЭМ.

Аргументы:

out	<i>bm</i>	Дескриптор МШ.
in	<i>uem</i>	Дескриптор УЭМ.

Возвращает:

Код завершения. См. **Коды завершения**.
При повторном открытии МШ будет возвращен дескриптор уже открытого МШ.

ViStatus uem_close (UEM_DEVHANDLE *anydev*)

Закрытие УЭМ или любого виртуального устройства в составе УЭМ.

При закрытии виртуального устройства автоматически выполняется действие **uem_stop()** для этого устройства, если необходимо, сброс виртуального устройства (**uem_reset()**) и освобождение всех объектов в ОЗУ виртуального устройства. Все дескрипторы данного виртуального устройства после этого считаются недействительными. Остальные виртуальные устройства не затрагиваются.

При закрытии УЭМ в целом выполняется закрытие всех виртуальных устройств, после чего разрывается связь с устройством УЭМ. Дескриптор УЭМ после этого считается недействительным.

Аргументы:

in	<i>anydev</i>	Дескриптор УЭМ или виртуального устройства (КШ, ОУ, МШ) в составе УЭМ.
----	---------------	--

Возвращает:

Код завершения. См. **Коды завершения**.

Действия с дескрипторами

Описания функций исследования дескрипторов объектов библиотеки и навигации между ними.

Перечисления

- enum **UEM_HANDLE_TYPE** { **UEM_INVH**, **UEM_UEM**, **UEM_BC**, **UEM_RT**, **UEM_BM**, **UEM_CSEG**, **UEM_BCP**, **UEM_RESP** }
Тип дескриптора объекта библиотеки УЭМ.

Функции

- **UEM_HANDLE_TYPE uem_handle_type** (**UEM_DEVHANDLE** anyobject)
Тип дескриптора объекта библиотеки УЭМ.
- **ViStatus uem_parent_dev** (**UEM_DEVHANDLE** anyobject, **UEM_DEVHANDLE *parentdev**)
Родительское устройство.
- **ViStatus uem_root_dev** (**UEM_DEVHANDLE** anyobject, **UEM_DEVHANDLE *uem**)
Физическое устройство УЭМ.
- **ViStatus uem_layer_handle** (**UEM_DEVHANDLE** uem, **ViUInt32 sel**, **ViSession *handle**)
Связь с ПО нижележащих слоев.

Селектор сеанса

- **#define UEM_SEL_UNMUEM 1**
Сеанс низкоуровневого драйвера УЭМ unmuem.
- **#define UEM_SEL_UNMBASE 2**
Сеанс драйвера носителя мезонинов unmbase.
- **#define UEM_SEL_UNBASE_INT 3**
"Внутренний" сеанс драйвера носителя мезонинов unmbase.

Подробное описание

Описания функций исследования дескрипторов объектов библиотеки и навигации между ними.

Приведенные здесь функции позволяют определить тип дескриптора и переходить по иерархии дескрипторов.

Макросы

#define UEM_SEL_UNMUEM 1

Сеанс низкоуровневого драйвера УЭМ unmuem.

#define UEM_SEL_UNMBASE 2

Сеанс драйвера носителя мезонинов unmbase.

#define UEM_SEL_UNBASE_INT 3

"Внутренний" сеанс драйвера носителя мезонинов unmbase.

Перечисления

enum UEM_HANDLE_TYPE

Тип дескриптора объекта библиотеки УЭМ.

Элементы перечислений:

UEM_INVH Недействительный дескриптор.

UEM_UEM Физическое устройство УЭМ.

UEM_BC Виртуальный КШ.

UEM_RT Виртуальное ОУ.

UEM_BM Виртуальный МШ.

UEM_CSEG Командный сегмент.

UEM_BCP Программа КШ.

UEM_RESP Ответный сегмент.

Функции

UEM_HANDLE_TYPE uem_handle_type (UEM_DEVHANDLE *anyobject*)

Тип дескриптора объекта библиотеки УЭМ.

Возвращаемое значение позволяет определить тип объекта, дескриптор которого предъявлен.

Аргументы:

in	<i>anyobject</i>	Дескриптор устройства или объекта в ОЗУ устройства.
----	------------------	---

Возвращает:

Тип дескриптора.

ViStatus uem_parent_dev (UEM_DEVHANDLE *anyobject*, UEM_DEVHANDLE * *parentdev*)

Родительское устройство.

Функция возвращает:

для объекта в ОЗУ УЭМ (допустимо указывать в аргументе *anyobject*) - дескриптор виртуального устройства, которому принадлежит данный объект,
для виртуального устройства - дескриптор УЭМ,
для УЭМ - дескриптор самого УЭМ.

Аргументы:

in	<i>anyobject</i>	Дескриптор устройства или объекта в ОЗУ, родительское устройство которого требуется выяснить.
out	<i>parentdev</i>	В этом аргументе передается адрес переменной, в которую будет записан искомый дескриптор.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_root_dev (UEM_DEVHANDLE *anyobject*, UEM_DEVHANDLE * *uem*)

Физическое устройство УЭМ.

Для любого объекта в ОЗУ или виртуального устройства функция возвращает дескриптор УЭМ, которому принадлежит этот объект или виртуальное устройство.

Аргументы:

in	<i>anyobject</i>	Дескриптор устройства или объекта в ОЗУ, по которому требуется определить дескриптор УЭМ.
out	<i>uem</i>	В этом аргументе передается адрес переменной, в которую будет записан искомый дескриптор УЭМ.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_layer_handle (UEM_DEVHANDLE *uem*, ViUInt32 *sel*, ViSession * *handle*)

Связь с ПО нижележащих слоев.

Настоящая расширенная библиотека полнофункциональна и позволяет решать все задачи по управлению устройствами УЭМ и их использованию без обращения к другим компонентам программного обеспечения.

(Естественным исключением является процедура инициализации, в ходе которой приложение обращается к функциям менеджера ресурсов VISA и к функции `unmbase_init()` драйвера носителя мезонина [4].)

Тем не менее библиотека не препятствует работе с нижележащими слоями программного обеспечения. Расширенная библиотека `uem` работает поверх драйвера мезонина `unmuem` [5, 6], который, в свою очередь, работает поверх драйвера носителя мезонинов `unmbase` [4].

Данная функция позволяет получить дескрипторы сеансов работы этих компонентов программного обеспечения и работать с ними напрямую.

Аргументы:

in	<i>uem</i>	Дескриптор УЭМ.
in	<i>sel</i>	Данный аргумент определяет выбор нижележащего ПО и его дескриптора, см. константы UEM_SEL_XXXXX .
out	<i>handle</i>	В этой переменной возвращается запрошенный дескриптор сеанса.

Возвращает:

Код завершения. См. **Коды завершения**.

Параметры конфигурации УЭМ

Описания параметров конфигурации и функций для работы с ними.

Группы

- **Описание параметров**
Описания параметров и определения констант - идентификаторов параметров конфигурации.
- **Значения параметров**
Определения констант - значений параметров конфигурации.

Функции

- ViStatus **uem_param_get** (UEM_DEVHANDLE uem, UEM_PARAMID paramid, UEM_DWORD *value)
Считывание значения конфигурационного параметра.
- ViStatus **uem_param_set** (UEM_DEVHANDLE uem, UEM_PARAMID paramid, UEM_DWORD value)
Запись значения конфигурационного параметра.

Подробное описание

Описания параметров конфигурации и функций для работы с ними.

Под параметром конфигурации понимается отдельный бит или связанная по смыслу и расположению группа битов в регистрах УЭМ. Параметры конфигурации относятся к УЭМ в целом, а не к виртуальным устройствам в составе УЭМ. Каждому параметру конфигурации присвоен фиксированный числовой идентификатор, который передается в функции считывания и записи параметров.

Функции

ViStatus uem_param_get (UEM_DEVHANDLE *uem*, UEM_PARAMID *paramid*, UEM_DWORD **value*)

Считывание значения конфигурационного параметра.

Аргументы:

in	<i>uem</i>	Дескриптор УЭМ. Допускается указывать вместо дескриптора УЭМ дескриптор любого виртуального устройства в составе этого УЭМ. Функция выполняет такой вызов, как если бы был указан дескриптор УЭМ.
in	<i>paramid</i>	Идентификатор параметра. См. Описание параметров .
out	<i>value</i>	Значение параметра. См. Значения параметров .

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_param_set (UEM_DEVHANDLE *uem*, UEM_PARAMID *paramid*, UEM_DWORD *value*)

Запись значения конфигурационного параметра.

Аргументы:

in	<i>uem</i>	Дескриптор УЭМ. Допускается указывать вместо дескриптора УЭМ дескриптор любого виртуального устройства в составе этого УЭМ. Функция выполняет такой вызов, как если бы был указан дескриптор УЭМ.
in	<i>paramid</i>	Идентификатор параметра. См. Описание параметров.
in	<i>value</i>	Значение параметра. См. Значения параметров.

Возвращает:

Код завершения. См. **Коды завершения.**

Описание параметров

Описания параметров и определения констант - идентификаторов параметров конфигурации.

Макросы

- #define **UEM_TMTA_DIS** UEMi_MAKE_PARAMID (0, 2, 2)
Запрет работы передатчика шины А.
- #define **UEM_TMTB_DIS** UEMi_MAKE_PARAMID (0, 3, 3)
Запрет работы передатчика шины Б.
- #define **UEM_RCVA_DIS** UEMi_MAKE_PARAMID (0, 4, 4)
Запрет работы приемника шины А.
- #define **UEM_RCVB_DIS** UEMi_MAKE_PARAMID (0, 5, 5)
Запрет работы приемника шины Б.
- #define **UEM_SYNC_IN_1_ENA** UEMi_MAKE_PARAMID (0, 8, 8)
Разрешение входной синхронизации 1.
- #define **UEM_SYNC_IN_2_ENA** UEMi_MAKE_PARAMID (0, 9, 9)
Разрешение входной синхронизации 2.
- #define **UEM_SYNC_OUT_1_ENA** UEMi_MAKE_PARAMID (0, 10, 10)
Разрешение выходной синхронизации 1.
- #define **UEM_SYNC_OUT_2_ENA** UEMi_MAKE_PARAMID (0, 11, 11)
Разрешение выходной синхронизации 2.
- #define **UEM_BRCST_DIS** UEMi_MAKE_PARAMID(0, 12, 12)
Запрет групповых сообщений.
- #define **UEM_SYNC_IN_1_INTGEN** UEMi_MAKE_PARAMID(0, 13, 13)
Разрешение внутренней эмуляции сигнала входной синхронизации 1.
- #define **UEM_ERR_INJ_DIS** UEMi_MAKE_PARAMID (0, 14, 14)
Запрет внесения ошибок кодирования в передаваемую в МКПД информацию для КШ и ОУ.
- #define **UEM_SYNC_IN_2_INTGEN** UEMi_MAKE_PARAMID(0, 29, 29)
Разрешение внутренней эмуляции сигнала входной синхронизации 2.
- #define **UEM_TMT_RES** UEMi_MAKE_PARAMID (5, 4, 4)
Сброс настроек передатчиков (только запись).

- #define **UEM_SYNC_IN_1_SET** UEMi_MAKE_PARAMID (5, 10, 10)
Программная генерация сигнала входной синхронизации 1 (только запись).
- #define **UEM_SYNC_IN_2_SET** UEMi_MAKE_PARAMID (5, 11, 11)
Программная генерация сигнала входной синхронизации 2 (только запись).
- #define **UEM_SYNC_OUT_1_SET** UEMi_MAKE_PARAMID (5, 12, 12)
Программная генерация сигнала выходной синхронизации 1 (только запись).
- #define **UEM_SYNC_OUT_2_SET** UEMi_MAKE_PARAMID (5, 13, 13)
Программная генерация сигнала выходной синхронизации 2 (только запись).
- #define **UEM_DB_ACT** UEMi_MAKE_PARAMID (5, 31, 31)
Обнаружена передача данных по шине (только чтение).
- #define **UEM_TXA_RFT** UEMi_MAKE_PARAMID (6, 31, 16)
Управление длительностью фронта и среза при передаче в шину А.
- #define **UEM_TXA_VPP** UEMi_MAKE_PARAMID (6, 15, 0)
Управление размахом сигнала при передаче в шину А.
- #define **UEM_TXB_RFT** UEMi_MAKE_PARAMID (7, 31, 16)
Управление длительностью фронта и среза при передаче в шину Б.
- #define **UEM_TXB_VPP** UEMi_MAKE_PARAMID (7, 15, 0)
Управление размахом сигнала при передаче в шину Б.
- #define **UEM_MC_DIS** UEMi_MAKE_PARAMID (0x0C, 0, 0)
Запрет команд управления.
- #define **UEM_BTMT_DIS** UEMi_MAKE_PARAMID (0x0C, 1, 1)
Запрет блокирования и разблокирования передатчиков ОУ.
- #define **UEM_BRTF_DIS** UEMi_MAKE_PARAMID (0x0C, 2, 2)
Запрет блокирования и разблокирования признака неисправности ОУ.
- #define **UEM_SYNC2_VRTA** UEMi_MAKE_PARAMID (0x0C, 7, 3)
Номер ОУ – условие выработки сигнала выходной синхронизации 2.
- #define **UEM_SYNC1_C_D** UEMi_MAKE_PARAMID (0x0C, 8, 8)
Командное/ответное (1) слово или слово данных (0) – условие выработки сигнала выходной синхронизации 1.
- #define **UEM_SYNC1_ERR** UEMi_MAKE_PARAMID (0x0C, 9, 9)
Наличие ошибок в слове – условие выработки сигнала выходной синхронизации 1.
- #define **UEM_SYNC1_GAPB** UEMi_MAKE_PARAMID (0x0C, 10, 10)
Наличие паузы перед словом - условие выработки сигнала выходной синхронизации 1.
- #define **UEM_SYNC1_CH** UEMi_MAKE_PARAMID (0x0C, 12, 11)
Слово передается по указанной шине - условие выработки сигнала выходной синхронизации 1.
- #define **UEM_IST2** UEMi_MAKE_PARAMID (0x0C, 31, 16)
Период внутренней генерации сигнала входной синхронизации 2.
- #define **UEM_IST1** UEMi_MAKE_PARAMID (8, 31, 16)
Период внутренней генерации сигнала входной синхронизации 1.
- #define **UEM_BM_WORD_PATTERN** UEMi_MAKE_PARAMID (0x0F, 15, 0)
Шаблон (значение) слова - условие выработки сигнала выходной синхронизации 1.
- #define **UEM_BM_WORD_MASK** UEMi_MAKE_PARAMID (0x0F, 31, 16)
Маска побитного сравнения слова с шаблоном - условие выработки сигнала выходной синхронизации 1.

Подробное описание

Описания параметров и определения констант - идентификаторов параметров конфигурации.

В этом разделе перечислены константы, идентифицирующие параметры конфигурации УЭМ. Они предназначены для использования в качестве значения аргумента **paramid** в функциях **uem_param_set()**, **uem_param_get()**.

Для каждого идентификатора параметра приведено описание назначения этого параметра.

Могут быть заданы следующие конфигурационные параметры УЭМ:

- Параметры, задающие включение и отключение шин А и Б.
- Параметры, задающие правила функционирования КШ, ОУ и МШ.
- Параметры, управляющие внесением ошибок в передаваемую в МКПД информацию.
- Параметры, управляющие всеми видами синхронизации.
- Электрические параметры передатчиков.

Макросы

```
#define UEM_TMTA_DIS UEMi_MAKE_PARAMID ( 0, 2, 2)
```

Запрет работы передатчика шины А.

Установленный флаг (UEM_TMTA_DIS=1) запрещает работу передатчика шины А (основной магистрали МКПД).

В этом случае прием передаваемой информации по данной шине виртуальными устройствами производится через внутренние связи (режим «off-line»).

См. **Значения параметра UEM_TMTA_DIS**.

```
#define UEM_TMTB_DIS UEMi_MAKE_PARAMID ( 0, 3, 3)
```

Запрет работы передатчика шины Б.

Установленный флаг (UEM_TMTB_DIS=1) запрещает работу передатчика шины Б (резервной магистрали МКПД).

В этом случае прием передаваемой информации по данной шине виртуальными устройствами производится через внутренние связи (режим «off-line»). См. **Значения параметра UEM_TMTB_DIS**.

```
#define UEM_RCVA_DIS UEMi_MAKE_PARAMID ( 0, 4, 4)
```

Запрет работы приемника шины А.

Установленный флаг (UEM_RCVA_DIS=1) запрещает работу приемника шины А (основной магистрали МКПД).

В этом случае не анализируется состояние входных сигналов управления МКПД, подключаемых к шине А. Прием передаваемой информации по данной шине виртуальными устройствами производится через внутренние связи (режим «off-line»).

См. **Значения параметра UEM_RCVA_DIS**.

```
#define UEM_RCVB_DIS UEMi_MAKE_PARAMID ( 0, 5, 5)
```

Запрет работы приемника шины Б.

Установленный флаг (UEM_RCVB_DIS=1) запрещает работу приемника шины Б (резервной магистрали МКПД).

В этом случае не анализируется состояние входных сигналов управления МКПД, подключаемых к шине Б. Прием передаваемой информации по данной шине виртуальными устройствами производится через внутренние связи (режим «off-line»).

См. Значения параметра UEM_RCVB_DIS.

#define UEM_SYNC_IN_1_ENA UEMi_MAKE_PARAMID (0, 8, 8)

Разрешение входной синхронизации 1.

Установленный флаг (UEM_SYNC_IN_1_ENA=1) разрешает обработку входного сигнала синхронизации 1. Сигнал входной синхронизации 1 (**sync_in_1**) обеспечивает синхронную с внешней аппаратурой пересылку в МКПД слов для КШ.

См. Управление синхронизацией.

См. Значения параметра UEM_SYNC_IN_1_ENA.

#define UEM_SYNC_IN_2_ENA UEMi_MAKE_PARAMID (0, 9, 9)

Разрешение входной синхронизации 2.

Установленный флаг (UEM_SYNC_IN_2_ENA=1) разрешает обработку входного сигнала синхронизации 2. Входной сигнал синхронизации 2 (**sync_in_2**) обеспечивает запуск в работу КШ, предварительно сконфигурированного с требуемыми значениями параметров, а также – повторный запуск.

См. Управление синхронизацией.

См. Значения параметра UEM_SYNC_IN_2_ENA.

#define UEM_SYNC_OUT_1_ENA UEMi_MAKE_PARAMID (0, 10, 10)

Разрешение выходной синхронизации 1.

Установленный флаг (UEM_SYNC_OUT_1_ENA=1) разрешает выработку сигнала выходной синхронизации 1. Сигнал выходной синхронизации 1 (**sync_out_1**), если разрешен, формируется, когда МШ обнаружил слово в МКПД с заданным значением разрядов и заданными характеристиками. Характеристики, определяющие условия выработки сигнала выходной синхронизации 1, задаются параметрами **UEM_SYNC1_C_D**, **UEM_SYNC1_ERR**, **UEM_SYNC1_GAPB**, **UEM_SYNC1_CH**.

См. Управление синхронизацией.

См. Значения параметра UEM_SYNC_OUT_1_ENA.

#define UEM_SYNC_OUT_2_ENA UEMi_MAKE_PARAMID (0, 11, 11)

Разрешение выходной синхронизации 2.

Установленный флаг (UEM_SYNC_OUT_2_ENA=1) разрешает выработку сигнала выходной синхронизации 2. Сигнал выходной синхронизации 2 (**sync_out_2**), если разрешен, вырабатывается, когда МШ обнаружил достоверное командное (ответное) слово с заданным адресом ОУ. Адрес ОУ задается параметром **UEM_SYNC2_VRTA**.

См. Управление синхронизацией.

См. Значения параметра UEM_SYNC_OUT_2_ENA.

#define UEM_BRCST_DIS UEMi_MAKE_PARAMID (0, 12, 12)

Запрет групповых сообщений.

Установленный флаг (UEM_BRCST_DIS=1) запрещает групповые сообщения.

В этом случае ОУ с адресом 31 (если данный адрес был активизирован) используется в качестве «обычного», «не широковещательного» адреса ОУ, используемого для приема и передачи информации.

В аппаратуре этот флаг влияет на подсистему ОУ, в расширенной библиотеке - на работу виртуальных устройств всех типов.

См. **Значения параметра UEM_BRCST_DIS.**

#define UEM_SYNC_IN_1_INTGEN UEMi_MAKE_PARAMID(0, 13, 13)

Разрешение внутренней эмуляции сигнала входной синхронизации 1.

Установленный флаг (UEM_SYNC_IN_1_INTGEN=1) разрешает внутреннюю эмуляцию сигнала входной синхронизации 1.

См. **Управление синхронизацией.**

См. **Значения параметра UEM_SYNC_IN_1_INTGEN.**

#define UEM_ERR_INJ_DIS UEMi_MAKE_PARAMID (0, 14, 14)

Запрет внесения ошибок кодирования в передаваемую в МКПД информацию для КШ и ОУ.

Установленный флаг (UEM_ERR_INJ_DIS=1) запрещает внесения ошибок кодирования в передаваемую в МКПД информацию для КШ и ОУ.

В этом случае информация о внесении ошибок в командные и ответные сегменты сообщений будет игнорироваться.

См. **Значения параметра UEM_ERR_INJ_DIS.**

#define UEM_SYNC_IN_2_INTGEN UEMi_MAKE_PARAMID(0, 29, 29)

Разрешение внутренней эмуляции сигнала входной синхронизации 2.

Установленный флаг (UEM_SYNC_IN_2_INTGEN=1) разрешает внутреннюю эмуляцию сигнала входной синхронизации 2.

См. **Управление синхронизацией.**

См. **Значения параметра UEM_SYNC_IN_2_INTGEN.**

#define UEM_TMT_RES UEMi_MAKE_PARAMID (5, 4, 4)

Сброс настроек передатчиков (только запись).

Параметр доступен только по записи.

При установлении данного параметра конфигурации в 1 осуществляется приведение в исходное состояние настроек по регулировке параметров выходных сигналов передатчиков и установка значений по умолчанию в параметрах **UEM_TXA_RFT**, **UEM_TXA_VPP**, **UEM_TXB_RFT**, **UEM_TXB_VPP**.

См. **Значение UEM_SET.**

#define UEM_SYNC_IN_1_SET UEMi_MAKE_PARAMID (5, 10, 10)

Программная генерация сигнала входной синхронизации 1 (только запись).

Флаг доступен только по записи.

Установка данного флага (UEM_SYNC_IN_1_SET=1) является командой программной имитации поступления входного сигнала внешней синхронизации 1.

См. **Управление синхронизацией.**

См. **Значение UEM_SET.**

#define UEM_SYNC_IN_2_SET UEMi_MAKE_PARAMID (5, 11, 11)

Программная генерация сигнала входной синхронизации 2 (только запись).

Флаг доступен только по записи.

Установка данного флага (UEM_SYNC_IN_2_SET=1) является командой программной имитации поступления входного сигнала внешней синхронизации 2.

См. **Управление синхронизацией.**

См. **Значение UEM_SET.**

#define UEM_SYNC_OUT_1_SET UEMi_MAKE_PARAMID (5, 12, 12)

Программная генерация сигнала выходной синхронизации 1 (только запись).

Флаг доступен только по записи.

Установка данного флага (UEM_SYNC_OUT_1_SET=1) является командой программной имитации выработки выходного сигнала внешней синхронизации 1.

См. **Управление синхронизацией.**

См. **Значение UEM_SET.**

#define UEM_SYNC_OUT_2_SET UEMi_MAKE_PARAMID (5, 13, 13)

Программная генерация сигнала выходной синхронизации 2 (только запись).

Флаг доступен только по записи.

Установка данного флага (UEM_SYNC_OUT_2_SET=1) является командой программной имитации выработки выходного сигнала внешней синхронизации 2.

См. **Управление синхронизацией.**

См. **Значение UEM_SET.**

#define UEM_DB_ACT UEMi_MAKE_PARAMID (5, 31, 31)

Обнаружена передача данных по шине (только чтение).

Флаг доступен только по чтению.

Установленный флаг (UEM_DB_ACT=1) указывает на текущую активность в любой из шин МКПД по информации декодеров, с «инерционностью» 0,5с от момента обнаружения отсутствия активности по обеим шинам МКПД.

См. **Управление синхронизацией.**

#define UEM_TXA_RFT UEMi_MAKE_PARAMID (6, 31, 16)

Управление длительностью фронта и среза при передаче в шину А.

Параметр задает длительность фронта (среза) импульсов выходного сигнала передатчика МКПД для шины А.

В таблице 1 в разделе **Параметры передатчиков и характеристики выходных сигналов** приведена приблизительная оценка соответствия значений данного параметра и значений сигнала МКПД.

См. **Значения параметров UEM_TXA_RFT, UEM_TXB_RFT.**

#define UEM_TXA_VPP UEMi_MAKE_PARAMID (6, 15, 0)

Управление размахом сигнала при передаче в шину А.

Параметр задает размах выходного сигнала передатчика МКПД для шины А. Допустимые значения 0x000E – 0x01FF. Значение по умолчанию 0x0136.

В таблице 2 в разделе **Параметры передатчиков и характеристики выходных сигналов** приведена приблизительная оценка соответствия значений параметров и значений сигналов МКПД при условии работы на номинальную эквивалентную нагрузку по ГОСТ Р 52070.

См. **Значения параметров UEM_TXA_VPP, UEM_TXB_VPP.**

#define UEM_TXB_RFT UEMi_MAKE_PARAMID (7, 31, 16)

Управление длительностью фронта и среза при передаче в шину Б.

Параметр, задающий длительность фронта (среза) импульсов выходного сигнала передатчика МКПД для шины Б.

В таблице 1 в разделе **Параметры передатчиков и характеристики выходных сигналов** приведена приблизительная оценка соответствия значений данного параметра и значений сигнала МКПД.

См. **Значения параметров UEM_TXA_RFT, UEM_TXB_RFT.**

#define UEM_TXB_VPP UEMi_MAKE_PARAMID (7, 15, 0)

Управление размахом сигнала при передаче в шину Б.

Параметр, задающий размах выходного сигнала передатчика МКПД для шины Б. Разряды [15:9] доступны по записи и по чтению. Значения по умолчанию 0136h.

В таблице 2 в разделе **Параметры передатчиков и характеристики выходных сигналов** приведена приблизительная оценка соответствия значений данного параметра и значений сигнала МКПД при условии работы на номинальную эквивалентную нагрузку по ГОСТ Р 52070.

См. **Значения параметров UEM_TXA_VPP, UEM_TXB_VPP.**

#define UEM_MC_DIS UEMi_MAKE_PARAMID (0x0C, 0, 0)

Запрет команд управления.

Установленный флаг (UEM_MC_DIS=1) является признаком «запрета» команд управления, т.е КС с кодами поадресов 00000 и 11111 будут интерпретироваться активными ОУ как команды информационного обмена. В аппаратуре этот флаг влияет на подсистему ОУ, в расширенной библиотеке - на работу виртуальных устройств всех типов.

См. **Значения параметра UEM_MC_DIS.**

#define UEM_BTMT_DIS UEMi_MAKE_PARAMID (0x0C, 1, 1)

Запрет блокирования и разблокирования передатчиков ОУ.

Установленный флаг (UEM_BTMT_DIS=1) является признаком запрета для ОУ блокировок и разблокировок передатчиков при поступлении достоверных КУ «Блокировать передатчик», «Разблокировать передатчик», «Установить ОУ в исходное состояние», в том числе в групповых сообщениях (по умолчанию, при нулевом значении флага, такие блокировки и разблокировки осуществляются индивидуально для каждого адреса ОУ с использованием внутренних сигналов, запрещающих передачу информации по соответствующей шине (шинам)). Имеет смысл только при сброшенном флаге UEM_MC_DIS.

См. Значения параметра UEM_BTMT_DIS.

#define UEM_BRTF_DIS UEMi_MAKE_PARAMID (0x0C, 2, 2)

Запрет блокирования и разблокирования признака неисправности ОУ.

Установленный флаг (UEM_BRTF_DIS=1) является признаком запрета для ОУ блокировок и разблокировок признака «Неисправность ОУ» в ОС при поступлении достоверных КУ «Блокировать признак неисправности ОУ», «Разблокировать признак неисправности ОУ», «Установить ОУ в исходное состояние», в том числе в групповых сообщениях (по умолчанию, при нулевом значении флага такие блокировки и разблокировки осуществляются индивидуально для каждого адреса ОУ с использованием внутренних сигналов, блокирующих установку «1» в соответствующем разряде ОС, исключая режим прямого формирования признаков ОС). Имеет смысл только при сброшенном флаге UEM_MC_DIS.

См. Значения параметра UEM_BRTF_DIS.

#define UEM_SYNC2_VRTA UEMi_MAKE_PARAMID(0x0C, 7, 3)

Номер ОУ – условие выработки сигнала выходной синхронизации 2.

Параметр, задающий адрес ОУ, при обнаружении которого в достоверном КС (ОС) будет выработан выходной сигнал внешней синхронизации 2. Сигнал вырабатывается, если его выработка разрешена.

См. Управление синхронизацией.

См. Значения параметра UEM_SYNC2_VRTA.

#define UEM_SYNC1_C_D_ UEMi_MAKE_PARAMID (0x0C, 8, 8)

Командное/ответное (1) слово или слово данных (0) – условие выработки сигнала выходной синхронизации 1.

Параметр, задающий тип синхроимпульса слова, при обнаружении которого может быть выработан выходной сигнал синхронизации 1 – командное (ответное) (если UEM_SYNC1_C_D_=1) или слово данных (если UEM_SYNC1_C_D_=0). Сигнал вырабатывается, если его выработка разрешена, а также выполнены условия выработки, заданные другими параметрами.

См. Управление синхронизацией.

См. Значения параметра UEM_SYNC1_C_D_.

#define UEM_SYNC1_ERR UEMi_MAKE_PARAMID (0x0C, 9, 9)

Наличие ошибок в слове – условие выработки сигнала выходной синхронизации 1.

Параметр, задающий условие наличия или отсутствия ошибки в слове, при обнаружении которого может быть выработан выходной сигнал синхронизации 1 – при условии наличия ошибки (если значение установлено в «1») или при условии отсутствия ошибки (если значение установлено в «0»). Сигнал вырабатывается, если его выработка разрешена, а также выполнены условия выработки, заданные другими параметрами.

См. **Управление синхронизацией.**

См. **Значения параметра UEM_SYNC1_ERR.**

#define UEM_SYNC1_GAPB UEMi_MAKE_PARAMID (0x0C, 10, 10)

Наличие паузы перед словом - условие выработки сигнала выходной синхронизации 1.

Параметр, при единичном значении которого выходной сигнал внешней синхронизации 1 может вырабатываться только при наличии паузы перед заданным словом, при нулевом значении – только для слова, имеющего соприкосновение с предыдущим словом. Сигнал вырабатывается, если его выработка разрешена, а также выполнены условия выработки, заданные другими параметрами.

См. **Управление синхронизацией.**

См. **Значения параметра UEM_SYNC1_GAPB.**

#define UEM_SYNC1_CH UEMi_MAKE_PARAMID (0x0C, 12, 11)

Слово передается по указанной шине - условие выработки сигнала выходной синхронизации 1.

Данный параметр определяет, будет ли сигнал выходной синхронизации 1 вырабатываться при обнаружении слова в шине А, или в шине Б, или независимо от шины, в которой слово обнаружено. Сигнал вырабатывается, если его выработка разрешена, а также выполнены условия выработки, заданные другими параметрами.

Допустимые значения: **UEM_SYNC1_CH_A, UEM_SYNC1_CH_B, UEM_SYNC1_ACH.**

См. **Управление синхронизацией.**

#define UEM_IST2 UEMi_MAKE_PARAMID (0x0C, 31, 16)

Период внутренней генерации сигнала входной синхронизации 2.

В данном параметре задается период повторения сигнала входной синхронизации 2 при внутренней эмуляции, с разрешением 100 мкс на МЗР.

См. **Управление синхронизацией.**

См. **Значения параметров UEM_IST1, UEM_IST2.**

Нулевое значение параметра интерпретируется аппаратурой как 0x10000 (6,5536 с).

#define UEM_IST1 UEMi_MAKE_PARAMID (8, 31, 16)

Период внутренней генерации сигнала входной синхронизации 1.

В данном параметре задается период повторения сигнала входной синхронизации 1 при внутренней эмуляции с разрешением 100 мкс на МЗР.

См. **Управление синхронизацией.**

См. **Значения параметров UEM_IST1, UEM_IST2.**

Нулевое значение параметра интерпретируется аппаратурой как 0x10000 (6,5536 с).

#define UEM_BM_WORD_PATTERN UEMi_MAKE_PARAMID (0x0F, 15, 0)

Шаблон (значение) слова - условие выработки сигнала выходной синхронизации 1.

В данном параметре задаются значения битов слова, при обнаружении которых может быть выработан сигнал выходной синхронизации 1. На совпадение проверяются только биты, для которых биты в соответствующих позициях параметра **UEM_BM_WORD_MASK** (маски) установлены в 1. Сигнал вырабатывается, если его выработка разрешена, а также выполнены условия выработки, заданные другими параметрами.

См. **Управление синхронизацией**.

См. **Значения параметров UEM_BM_WORD_PATTERN, UEM_BM_WORD_MASK**.

#define UEM_BM_WORD_MASK UEMi_MAKE_PARAMID (0x0F, 31, 16)

Маска побитного сравнения слова с шаблоном - условие выработки сигнала выходной синхронизации 1.

Используется при сравнении обнаруженного слова МКПД с эталоном.

См. **UEM_BM_WORD_PATTERN**.

См. **Управление синхронизацией**.

См. **Значения параметров UEM_BM_WORD_PATTERN, UEM_BM_WORD_MASK**.

Значения параметров

Определения констант - значений параметров конфигурации.

Значения параметра UEM_TMTA_DIS

- **#define UEM_TMTA_DISABLED 1**
Передатчик шины А отключен.
- **#define UEM_TMTA_ENABLED 0**
Передатчик шины А включен.
- **#define UEM_TMTA_DEFAULT (UEM_TMTA_ENABLED)**
По умолчанию: Передатчик шины А включен.

Значения параметра UEM_TMTB_DIS

- **#define UEM_TMTB_DISABLED 1**
Передатчик шины Б отключен.
- **#define UEM_TMTB_ENABLED 0**
Передатчик шины Б включен.
- **#define UEM_TMTB_DEFAULT (UEM_TMTB_ENABLED)**
По умолчанию: Передатчик шины Б включен.

Значения параметра UEM_RCVA_DIS

- **#define UEM_RCVA_DISABLED 1**
Приемник шины А отключен.

- **#define UEM_RCVA_ENABLED 0**
Приемник шины А включен.
- **#define UEM_RCVA_DEFAULT (UEM_RCVA_ENABLED)**
По умолчанию: Приемник шины А включен.

Значения параметра UEM_RCVB_DIS

- **#define UEM_RCVB_DISABLED 1**
Приемник шины Б отключен.
- **#define UEM_RCVB_ENABLED 0**
Приемник шины Б включен.
- **#define UEM_RCVB_DEFAULT (UEM_RCVB_ENABLED)**
По умолчанию: Приемник шины Б включен.

Значения параметра UEM_SYNC_IN_1_ENA

- **#define UEM_SYNC_IN_1_DISABLED 0**
Обработка входного сигнала синхронизации 1 запрещена.
- **#define UEM_SYNC_IN_1_ENABLED 1**
Обработка входного сигнала синхронизации 1 разрешена.
- **#define UEM_SYNC_IN_1_DEFAULT (UEM_SYNC_IN_1_DISABLED)**
По умолчанию: Обработка входного сигнала синхронизации 1 запрещена.

Значения параметра UEM_SYNC_IN_2_ENA

- **#define UEM_SYNC_IN_2_DISABLED 0**
Обработка входного сигнала синхронизации 2 запрещена.
- **#define UEM_SYNC_IN_2_ENABLED 1**
Обработка входного сигнала синхронизации 2 разрешена.
- **#define UEM_SYNC_IN_2_DEFAULT (UEM_SYNC_IN_2_DISABLED)**
По умолчанию: Обработка входного сигнала синхронизации 2 запрещена.

Значения параметра UEM_SYNC_OUT_1_ENA

- **#define UEM_SYNC_OUT_1_DISABLED 0**
Формирование выходного сигнала синхронизации 1 запрещено.
- **#define UEM_SYNC_OUT_1_ENABLED 1**
Формирование выходного сигнала синхронизации 1 разрешено.
- **#define UEM_SYNC_OUT_1_DEFAULT (UEM_SYNC_OUT_1_DISABLED)**
По умолчанию: Формирование выходного сигнала синхронизации 1 запрещено.

Значения параметра UEM_SYNC_OUT_2_ENA

- **#define UEM_SYNC_OUT_2_DISABLED 0**
Формирование выходного сигнала синхронизации 2 запрещено.
- **#define UEM_SYNC_OUT_2_ENABLED 1**
Формирование выходного сигнала синхронизации 2 разрешено.

- **#define UEM_SYNC_OUT_2_DEFAULT (UEM_SYNC_OUT_2_DISABLED)**
По умолчанию: Формирование выходного сигнала синхронизации 2 запрещено.

Значения параметра UEM_BRCST_DIS

- **#define UEM_BRCST_DISABLED 1**
Групповые сообщения запрещены.
- **#define UEM_BRCST_ENABLED 0**
Групповые сообщения разрешены.
- **#define UEM_BRCST_DEFAULT (UEM_BRCST_ENABLED)**
По умолчанию: Групповые сообщения разрешены.

Значения параметра UEM_SYNC_IN_1_INTGEN

- **#define UEM_SYNC_IN_1_INTGEN_DISABLED 0**
Внутренняя генерация запрещена.
- **#define UEM_SYNC_IN_1_INTGEN_ENABLED 1**
Внутренняя генерация разрешена.
- **#define UEM_SYNC_IN_1_INTGEN_DEFAULT (UEM_SYNC_IN_1_INTGEN_DISABLED)**
По умолчанию: запрещена.

Значения параметра UEM_SYNC_IN_2_INTGEN

- **#define UEM_SYNC_IN_2_INTGEN_DISABLED 0**
Внутренняя генерация запрещена.
- **#define UEM_SYNC_IN_2_INTGEN_ENABLED 1**
Внутренняя генерация разрешена.
- **#define UEM_SYNC_IN_2_INTGEN_DEFAULT (UEM_SYNC_IN_2_INTGEN_DISABLED)**
По умолчанию: запрещена.

Значения параметра UEM_ERR_INJ_DIS

- **#define UEM_ERR_INJ_DISABLED 1**
Внесение ошибок кодирования запрещено.
- **#define UEM_ERR_INJ_ENABLED 0**
Внесение ошибок кодирования разрешено.
- **#define UEM_ERR_INJ_DEFAULT (UEM_ERR_INJ_ENABLED)**
По умолчанию: Внесение ошибок кодирования разрешено.

Значение UEM_SET

Значение **UEM_SET**, тождественно равно 1, используется с параметрами, фактически являющимися командами, вызывающими определенные действия аппаратуры УЭМ. Такие параметры доступны только по записи, причем действенной является именно запись значения 1. Такие параметры не имеют и не требуют установки значений по умолчанию.

- **#define UEM_SET 1**
Установка параметра в 1.

Значения параметра UEM_DB_ACT

Данный параметр доступен только по чтению, он не имеет и не требует установки значения по умолчанию.

- #define UEM_DB_INACTIVE 0
Нет активности на шине данных.
- #define UEM_DB_ACTIVE 1
Обнаружена активность на шине данных.

Значения параметров UEM_TXA_RFT, UEM_TXB_RFT

- #define UEM_RFT_MIN 0
Минимальное значение.
- #define UEM_RFT_MAX 31
Максимальное значение.
- #define UEM_RFT_SINE 32
Специальное значение. Включает формирование синусоидального сигнала.
- #define UEM_RFT_DEFAULT 18
Значение по умолчанию (18).

Значения параметров UEM_TXA_VPP, UEM_TXB_VPP

- #define UEM_VPP_MIN 0x00E
Минимальное значение.
- #define UEM_VPP_MAX 0x01FF
Максимальное значение.
- #define UEM_VPP_DEFAULT 0x0136
Значение по умолчанию (0x0136).

Значения параметра UEM_MC_DIS

- #define UEM_MC_DISABLED 1
Команды управления запрещены.
- #define UEM_MC_ENABLED 0
Команды управления разрешены.
- #define UEM_MC_DEFAULT (UEM_MC_ENABLED)
По умолчанию: Команды управления разрешены.

Значения параметра UEM_BTMT_DIS

- #define UEM_BTMT_DISABLED 1
Блокирование и разблокирование передатчиков ОУ запрещено.
- #define UEM_BTMT_ENABLED 0
Блокирование и разблокирование передатчиков ОУ разрешено.
- #define UEM_BTMT_DEFAULT (UEM_BTMT_ENABLED)
По умолчанию: Блокирование и разблокирование передатчиков ОУ разрешено.

Значения параметра UEM_BRTF_DIS

- **#define UEM_BRTF_DISABLED 1**
Блокирование и разблокирование признака неисправности ОУ запрещено.
- **#define UEM_BRTF_ENABLED 0**
Блокирование и разблокирование признака неисправности ОУ разрешено.
- **#define UEM_BRTF_DEFAULT (UEM_BRTF_ENABLED)**
По умолчанию: Блокирование и разблокирование признака неисправности ОУ разрешено.

Значения параметра UEM_SYNC2_VRTA

- **#define UEM_SYNC2_VRTA_MIN 0**
Минимальное значение.
- **#define UEM_SYNC2_VRTA_MAX 31**
Максимальное значение.
- **#define UEM_SYNC2_VRTA_DEFAULT 0**
Значение по умолчанию.

Значения параметра UEM_SYNC1_C_D_

- **#define UEM_SYNC1_ON_COMMAND 1**
Синхроимпульс командного/ответного слова.
- **#define UEM_SYNC1_ON_DATA 0**
Синхроимпульс слова данных.
- **#define UEM_SYNC1_C_D_DEFAULT (UEM_SYNC1_ON_DATA)**
По умолчанию: Синхроимпульс слова данных.

Значения параметра UEM_SYNC1_ERR

- **#define UEM_SYNC1_ON_ERROR 1**
При наличии ошибки в слове.
- **#define UEM_SYNC1_ON_NO_ERROR 0**
При отсутствии ошибки в слове.
- **#define UEM_SYNC1_ERR_DEFAULT (UEM_SYNC1_ON_NO_ERROR)**
По умолчанию: При отсутствии ошибки в слове.

Значения параметра UEM_SYNC1_GAPB

- **#define UEM_SYNC1_ON_GAPB 1**
При наличии паузы перед словом.
- **#define UEM_SYNC1_ON_NO_GAPB 0**
При отсутствии паузы перед словом.
- **#define UEM_SYNC1_GAPB_DEFAULT (UEM_SYNC1_ON_NO_GAPB)**
По умолчанию: При отсутствии паузы перед словом.

Значения параметра UEM_SYNC1_CH

- `#define UEM_SYNC1_CH_A 0`
Слово передается по шине А.
- `#define UEM_SYNC1_CH_B 2`
Слово передается по шине Б.
- `#define UEM_SYNC1_CH 1`
Слово передается по любой шине.
- `#define UEM_SYNC1_CH_DEFAULT (UEM_SYNC1_CH_A)`
Значение UEM_SYNC1_CH по умолчанию: по шине А.

Значения параметров UEM_IST1, UEM_IST2

- `#define UEM_IST_MIN 1`
Минимальное значение.
- `#define UEM_IST_MAX 65536`
Максимальное значение.
- `#define UEM_IST_DEFAULT (UEM_IST_MAX)`
Значение по умолчанию.

Значения параметров UEM_BM_WORD_PATTERN, UEM_BM_WORD_MASK

- `#define UEM_BM_WORD_MIN 0x0000`
Минимальное значение.
- `#define UEM_BM_WORD_MAX 0xFFFF`
Максимальное значение.
- `#define UEM_BM_WORD_DEFAULT (UEM_BM_WORD_MIN)`
Значение по умолчанию.

Подробное описание

Определения констант - значений параметров конфигурации.

Константы - значения параметров конфигурации - определены для задания значений параметров конфигурации в более понятной форме, а также показывают значения параметров по умолчанию. Эти константы предназначены для использования в качестве значений аргумента **value** в функции **uem_param_set()**. Их также можно использовать для сравнения со значениями, возвращаемыми в аргументе **value** в функции **uem_param_get()**. Использование этих констант не обязательно.

Макросы

`#define UEM_TMTA_DISABLED 1`

Передатчик шины А отключен.

Используется с **UEM_TMTA_DIS**.

#define UEM_TMTA_ENABLED 0

Передатчик шины А включен.

Используется с **UEM_TMTA_DIS**.

#define UEM_TMTA_DEFAULT (UEM_TMTA_ENABLED)

По умолчанию: Передатчик шины А включен.

Используется с **UEM_TMTA_DIS**.

#define UEM_TMTB_DISABLED 1

Передатчик шины Б отключен.

Используется с **UEM_TMTB_DIS**.

#define UEM_TMTB_ENABLED 0

Передатчик шины Б включен.

Используется с **UEM_TMTB_DIS**.

#define UEM_TMTB_DEFAULT (UEM_TMTB_ENABLED)

По умолчанию: Передатчик шины Б включен.

Используется с **UEM_TMTB_DIS**.

#define UEM_RCVA_DISABLED 1

Приемник шины А отключен.

Используется с **UEM_RCVA_DIS**.

#define UEM_RCVA_ENABLED 0

Приемник шины А включен.

Используется с **UEM_RCVA_DIS**.

#define UEM_RCVA_DEFAULT (UEM_RCVA_ENABLED)

По умолчанию: Приемник шины А включен.

Используется с **UEM_RCVA_DIS**.

#define UEM_RCVB_DISABLED 1

Приемник шины Б отключен.

Используется с **UEM_RCVB_DIS**.

#define UEM_RCVB_ENABLED 0

Приемник шины Б включен.

Используется с **UEM_RCVB_DIS**.

#define UEM_RCVB_DEFAULT (UEM_RCVB_ENABLED)

По умолчанию: Приемник шины Б включен.
Используется с **UEM_RCVB_DIS**.

#define UEM_SYNC_IN_1_DISABLED 0

Обработка входного сигнала синхронизации 1 запрещена.
Используется с **UEM_SYNC_IN_1_ENA**.

#define UEM_SYNC_IN_1_ENABLED 1

Обработка входного сигнала синхронизации 1 разрешена.
Используется с **UEM_SYNC_IN_1_ENA**.

#define UEM_SYNC_IN_1_DEFAULT (UEM_SYNC_IN_1_DISABLED)

По умолчанию: Обработка входного сигнала синхронизации 1 запрещена.
Используется с **UEM_SYNC_IN_1_ENA**.

#define UEM_SYNC_IN_2_DISABLED 0

Обработка входного сигнала синхронизации 2 запрещена.
Используется с **UEM_SYNC_IN_2_ENA**.

#define UEM_SYNC_IN_2_ENABLED 1

Обработка входного сигнала синхронизации 2 разрешена.
Используется с **UEM_SYNC_IN_2_ENA**.

#define UEM_SYNC_IN_2_DEFAULT (UEM_SYNC_IN_2_DISABLED)

По умолчанию: Обработка входного сигнала синхронизации 2 запрещена.
Используется с **UEM_SYNC_IN_2_ENA**.

#define UEM_SYNC_OUT_1_DISABLED 0

Формирование выходного сигнала синхронизации 1 запрещено.
Используется с **UEM_SYNC_OUT_1_ENA**.

#define UEM_SYNC_OUT_1_ENABLED 1

Формирование выходного сигнала синхронизации 1 разрешено.
Используется с **UEM_SYNC_OUT_1_ENA**.

#define UEM_SYNC_OUT_1_DEFAULT (UEM_SYNC_OUT_1_DISABLED)

По умолчанию: Формирование выходного сигнала синхронизации 1 запрещено.
Используется с **UEM_SYNC_OUT_1_ENA**.

#define UEM_SYNC_OUT_2_DISABLED 0

Формирование выходного сигнала синхронизации 2 запрещено.
Используется с **UEM_SYNC_OUT_2_ENA**.

#define UEM_SYNC_OUT_2_ENABLED 1

Формирование выходного сигнала синхронизации 2 разрешено.
Используется с **UEM_SYNC_OUT_2_ENA**.

#define UEM_SYNC_OUT_2_DEFAULT (UEM_SYNC_OUT_2_DISABLED)

По умолчанию: Формирование выходного сигнала синхронизации 2 запрещено.
Используется с **UEM_SYNC_OUT_2_ENA**.

#define UEM_BRCST_DISABLED 1

Групповые сообщения запрещены.
Используется с **UEM_BRCST_DIS**.

#define UEM_BRCST_ENABLED 0

Групповые сообщения разрешены.
Используется с **UEM_BRCST_DIS**.

#define UEM_BRCST_DEFAULT (UEM_BRCST_ENABLED)

По умолчанию: Групповые сообщения разрешены.
Используется с **UEM_BRCST_DIS**.

#define UEM_SYNC_IN_1_INTGEN_DISABLED 0

Внутренняя генерация запрещена.
Используется с **UEM_SYNC_IN_1_INTGEN**.

#define UEM_SYNC_IN_1_INTGEN_ENABLED 1

Внутренняя генерация разрешена.
Используется с **UEM_SYNC_IN_1_INTGEN**.

#define UEM_SYNC_IN_1_INTGEN_DEFAULT (UEM_SYNC_IN_1_INTGEN_DISABLED)

По умолчанию: запрещена.

Используется с **UEM_SYNC_IN_1_INTGEN**.

#define UEM_SYNC_IN_2_INTGEN_DISABLED 0

Внутренняя генерация запрещена.

Используется с **UEM_SYNC_IN_2_INTGEN**.

#define UEM_SYNC_IN_2_INTGEN_ENABLED 1

Внутренняя генерация разрешена.

Используется с **UEM_SYNC_IN_2_INTGEN**.

#define UEM_SYNC_IN_2_INTGEN_DEFAULT (UEM_SYNC_IN_2_INTGEN_DISABLED)

По умолчанию: запрещена.

Используется с **UEM_SYNC_IN_2_INTGEN**.

#define UEM_ERR_INJ_DISABLED 1

Внесение ошибок кодирования запрещено.

Используется с **UEM_ERR_INJ_DIS**.

#define UEM_ERR_INJ_ENABLED 0

Внесение ошибок кодирования разрешено.

Используется с **UEM_ERR_INJ_DIS**.

#define UEM_ERR_INJ_DEFAULT (UEM_ERR_INJ_ENABLED)

По умолчанию: Внесение ошибок кодирования разрешено.

Используется с **UEM_ERR_INJ_DIS**.

#define UEM_SET 1

Установка параметра в 1.

Используется с **UEM_TMT_RES, UEM_SYNC_IN_1_SET, UEM_SYNC_IN_2_SET, UEM_SYNC_OUT_1_SET, UEM_SYNC_OUT_2_SET**.

#define UEM_DB_INACTIVE 0

Нет активности на шине данных.

Используется с **UEM_DB_ACT**.

#define UEM_DB_ACTIVE 1

Обнаружена активность на шине данных.

Используется с **UEM_DB_ACT**.

#define UEM_RFT_MIN 0

Минимальное значение.

Используется с **UEM_TXA_RFT, UEM_TXB_RFT.**

#define UEM_RFT_MAX 31

Максимальное значение.

Используется с **UEM_TXA_RFT, UEM_TXB_RFT.**

#define UEM_RFT_SINE 32

Специальное значение. Включает формирование синусоидального сигнала.

Используется с **UEM_TXA_RFT, UEM_TXB_RFT.**

#define UEM_RFT_DEFAULT 18

Значение по умолчанию (18).

Используется с **UEM_TXA_RFT, UEM_TXB_RFT.**

#define UEM_VPP_MIN 0x00E

Минимальное значение.

Используется с **UEM_TXA_VPP, UEM_TXB_VPP.**

#define UEM_VPP_MAX 0x01FF

Максимальное значение.

Используется с **UEM_TXA_VPP, UEM_TXB_VPP.**

#define UEM_VPP_DEFAULT 0x0136

Значение по умолчанию (0x0136).

Используется с **UEM_TXA_VPP, UEM_TXB_VPP.**

#define UEM_MC_DISABLED 1

Команды управления запрещены.

Используется с **UEM_MC_DIS.**

#define UEM_MC_ENABLED 0

Команды управления разрешены.

Используется с **UEM_MC_DIS.**

#define UEM_MC_DEFAULT (UEM_MC_ENABLED)

По умолчанию: Команды управления разрешены.

Используется с **UEM_MC_DIS**.

#define UEM_BTMT_DISABLED 1

Блокирование и разблокирование передатчиков ОУ запрещено.
Используется с **UEM_BTMT_DIS**.

#define UEM_BTMT_ENABLED 0

Блокирование и разблокирование передатчиков ОУ разрешено.
Используется с **UEM_BTMT_DIS**.

#define UEM_BTMT_DEFAULT (UEM_BTMT_ENABLED)

По умолчанию: Блокирование и разблокирование передатчиков ОУ разрешено.
Используется с **UEM_BTMT_DIS**.

#define UEM_BRTF_DISABLED 1

Блокирование и разблокирование признака неисправности ОУ запрещено.
Используется с **UEM_BRTF_DIS**.

#define UEM_BRTF_ENABLED 0

Блокирование и разблокирование признака неисправности ОУ разрешено.
Используется с **UEM_BRTF_DIS**.

#define UEM_BRTF_DEFAULT (UEM_BRTF_ENABLED)

По умолчанию: Блокирование и разблокирование признака неисправности ОУ разрешено.
Используется с **UEM_BRTF_DIS**.

#define UEM_SYNC2_VRTA_MIN 0

Минимальное значение.
Используется с **UEM_SYNC2_VRTA**.

#define UEM_SYNC2_VRTA_MAX 31

Максимальное значение.
Используется с **UEM_SYNC2_VRTA**.

#define UEM_SYNC2_VRTA_DEFAULT 0

Значение по умолчанию.
Используется с **UEM_SYNC2_VRTA**.

#define UEM_SYNC1_ON_COMMAND 1

Синхроимпульс командного/ответного слова.
Используется с **UEM_SYNC1_C_D_**.

#define UEM_SYNC1_ON_DATA 0

Синхроимпульс слова данных.
Используется с **UEM_SYNC1_C_D_**.

#define UEM_SYNC1_C_D_DEFAULT (UEM_SYNC1_ON_DATA)

По умолчанию: Синхроимпульс слова данных.
Используется с **UEM_SYNC1_C_D_**.

#define UEM_SYNC1_ON_ERROR 1

При наличии ошибки в слове.
Используется с **UEM_SYNC1_ERR**.

#define UEM_SYNC1_ON_NO_ERROR 0

При отсутствии ошибки в слове.
Используется с **UEM_SYNC1_ERR**.

#define UEM_SYNC1_ERR_DEFAULT (UEM_SYNC1_ON_NO_ERROR)

По умолчанию: При отсутствии ошибки в слове.
Используется с **UEM_SYNC1_ERR**.

#define UEM_SYNC1_ON_GAPB 1

При наличии паузы перед словом.
Используется с **UEM_SYNC1_GAPB**.

#define UEM_SYNC1_ON_NO_GAPB 0

При отсутствии паузы перед словом.
Используется с **UEM_SYNC1_GAPB**.

#define UEM_SYNC1_GAPB_DEFAULT (UEM_SYNC1_ON_NO_GAPB)

По умолчанию: При отсутствии паузы перед словом.
Используется с **UEM_SYNC1_GAPB**.

#define UEM_SYNC1_CH_A 0

Слово передается по шине А.

Используется с **UEM_SYNC1_CH**.

#define UEM_SYNC1_CH_B 2

Слово передается по шине Б.

Используется с **UEM_SYNC1_CH**.

#define UEM_SYNC1_ACH 1

Слово передается по любой шине.

Используется с **UEM_SYNC1_CH**.

#define UEM_SYNC1_CH_DEFAULT (UEM_SYNC1_CH_A)

Значение **UEM_SYNC1_CH** по умолчанию: по шине А.

Используется с **UEM_SYNC1_CH**.

#define UEM_IST_MIN 1

Минимальное значение.

Используется с **UEM_IST1, UEM_IST2**.

#define UEM_IST_MAX 65536

Максимальное значение.

Используется с **UEM_IST1, UEM_IST2**.

#define UEM_IST_DEFAULT (UEM_IST_MAX)

Значение по умолчанию.

Используется с **UEM_IST1, UEM_IST2**.

#define UEM_BM_WORD_MIN 0x0000

Минимальное значение.

Используется с **UEM_BM_WORD_PATTERN, UEM_BM_WORD_MASK**.

#define UEM_BM_WORD_MAX 0xFFFF

Максимальное значение.

Используется с **UEM_BM_WORD_PATTERN, UEM_BM_WORD_MASK**.

#define UEM_BM_WORD_DEFAULT (UEM_BM_WORD_MIN)

Значение по умолчанию.

Используется с **UEM_BM_WORD_PATTERN, UEM_BM_WORD_MASK**.

Параметры интервалов времени

Описания параметров временных интервалов и функций для работы с ними.

Макросы

- #define **UEM_MIN_T1_DEFAULT** (4*4)
Минимальная пауза между командным и ответным сегментами (min t1), значение по умолчанию.
- #define **UEM_MIN_T2_DEFAULT** (4*4)
Минимальная пауза между сообщениями (min t2), значение по умолчанию.
- #define **UEM_RTMO_DEFAULT** (14*4)
Таймаут ответа ОУ, значение по умолчанию.

Перечисления

- enum **UEM_TIME_PARAM** { **UEM_MIN_T1**, **UEM_MIN_T2**, **UEM_RTMO** }
Идентификатор (селектор) параметра интервала времени

Функции

- ViStatus **uem_timing_set** (UEM_DEVHANDLE anydev, UEM_TIME_PARAM param, UEM_DWORD value)
Установка параметра интервала времени.
- ViStatus **uem_timing_get** (UEM_DEVHANDLE anydev, UEM_TIME_PARAM param, UEM_DWORD *value)
Считывание параметра интервала времени.

Подробное описание

Описания параметров временных интервалов и функций для работы с ними.

Параметры интервалов времени - это нормативные значения интервалов времени между сообщениями или частями сообщений. Имеется три таких параметра:

- минимальная пауза между командным и ответным сегментами min t1,
- минимальная пауза между сообщениями min t2,
- максимальное время ожидания (таймаут) ответа ОУ.

Смысл и значения этих параметров определяются в [1].

В отличие от конфигурационных параметров (см. **Параметры конфигурации УЭМ**), параметры интервалов времени:

- являются параметрами программного обеспечения, а не аппаратуры,
- могут быть заданы как для УЭМ в целом, так и отдельно для виртуальных устройств (КШ/ОУ/МШ) в составе УЭМ.

В виртуальном КШ параметры интервалов времени используются для расчета стандартных пауз между сообщениями. В виртуальном ОУ параметр min t1 используется как значение стандартной паузы перед передачей ответного сегмента. В виртуальном МШ все три параметра используются в алгоритме разбора трассы, для разделения потока слов на сообщения и детектирования ошибок нарушения минимальных и максимальных пауз.

Значение задается в единицах, равных 0,25 мкс.

По умолчанию параметры интервалов времени заданы в соответствии с ГОСТ [1].

Для специальных целей параметры интервалов времени могут быть установлены в нестандартные значения.

Макросы

#define UEM_MIN_T1_DEFAULT (4*4)

Минимальная пауза между командным и ответным сегментами (min t1), значение по умолчанию.

#define UEM_MIN_T2_DEFAULT (4*4)

Минимальная пауза между сообщениями (min t2), значение по умолчанию.

#define UEM_RTMO_DEFAULT (14*4)

Таймаут ответа ОУ, значение по умолчанию.

Перечисления

enum UEM_TIME_PARAM

Идентификатор (селектор) параметра интервала времени.

Элементы перечислений:

UEM_MIN_T1 Минимальная пауза между командным и ответным сегментами (min t1).

UEM_MIN_T2 Минимальная пауза между сообщениями (min t2).

UEM_RTMO Таймаут ответа ОУ.

Функции

ViStatus uem_timing_set (UEM_DEVHANDLE *anydev*, UEM_TIME_PARAM *param*, UEM_DWORD *value*)

Установка параметра интервала времени.

Аргументы:

in	<i>anydev</i>	Дескриптор устройства или виртуального устройства.
in	<i>param</i>	Идентификатор параметра (элемент перечисления UEM_TIME_PARAM).
in	<i>value</i>	Значение параметра. В единицах по 0,25 мкс.

Возвращает:

Код завершения. См. **Коды завершения**.

При задании в аргументе **anydev** дескриптора виртуального устройства указанный параметр устанавливается для указанного виртуального устройства.

При задании в аргументе **anydev** дескриптора УЭМ указанный параметр устанавливается для всех виртуальных устройств этого УЭМ.

**ViStatus uem_timing_get (UEM_DEVHANDLE *anydev*, UEM_TIME_PARAM *param*, UEM_DWORD *
value)**

Считывание параметра интервала времени.

Аргументы:

in	<i>anydev</i>	Дескриптор устройства или виртуального устройства.
in	<i>param</i>	Идентификатор параметра (элемент перечисления UEM_TIME_PARAM).
out	<i>value</i>	Значение параметра. В единицах по 0,25 мкс.

Возвращает:

Код завершения. См. **Коды завершения**.

Встроенный счетчик времени

Функции для работы со встроенным счетчиком времени и метками времени.

Структуры данных

- union **UEM_TIME_TAG**
Формат метки времени

Формат метки времени

Определения типов

- typedef ViUInt64 **UEM_TIME_TAG_LIN**
Метка времени в линейном формате.

Функции

- ViStatus **uem_time_tag_get** (UEM_DEVHANDLE uem, UEM_TIME_TAG *time_tag)
Считывание встроенного счетчика времени.
- ViStatus **uem_time_tag_set** (UEM_DEVHANDLE uem, UEM_TIME_TAG *time_tag)
Установка значения встроенного счетчика времени.
- ViStatus **uem_time_tag_reset** (UEM_DEVHANDLE uem)
Сброс встроенного счетчика времени.
- **UEM_TIME_TAG_LIN uem_time_tag_to_linear** (UEM_TIME_TAG *time_tag)
Перевод метки времени в линейный формат.
- void **uem_time_tag_to_struct** (UEM_TIME_TAG *time_tag, UEM_TIME_TAG_LIN linear)
Перевод метки времени из линейного в структурированный формат.

Подробное описание

Функции для работы со встроенным счетчиком времени и метками времени.

В УЭМ имеется встроенный счетчик времени с разрешением 0,25 мкс на единицу младшего разряда и диапазоном 366 суток.

В этом разделе предоставляются функции считывания и установки значения счетчика времени. Эти значения будем называть метками времени. Это структурированные значения, описываемые типом данных **UEM_TIME_TAG**. Метки времени в таком же формате используются в функциях монитора шины.

Рекомендуется в начале работы приложения с УЭМ установить счетчик времени в некоторое определенной значение. Это лучше всего сделать одним из следующих способов:

- установить счетчик времени в значение, соответствующее текущей дате и времени (при помощи **uem_time_tag_set()**), в этом случае счетчик времени показывает текущее время,
- установить счетчик времени в значение 0 (при помощи **uem_time_tag_reset()**), в этом случае счетчик времени показывает интервал времени, прошедший с момента запуска приложения.

Для вычислений с метками времени иногда удобнее не структурированный, а линейный формат, который представляет собой простое количество единиц времени по 0,25 мкс, прошедших с момента заданного "начала времени". Для перевода метки времени в линейный формат и обратно служат функции **uem_time_tag_to_linear()** и **uem_time_tag_to_struct()**.

Типы

typedef ViUInt64 UEM_TIME_TAG_LIN

Метка времени в линейном формате.
В единицах по 0,25 мкс.

Функции

ViStatus uem_time_tag_get (UEM_DEVHANDLE *uem*, UEM_TIME_TAG * *time_tag*)

Считывание встроенного счетчика времени.

Аргументы:

in	<i>uem</i>	- Дескриптор УЭМ. Допускается указывать вместо дескриптора УЭМ дескриптор любого виртуального устройства в составе этого УЭМ. Функция выполняет такой вызов, как если бы был указан дескриптор УЭМ.
out	<i>time_tag</i>	- В этом аргументе передается адрес структуры, в которую будет записано текущее значение счетчика.

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_time_tag_set (UEM_DEVHANDLE *uem*, UEM_TIME_TAG * *time_tag*)

Установка значения встроенного счетчика времени.

Аргументы:

in	<i>uem</i>	- Дескриптор УЭМ. Допускается указывать вместо дескриптора УЭМ дескриптор любого виртуального устройства в составе этого УЭМ. Функция выполняет такой вызов, как если бы был указан дескриптор УЭМ.
in	<i>time_tag</i>	- В этом аргументе передается адрес структуры, из которой берется новое значение счетчика времени.

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_time_tag_reset (UEM_DEVHANDLE *uem*)

Сброс встроенного счетчика времени.
Встроенный счетчик устанавливается в значение 0.

Аргументы:

in	<i>uem</i>	- Дескриптор УЭМ. Допускается указывать вместо дескриптора УЭМ дескриптор любого виртуального устройства в составе этого УЭМ. Функция выполняет такой вызов, как если бы был указан дескриптор УЭМ.
----	------------	--

Возвращает:

Код завершения. См. **Коды завершения**.

UEM_TIME_TAG_LIN uem_time_tag_to_linear (UEM_TIME_TAG * *time_tag*)

Перевод метки времени в линейный формат.

Аргументы:

in	<i>time_tag</i>	- В этом аргументе передается адрес структуры, содержащей значение метки времени.
----	-----------------	---

Возвращает:

Значение метки времени в линейном формате.

void uem_time_tag_to_struct (UEM_TIME_TAG * *time_tag*, UEM_TIME_TAG_LIN *linear*)

Перевод метки времени из линейного в структурированный формат.

Аргументы:

out	<i>time_tag</i>	- В этом аргументе передается адрес структуры, в которую будет записано значение метки времени.
in	<i>linear</i>	- Значение метки времени в линейном формате.

Возвращает:

Функция не возвращает значения.

Определения типов данных для КШ, ОУ, МШ

Определения типов данных для КШ, ОУ, МШ.

Структуры данных

- struct **UEM_DATA**
Блок слов данных.
- struct **UEM_CMD_SEG**
Образ командного сегмента.
- struct **UEM_RESP_SEG**
Ответный сегмент.
- struct **UEM_RAW_BM_MESSAGE**
Принятое сообщение в аппаратном формате.
- struct **UEM_SEGMENT_DESCR**
Описатель сегмента в мониторе шины.
- struct **UEM_BM_MESSAGE**
Разобранное сообщение МШ.

Определения типов

- typedef ViUInt32 **UEM_ERROR_FLAGS**
Признаки ошибок распознавания сообщения в мониторе шины.

Перечисления

- enum **UEM_FORMAT** { **UEM_UNF**, **UEM_F1**, **UEM_F2**, **UEM_F3**, **UEM_F4**, **UEM_F5**, **UEM_F6**, **UEM_F7**, **UEM_F8**, **UEM_F9**, **UEM_F10** }
Форматы сообщений (номера по ГОСТ [1]).
- enum **UEM_CHANNEL** { **UEM_CH_A**, **UEM_CH_B** }
Селектор шины (А/Б).
- enum **UEM_SYNC** { **UEM_SYNC_D**, **UEM_SYNC_C** }
Селектор синхроимпульса.

Более mnemonicные обозначения форматов сообщений

- #define **UEM_BCRT** **UEM_F1**
КШОУ.
- #define **UEM_RTBC** **UEM_F2**
ОУКШ.
- #define **UEM_RTRT** **UEM_F3**
ОВОУ.
- #define **UEM_MC** **UEM_F4**
Команда управления.
- #define **UEM_MCRTBC** **UEM_F5**
Команда управления со словом данных, передаваемым от ОУ к КШ.
- #define **UEM_MCBCRT** **UEM_F6**
Команда управления со словом данных, передаваемым от КШ к ОУ.
- #define **UEM_BCRTь** **UEM_F7**
КШОУ ГРУППОВОЕ.
- #define **UEM_RTRTь** **UEM_F8**
ОВОУ ГРУППОВОЕ.
- #define **UEM_MCь** **UEM_F9**
Команда управления групповая.

- **#define UEM_MCBCRTЬ UEM_F10**
Команда управления групповая со словом данных, передаваемым от КШ к ОУ.

Число слов данных

- **#define UEM_NDATA_MIN 0**
Число слов данных - минимальное значение.
- **#define UEM_NDATA_MAX 62**
Число слова данных - максимальное значение.
- **#define UEM_NDATA_BY_CW 63**
Число слов данных определяется командным словом.

Признаки ошибок распознавания сообщения

Комбинации перечисленных констант (по |) определяют содержание элементов типа **UEM_ERROR_FLAGS**.

- **#define UEM_ERRF_ERROR 1**
Наличие любой ошибки (суммарный флаг).
- **#define UEM_ERRF_ENCODING (1<<1)**
Наличие ошибки кодирования слов (суммарный флаг).
- **#define UEM_ERRF_FORMAT (1<<2)**
Наличие нарушения формата, ошибки состава сообщения (суммарный флаг).
- **#define UEM_ERRF_MINGAP (1<<3)**
Временной интервал меньше допустимого.
- **#define UEM_ERRF_NO_RESPONSE (1<<4)**
Отсутствие ответа.
- **#define UEM_ERRF_SYNC_TYPE (1<<5)**
Неверный тип синхроимпульса.
- **#define UEM_ERRF_MISSING_CWSW (1<<6)**
Отсутствует командное или ответное слово.
- **#define UEM_ERRF_EXTRA_CWSW (1<<7)**
Лишнее командное или ответное слово.
- **#define UEM_ERRF_MISSING_DW (1<<8)**
Недостаточно слов данных.
- **#define UEM_ERRF_EXTRA_DW (1<<9)**
Лишние слова данных.
- **#define UEM_ERRF_INCORRECT_RTN (1<<10)**
Некорректный адрес ОУ.
- **#define UEM_ERRF_RTRT_FORMAT (1<<11)**
Ошибка формата ОУОУ (одинаковые адреса ОУ, несовпадение числа СД).
- **#define UEM_ERRF_INC_MODE_CODE (1<<12)**
Некорректная команда управления.
- **#define UEM_ERRF_FORMAT_MC (1<<13)**
Ошибка формата команды управления.
- **#define UEM_ERRF_GAPN (1<<20)**
Недостовверная информация (сигнал) во время паузы перед словом.
- **#define UEM_ERRF_PARITY (1<<21)**
Ошибка четности.
- **#define UEM_ERRF_LESS_BITS (1<<22)**

- *Укороченное слово.*
• **#define UEM_ERRF_MORE_BITS** (1<<23)
Удлинённое слово.
 - **#define UEM_ERRF_ENC** (1<<24)
Ошибка бифазного кодирования.
 - **#define UEM_ERRF_DT** (1<<30)
Несоблюдение минимальной паузы перед словом, по данным аппаратного декодера.
 - **#define UEM_ERRF_ENCODING2** (1<<31)
Наличие любой ошибки кодирования слов (суммарный флаг, по данным аппаратного декодера).
-

Подробное описание

Определения типов данных для КШ, ОУ, МШ.

Данный раздел содержит определения структур и типов данных, ориентированных на протокол МКПД [1] и использующихся в функциях виртуальных КШ, ОУ, МШ.

Макросы

#define UEM_BCRT UEM_F1

КШОУ.

#define UEM_RTBC UEM_F2

ОУКШ.

#define UEM_RTRT UEM_F3

ОУОУ.

#define UEM_MC UEM_F4

Команда управления.

#define UEM_MCRTBC UEM_F5

Команда управления со словом данных, передаваемым от ОУ к КШ.

#define UEM_MBCRT UEM_F6

Команда управления со словом данных, передаваемым от КШ к ОУ.

#define UEM_BCRTb UEM_F7

КШОУ ГРУППОВОЕ.

#define UEM_RTRTb UEM_F8

ОУОУ ГРУППОВОЕ.

#define UEM_MCb UEM_F9

Команда управления групповая.

#define UEM_MCBCRTb UEM_F10

Команда управления групповая со словом данных, передаваемым от КШ к ОУ.

#define UEM_NDATA_MIN 0

Число слов данных - минимальное значение.

#define UEM_NDATA_MAX 62

Число слова данных - максимальное значение.

Примечание: Аппаратно УЭМ может формировать командные и ответные сегменты, содержащие до 62 слов данных.

#define UEM_NDATA_BY_CW 63

Число слов данных определяется командным словом.

Используется в `uem_response_create()`.

#define UEM_ERRF_ERROR 1

Наличие любой ошибки (суммарный флаг).

#define UEM_ERRF_ENCODING (1<<1)

Наличие ошибки кодирования слов (суммарный флаг).

#define UEM_ERRF_FORMAT (1<<2)

Наличие нарушения формата, ошибки состава сообщения (суммарный флаг).

#define UEM_ERRF_MINGAP (1<<3)

Временной интервал меньше допустимого.

#define UEM_ERRF_NO_RESPONSE (1<<4)

Отсутствие ответа.

#define UEM_ERRF_SYNC_TYPE (1<<5)

Неверный тип синхроимпульса.

#define UEM_ERRF_MISSING_CWSW (1<<6)

Отсутствует командное или ответное слово.

#define UEM_ERRF_EXTRA_CWSW (1<<7)

Лишнее командное или ответное слово.

#define UEM_ERRF_MISSING_DW (1<<8)

Недостаточно слов данных.

#define UEM_ERRF_EXTRA_DW (1<<9)

Лишние слова данных.

#define UEM_ERRF_INCORRECT_RTN (1<<10)

Некорректный адрес ОУ.

#define UEM_ERRF_RTRT_FORMAT (1<<11)

Ошибка формата ОУОУ (одинаковые адреса ОУ, несовпадение числа СД).

#define UEM_ERRF_INC_MODE_CODE (1<<12)

Некорректная команда управления.

#define UEM_ERRF_FORMAT_MC (1<<13)

Ошибка формата команды управления.

#define UEM_ERRF_GAPN (1<<20)

Недостоверная информация (сигнал) во время паузы перед словом.

#define UEM_ERRF_PARITY (1<<21)

Ошибка четности.

#define UEM_ERRF_LESS_BITS (1<<22)

Укороченное слово.

#define UEM_ERRF_MORE_BITS (1<<23)

Удлиненное слово.

#define UEM_ERRF_ENC (1<<24)

Ошибка бифазного кодирования.

#define UEM_ERRF_DT (1<<30)

Несоблюдение минимальной паузы перед словом, по данным аппаратного декодера.

#define UEM_ERRF_ENCODING2 (1<<31)

Наличие любой ошибки кодирования слов (суммарный флаг, по данным аппаратного декодера).

Типы

typedef ViUInt32 UEM_ERROR_FLAGS

Признаки ошибок распознавания сообщения в мониторе шины.

Перечисления

enum UEM_FORMAT

Форматы сообщений (номера по ГОСТ [1]).

Элементы перечислений:

UEM_UNF Сообщение с нарушенным форматом.

UEM_F1 Формат 1.

UEM_F2 Формат 2.

UEM_F3 Формат 3.

UEM_F4 Формат 4.

UEM_F5 Формат 5.

UEM_F6 Формат 6.

UEM_F7 Формат 7.

UEM_F8 Формат 8.

UEM_F9 Формат 9.

UEM_F10 Формат 10.

enum UEM_CHANNEL

Селектор шины (А/Б).

Элементы перечислений:

UEM_CH_A Шина А.

UEM_CH_B Шина Б.

enum UEM_SYNC

Селектор синхроимпульса.

Элементы перечислений:

UEM_SYNC_D Синхроимпульс слова данных.

UEM_SYNC_C Синхроимпульс командного и ответного слова.

Функции КШ

Описания функций контроллера шины.

Группы

- **Заполнение образа командного сегмента в ОЗУ ПЭВМ**
Функции заполнения образа командного сегмента в ОЗУ ПЭВМ.
 - **Внесение ошибок состава сообщения**
Порядок действий по внесению ошибок состава сообщения.
 - **Создание и настройка командных сегментов**
Функции создания и настройки командных сегментов.
 - **Создание и настройка кадров и программы КШ**
Функции создания и настройки кадров и программ КШ.
 - **Запуск и остановка КШ**
Порядок запуска, остановки и контроля активности КШ.
 - **Передача сообщений**
Описание функции передачи отдельного сообщения.
-

Подробное описание

Описания функций контроллера шины.

Для использования этих функций необходимо получить дескриптор виртуального КШ при помощи функции **uem_bc_init()**.

Программирование КШ заключается в создании в ОЗУ КШ и настройке специальных объектов:

- командных сегментов (см. **Командные и ответные сегменты**),
- кадров - последовательностей командных сегментов,
- программ КШ - последовательностей кадров.

Командные сегменты создаются в 2 этапа:

1. Формируется образ сегмента в ОЗУ управляющей ПЭВМ в соответствии с форматом сообщения МКПД.
2. На основе этого образа создается командный сегмент в ОЗУ КШ.

После создания в командные сегменты могут быть внесены ошибки различного рода.

Кадры и программа КШ определяют состав и условия повторения последовательностей сообщений.

В заключение конфигурирования нужно указать КШ, какая именно программа КШ является исполняемой (даже если она одна).

После этого КШ может быть запущен в работу.

Заполнение образа командного сегмента в ОЗУ ПЭВМ

Функции заполнения образа командного сегмента в ОЗУ ПЭВМ.

Функции

- ViStatus **uem_bc_cseg_format** (UEM_DEVHANDLE bc, UEM_CMD_SEG *cseg_data, UEM_CHANNEL ch, UEM_FORMAT format, UEM_WORD rt, UEM_WORD sa, UEM_WORD ndatawords, UEM_WORD *datawords)
Формирование образов командных сегментов для сообщений форматов 1,2,7 и неформатных сообщений.
- ViStatus **uem_bc_cseg_format_RTRT** (UEM_DEVHANDLE bc, UEM_CMD_SEG *cseg_data, UEM_CHANNEL ch, UEM_FORMAT format, UEM_WORD rtrrx, UEM_WORD sarx, UEM_WORD rrtx, UEM_WORD satx, UEM_WORD ndatawords)
Формирование образов командных сегментов для сообщений форматов 3,8.

- ViStatus **uem_bc_cseg_format_MODE** (**UEM_DEVHANDLE** bc, **UEM_CMD_SEG** *cseg_data, **UEM_CHANNEL** ch, **UEM_FORMAT** format, **UEM_WORD** rt, **UEM_WORD** mode, **UEM_WORD** modecode, **UEM_WORD** dataword)

Формирование образов командных сегментов для сообщений форматов 4,5,6,9,10.

Подробное описание

Функции заполнения образа командного сегмента в ОЗУ ПЭВМ.

Функции **uem_bc_cseg_format()**, **uem_bc_cseg_format_RTRT()**, **uem_bc_cseg_format_MODE()** заполняют командный сегмент **cseg_data** в соответствии с заданным форматом сообщения.

- Функция **uem_bc_cseg_format()** предназначена для формирования образов сегментов для сообщений форматов 1,2,7 и неформатных.
- Функция **uem_bc_cseg_format_RTRT()** предназначена для формирования образов сегментов для сообщений форматов 3 и 8.
- Функция **uem_bc_cseg_format_MODE()** предназначена для формирования образов сегментов для сообщений форматов 4,5,6,9,10.

Функции различаются набором аргументов в соответствии с форматами сообщений. На работу функций влияют значения параметров конфигурации **UEM_MC_DIS**, **UEM_BRCST_DIS**.

Функции

ViStatus **uem_bc_cseg_format** (**UEM_DEVHANDLE** bc, **UEM_CMD_SEG** * cseg_data, **UEM_CHANNEL** ch, **UEM_FORMAT** format, **UEM_WORD** rt, **UEM_WORD** sa, **UEM_WORD** ndatawords, **UEM_WORD** * datawords)

Формирование образов командных сегментов для сообщений форматов 1,2,7 и неформатных сообщений.

Аргументы:

in	bc	Дескриптор КШ.
out	cseg_data	Образ командного сегмента в ОЗУ ПЭВМ.
in	ch	Выбор шины (А/Б).
in	format	Формат сообщения.
in	rt	Адрес ОУ.
in	sa	Подадрес.
in	ndatawords	Число слов данных.
in	datawords	Массив со словами данных. Должен быть задан в сообщениях форматов 1, 7, в неформатных сообщениях (UEM_UNF), в остальных случаях игнорируется.

Возвращает:

Код завершения. См. Коды завершения.

В неформатных сообщениях (format == **UEM_UNF**) аргументы **rt** и **sa** игнорируются, используются аргументы **ndatawords** и **datawords**.

ViStatus uem_bc_cseg_format_RTRT (UEM_DEVHANDLE *bc*, UEM_CMD_SEG * *cseg_data*, UEM_CHANNEL *ch*, UEM_FORMAT *format*, UEM_WORD *rtrx*, UEM_WORD *sarx*, UEM_WORD *rttx*, UEM_WORD *satx*, UEM_WORD *ndatawords*)

Формирование образов командных сегментов для сообщений форматов 3,8.

Аргументы:

in	<i>bc</i>	Дескриптор КШ.
out	<i>cseg_data</i>	Образ командного сегмента в ОЗУ ПЭВМ.
in	<i>ch</i>	Выбор шины (А/Б).
in	<i>format</i>	Формат сообщения.
in	<i>rtrx</i>	Адрес принимающего ОУ.
in	<i>sarx</i>	Подадрес принимающего ОУ.
in	<i>rttx</i>	Адрес передающего ОУ.
in	<i>satx</i>	Подадрес передающего ОУ.
in	<i>ndatawords</i>	Число слов данных.

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_bc_cseg_format_MODE (UEM_DEVHANDLE *bc*, UEM_CMD_SEG * *cseg_data*, UEM_CHANNEL *ch*, UEM_FORMAT *format*, UEM_WORD *rt*, UEM_WORD *mode*, UEM_WORD *modecode*, UEM_WORD *dataword*)

Формирование образов командных сегментов для сообщений форматов 4,5,6,9,10.

Аргументы:

in	<i>bc</i>	Дескриптор КШ.
out	<i>cseg_data</i>	Образ командного сегмента в ОЗУ ПЭВМ.
in	<i>ch</i>	Выбор шины (А/Б).
in	<i>format</i>	Формат сообщения. В дополнение к перечисленным форматам в этом аргументе допускается указывать значение UEM_UNF . Это служит указанием функции подобрать нужный формат автоматически. Для указания группового формата в этом случае следует задать аргумент rt = 31 .
in	<i>rt</i>	Адрес ОУ.
in	<i>mode</i>	Режим управления в командах управления.
in	<i>modecode</i>	Код команды управления.
in	<i>dataword</i>	Слово данных в команде управления. Игнорируется, если КУ не предполагает передачу СД в командном сегменте.

Возвращает:

Код завершения. См. Коды завершения.

Внесение ошибок состава сообщения

Порядок действий по внесению ошибок состава сообщения.

Под ошибками состава сообщения понимается несоответствие числа слов данных формату сообщения и командному слову, в том числе: неверное число слов данных, наличие слов данных в сегментах, в которых их быть не должно, отсутствие слов данных в сегментах, в которых они должны быть, а также

- несоответствие друг другу командных слов в сообщениях ОУОУ (форматов 3, 8), групповые адреса в командных словах, в которых они недопустимы, и некоторые другие.

Для внесения ошибок состава сообщения в командный сегмент КШ следует:

1. Заполнить образ командного сегмента корректным командным сегментом при помощи функции **uem_bc_cseg_format()**, **uem_bc_cseg_format_RTRT()** или **uem_bc_cseg_format_MODE()**.
2. Изменить этот образ программно.

Альтернативно, приложение может заполнить образ командного сегмента самостоятельно с самого начала.

Создание и настройка командных сегментов

Функции создания и настройки командных сегментов.

Перечисления

- enum **UEM_CSEG_TYPE** { **UEM_CSEG_NORMAL**, **UEM_CSEG_OVERLAY**, **UEM_CSEG_GAP** }
Тип командного сегмента.

Функции

- ViStatus **uem_bc_cseg_create** (**UEM_DEVHANDLE** bc, **UEM_OBJHANDLE** *cseg, **UEM_CMD_SEG** *cseg_data)
Создание командного сегмента.
- ViStatus **uem_cseg_read** (**UEM_OBJHANDLE** cseg, **UEM_CMD_SEG** *cseg_data)
Чтение командного сегмента.
- ViStatus **uem_cseg_gap_set** (**UEM_OBJHANDLE** cseg, **UEM_WORD** gap, **UEM_WORD** gap_flags, **UEM_WORD** gap_timeout)
Программирование паузы перед сообщением.
- ViStatus **uem_cseg_gap_get** (**UEM_OBJHANDLE** cseg, **UEM_WORD** *gap, **UEM_WORD** *gap_flags, **UEM_WORD** *gap_timeout)
Считывание паузы перед сообщением.
- ViStatus **uem_cseg_gap_reset** (**UEM_OBJHANDLE** cseg)
Сброс паузы перед сообщением.
- ViStatus **uem_cseg_word_gap_set** (**UEM_OBJHANDLE** cseg, **UEM_WORD** wordnumber, **UEM_WORD** gap, **UEM_WORD** gap_flags, **UEM_WORD** gap_timeout)
Программирование паузы между словами.
- ViStatus **uem_cseg_word_gap_get** (**UEM_OBJHANDLE** cseg, **UEM_WORD** wordnumber, **UEM_WORD** *gap, **UEM_WORD** *gap_flags, **UEM_WORD** *gap_timeout)
Считывание паузы перед словом.
- ViStatus **uem_cseg_error_set** (**UEM_OBJHANDLE** cseg, **UEM_WORD** wordnumber, **UEM_ERROR_TYPE** error_type, ViInt32 error_pos, ViInt32 error_param)
Внесение ошибок кодирования.
- ViStatus **uem_cseg_error_get** (**UEM_OBJHANDLE** cseg, **UEM_WORD** wordnumber, **UEM_ERROR_TYPE** *error_type, ViInt32 *error_pos, ViInt32 *error_param)
Считывание внесенных ошибок кодирования.
- ViStatus **uem_cseg_sync_set** (**UEM_OBJHANDLE** cseg, **UEM_WORD** wordnumber, **UEM_SYNC** sync)
Установка типа синхроимпульса.
- ViStatus **uem_cseg_sync_get** (**UEM_OBJHANDLE** cseg, **UEM_WORD** wordnumber, **UEM_SYNC** *sync)
Считывание типа синхроимпульса.

- ViStatus **uem_bc_gap_create** (UEM_DEVHANDLE bc, UEM_OBJHANDLE *cseg, UEM_WORD gap, UEM_WORD gap_flags, UEM_WORD gap_timeout, UEM_CHANNEL ch)
Создание паузы.
- ViStatus **uem_bc_cseg_overlay** (UEM_DEVHANDLE bc, UEM_OBJHANDLE *cseg_o, UEM_OBJHANDLE cseg_1, UEM_WORD gap, UEM_OBJHANDLE cseg_2)
Создание сообщения с наложением.
- ViStatus **uem_cseg_type** (UEM_OBJHANDLE cseg, UEM_CSEG_TYPE *type)
Запрос типа командного сегмента.
- ViStatus **uem_cseg_desrtoy** (UEM_OBJHANDLE cseg)
Уничтожение командного сегмента.

Флаги отсчета паузы

Пауза отсчитывается от одного из определенных событий.

Флаги отсчета паузы определяют выбор события.

Пауза может отсчитываться от начала либо конца предшествующего сегмента, переданного КШ либо другим абонентом (ОУ), по той же либо по альтернативной шине. Выбор каждой из альтернатив управляется отдельным флагом. Для начала отсчета паузы также может использоваться сигнал внешней синхронизации **sync_in_1**. Поступление этого сигнала используется *вместо* события начала или конца сегмента, переданного КШ. Если же задан отсчет паузы от начала (или конца) сегмента, переданного другим абонентом, то учитываются оба события - начало (или конец) сегмента и поступление сигнала. При задании отсчета паузы от начала (или конца) сегмента другого абонента, или от поступления сигнала, или обоих, то есть - от внешних событий, которые теоретически могут и не наступить, используется таймаут, который задается в функции **uem_cseg_gap_set()**, и по истечении которого отсчет паузы начинается без дальнейшего ожидания внешних событий.

- #define **UEM_CSEG_GAP_FROM_START** 0x20
Отсчет паузы от начала предыдущего сегмента.
- #define **UEM_CSEG_GAP_FROM_END** 0
Отсчет паузы от конца предыдущего сегмента.
- #define **UEM_CSEG_GAP_ESYNC** 0x10
Отсчет паузы после сигнала внешней синхронизации (только для КШ).
- #define **UEM_CSEG_GAP_ALT_BUS** 0x08
Отсчет паузы от сегмента по альтернативной шине (только для КШ).
- #define **UEM_CSEG_GAP_THIS_BUS** 0
Отсчет паузы от сегмента по этой же шине (только для КШ).
- #define **UEM_CSEG_GAP_ALT_AB** 0x04
Отсчет паузы от сегмента другого абонента (ОУ) (только для КШ).
- #define **UEM_CSEG_GAP_THIS_AB** 0
Отсчет паузы от собственного сегмента (только для КШ).
- #define **UEM_CSEG_GAP_DEFAULT_FLAGS** (UEM_CSEG_GAP_FROM_START)
Стандартный набор флагов отсчета паузы.

Диапазоны значений параметров отсчета паузы

- #define **UEM_CSEG_GAP_MIN** 0
Минимальное значение паузы.
- #define **UEM_CSEG_GAP_MAX** 65535
Максимальное значение паузы.

- **#define UEM_CSEG_GAP_DEFAULT_VALUE 0**
Значение паузы по умолчанию.
 - **#define UEM_CSEG_GAP_TIMEOUT_MIN 0**
Минимальное значение таймаута отсчета паузы.
 - **#define UEM_CSEG_GAP_TIMEOUT_MAX 1023**
Максимальное значение таймаута отсчета паузы.
 - **#define UEM_CSEG_GAP_DEFAULT_TIMEOUT 0**
Таймаут отсчета паузы по умолчанию.
-

Подробное описание

Функции создания и настройки командных сегментов.

В данном разделе описаны функции создания командных сегментов и задания их дополнительных параметров, таких как паузы и ошибки кодирования.

Макросы

#define UEM_CSEG_GAP_FROM_START 0x20

Отсчет паузы от начала предыдущего сегмента.

Пауза может отсчитываться от начала или окончания переданного ранее сегмента слов. Если флаг установлен (**UEM_CSEG_GAP_FROM_START=1**), то отсчет паузы будет вестись от начала передачи этого сегмента слов, если не установлен – от окончания переданного ранее сегмента слов. Для ОУ флаг имеет смысл только для описания СД, так как для ОС (первого слова ответного сегмента) пауза всегда отсчитывается от окончания приема командного сегмента (либо ответного сегмента передающего ОУ для принимающего ОУ в сообщениях формата ОУОУ).

#define UEM_CSEG_GAP_FROM_END 0

Отсчет паузы от конца предыдущего сегмента.

Данный макрос введен, чтобы явно обозначить, что флаг **UEM_CSEG_GAP_FROM_START** сброшен. Макрос не должен указываться одновременно с **UEM_CSEG_GAP_FROM_START**.

#define UEM_CSEG_GAP_ESYNC 0x10

Отсчет паузы после сигнала внешней синхронизации (только для КШ).

Флаг **UEM_CSEG_GAP_ESYNC** определяет, задается ли отсчет паузы внешним синхросигналом или моментом, связанным с передачей или приемом предыдущего сегмента, в соответствии с состояниями флагов **UEM_CSEG_GAP_THIS_BUS / UEM_CSEG_GAP_ALT_BUS**, **UEM_CSEG_GAP_THIS_AB / UEM_CSEG_GAP_ALT_AB**, **UEM_CSEG_GAP_FROM_START / UEM_CSEG_GAP_FROM_END**. Единичное значение (**UEM_CSEG_GAP_ESYNC=1**) означает, что отсчет паузы начнется при поступлении внешнего синхросигнала. Нулевое значение – от момента, определяемого предыдущим сегментом. Если одновременно установлены флаги **UEM_CSEG_GAP_ESYNC** и **UEM_CSEG_GAP_ALT_AB**, то для начала отсчета паузы должны произойти оба события. Синхросигнал должен поступать на вход внутренней синхронизации 1. Дополнительно в УЭМ имеются средства внутренней генерации синхросигнала 1.

См. подробнее в разделе **Управление синхронизацией**.

#define UEM_CSEG_GAP_ALT_BUS 0x08

Отсчет паузы от сегмента по альтернативной шине (только для КШ).

Флаг **UEM_CSEG_GAP_ALT_BUS** задает шину, событие на которой определяет момент начала отсчета паузы. Единичное значение флага (**UEM_CSEG_GAP_ALT_BUS=1**) означает, что отсчет паузы будет вестись от события в резервной шине относительно шины, заданной текущим значением поля **ch** в **UEM_CMD_SEG**. Значения полей данной структуры учитываются при создании командного сегмента функцией **uem_bc_cseg_create()**. Значение поля **ch** может быть задано как вручную, так и с использованием функций **uem_bc_cseg_format()**, **uem_bc_cseg_format_RTRT()**, **uem_bc_cseg_format_MODE()**. При нулевом значении данного флага отсчет паузы начинается от события в той же шине. Для ОУ данный флаг незначим.

#define UEM_CSEG_GAP_THIS_BUS 0

Отсчет паузы от сегмента по этой же шине (только для КШ).

Данный макрос введен, чтобы явно обозначить, что флаг **UEM_CSEG_GAP_ALT_BUS** сброшен. Макрос не должен указываться одновременно с **UEM_CSEG_GAP_ALT_BUS**.

#define UEM_CSEG_GAP_ALT_AB 0x04

Отсчет паузы от сегмента другого абонента (ОУ) (только для КШ).

Флаг **UEM_CSEG_GAP_ALT_AB** задает отсчет паузы от начала или конца сегмента, переданного другим абонентом (ОУ), при единичном значении. При нулевом значении отсчет паузы выполняется от начала или конца сегмента, переданного самим КШ.

#define UEM_CSEG_GAP_THIS_AB 0

Отсчет паузы от собственного сегмента (только для КШ).

Данный макрос введен, чтобы явно обозначить, что флаг **UEM_CSEG_GAP_ALT_AB** сброшен. Макрос не должен указываться одновременно с **UEM_CSEG_GAP_ALT_AB**.

#define UEM_CSEG_GAP_DEFAULT_FLAGS (UEM_CSEG_GAP_FROM_START)

Стандартный набор флагов отсчета паузы.

#define UEM_CSEG_GAP_MIN 0

Минимальное значение паузы.

#define UEM_CSEG_GAP_MAX 65535

Максимальное значение паузы.

#define UEM_CSEG_GAP_DEFAULT_VALUE 0

Значение паузы по умолчанию.

#define UEM_CSEG_GAP_TIMEOUT_MIN 0

Минимальное значение таймаута отсчета паузы.

#define UEM_CSEG_GAP_TIMEOUT_MAX 1023

Максимальное значение таймаута отсчета паузы.

#define UEM_CSEG_GAP_DEFAULT_TIMEOUT 0

Таймаут отсчета паузы по умолчанию.

Перечисления

enum UEM_CSEG_TYPE

Тип командного сегмента.

Элементы перечислений:

UEM_CSEG_NORMAL Нормальный сегмент.

UEM_CSEG_OVERLAY Сегмент с наложением.

UEM_CSEG_GAP Пауза.

Функции

ViStatus uem_bc_cseg_create (UEM_DEVHANDLE *bc*, UEM_OBJHANDLE * *cseg*, UEM_CMD_SEG * *cseg_data*)

Создание командного сегмента в ОЗУ виртуального КШ.

Аргументы:

in	<i>bc</i>	Дескриптор виртуального КШ.
out	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
in	<i>cseg_data</i>	Образ командного сегмента в ОЗУ ПЭВМ.

Возвращает:

Код завершения. См. **Коды завершения**.

Командный сегмент создается с неустановленной (нулевой) паузой и без внесенных ошибок.

ViStatus uem_cseg_read (UEM_OBJHANDLE *cseg*, UEM_CMD_SEG * *cseg_data*)

Чтение командного сегмента.

Считывание командного сегмента из ОЗУ КШ обратно в ОЗУ ЭВМ.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
out	<i>cseg_data</i>	Образ командного сегмента в ОЗУ ПЭВМ.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_gap_set (UEM_OBJHANDLE cseg, UEM_WORD gap, UEM_WORD gap_flags, UEM_WORD gap_timeout)

Программирование паузы перед сообщением.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
in	<i>gap</i>	Значение паузы, в единицах по 0,25 мкс, 0-65535 (См. Диапазоны значений).
in	<i>gap_flags</i>	Флаги, определяющие способ отсчета паузы, комбинация бит UEM_CSEG_GAP_XXXX (Флаги отсчета паузы).
in	<i>gap_timeout</i>	Таймаут отсчета паузы (когда отсчет зависит от внешних событий), в мкс, 0-1023 (См. Диапазоны значений).

Возвращает:

Код завершения. См. **Коды завершения**.

Для сообщений, для которых не задана явно пауза при помощи данной функции, автоматически будет рассчитана и установлена стандартная пауза в функции **uem_bcp_append_cseg()** или **uem_bcp_install()**.

ViStatus uem_cseg_gap_get (UEM_OBJHANDLE cseg, UEM_WORD * gap, UEM_WORD * gap_flags, UEM_WORD * gap_timeout)

Считывание паузы перед сообщением.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
out	<i>gap</i>	Значение паузы, в единицах по 0,25 мкс, 0-65535.
out	<i>gap_flags</i>	Флаги, определяющие способ отсчета паузы. Комбинация бит UEM_CSEG_GAP_XXXX .
out	<i>gap_timeout</i>	Таймаут отсчета паузы (когда отсчет зависит от внешних событий), в мкс, 0-1023.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_gap_reset (UEM_OBJHANDLE cseg)

Сброс паузы перед сообщением.

Пауза перед сообщением сбрасывается в стандартное (нулевое) значение и помечается как неустановленная (как если бы функция **uim_cseg_gap_set()** к данному сообщению не применялась). Эта отметка влияет на автоматический расчет паузы в функции **uem_bcp_append_cseg()**, **uem_bcp_install()**.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
----	-------------	--

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_word_gap_set (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_WORD gap, UEM_WORD gap_flags, UEM_WORD gap_timeout)

Программирование паузы между словами.

Функция устанавливает паузу перед указанным словом командного сегмента. Функция полностью аналогична функции **uem_cseg_gap_set()**, но позволяет установить паузу не перед первым, а перед любым словом командного сегмента. Единственная причина использовать данную функцию - внесение ошибки типа "разрыв сегмента".

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КИШ.
in	<i>wordnumber</i>	Номер слова, перед которым устанавливается пауза. Нумерация с 0, сквозная, сначала все КС, потом все СД.
in	<i>gap</i>	Значение паузы, в единицах по 0,25 мкс, 0-65535. (См. Диапазоны значений).
in	<i>gap_flags</i>	Флаги, определяющие способ отсчета паузы, комбинация бит Флаги отсчета паузы .
in	<i>gap_timeout</i>	Таймаут отсчета паузы (когда отсчет зависит от внешних событий), в мкс, 0-1023. (См. Диапазоны значений).

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_word_gap_get (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_WORD * gap, UEM_WORD * gap_flags, UEM_WORD * gap_timeout)

Считывание паузы перед словом.

Считывает параметры паузы, установленные функцией **uem_cseg_word_gap_set()**.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КИШ.
in	<i>wordnumber</i>	Номер слова. Нумерация с 0, сквозная, сначала все КС, потом все СД.
out	<i>gap</i>	Значение паузы, в единицах по 0,25 мкс, 0-65535.
out	<i>gap_flags</i>	Флаги, определяющие способ отсчета паузы, комбинация бит Флаги отсчета паузы .
out	<i>gap_timeout</i>	Таймаут отсчета паузы (когда отсчет зависит от внешних событий), в мкс, 0-1023.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_error_set (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_ERROR_TYPE error_type, Vilnt32 error_pos, Vilnt32 error_param)

Внесение ошибок кодирования.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КИШ.
in	<i>wordnumber</i>	Номер слова, в которое вносится ошибка. Нумерация с 0, сквозная, сначала все КС, потом все СД.
in	<i>error_type</i>	Тип вносимой ошибки. См. Типы вносимых ошибок кодирования .
in	<i>error_pos</i>	Позиция ошибки. Интерпретируется в зависимости от типа ошибки.

in	<i>error_param</i>	Дополнительный параметр. Интерпретируется в зависимости от типа ошибки.
----	--------------------	---

Возвращает:

Код завершения. См. **Коды завершения**.

Допускается внести ошибки кодирования в несколько слов командного сегмента.

ViStatus uem_cseg_error_get (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_ERROR_TYPE * error_type, Vilnt32 * error_pos, Vilnt32 * error_param)

Считывание внесенных ошибок кодирования.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
in	<i>wordnumber</i>	Номер слова, в которое вносится ошибка. Нумерация с 0, сквозная, сначала все КС, потом все СД.
out	<i>error_type</i>	Тип вносимой ошибки. См. Типы вносимых ошибок кодирования .
out	<i>error_pos</i>	Позиция ошибки. Интерпретируется в зависимости от типа ошибки.
out	<i>error_param</i>	Дополнительный параметр. Тип вносимой ошибки. Интерпретируется в зависимости от типа ошибки.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_sync_set (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_SYNC sync)

Установка типа синхроимпульса.

При создании сегмента тип синхроимпульса для каждого слова уже установлен корректно. Программная установка типа синхроимпульса может использоваться для следующих целей:

- назначение командных слов в неформатном сообщении (**UEM_UNF**),
- внесение ошибок типа "неверный синхроимпульс",
- в сочетании с внесением ошибок кодирования **UEM_ERRT_BAD_SYNCHRO**, расширяет номенклатуру искаженных форм синхроимпульса.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
in	<i>wordnumber</i>	Номер слова, в котором устанавливается синхроимпульс. Нумерация с 0, сквозная, сначала все КС, потом все СД.
in	<i>sync</i>	Тип синхроимпульса.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_sync_get (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_SYNC * sync)

Считывание типа синхроимпульса.

Функция читает тип синхроимпульса в слове сегмента, установленный при создании сегмента или функцией **uem_cseg_sync_set()**.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
in	<i>wordnumber</i>	Номер слова. Нумерация с 0, сквозная, сначала все КС, потом все СД.
out	<i>sync</i>	Тип синхроимпульса.

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_bc_gap_create (UEM_DEVHANDLE bc, UEM_OBJHANDLE * cseg, UEM_WORD gap, UEM_WORD gap_flags, UEM_WORD gap_timeout, UEM_CHANNEL ch)

Создание паузы.

Данная функция создает командный сегмент особого вида, не содержащий передаваемых слов, а задающий только паузу в передаче командных сегментов. Аргументы функции аналогичны аргументам функции **uem_cseg_gap_set()**. Их можно прочитать обратно в ОЗУ управляющей ПЭВМ функцией **uem_cseg_gap_get()**. Командный сегмент "пауза" нельзя прочитать функцией **uem_cseg_read()**.

Аргументы:

in	<i>bc</i>	Дескриптор виртуального КШ.
out	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
in	<i>gap</i>	Значение паузы, в единицах по 0,25 мкс, 0-65535. (См. Диапазоны значений).
in	<i>gap_flags</i>	Флаги, определяющие способ отсчета паузы, комбинация бит UEM_CSEG_GAP_XXXX .
in	<i>gap_timeout</i>	Таймаут отсчета паузы (когда отсчет зависит от внешних событий), в мкс, 0-1023. (См. Диапазоны значений).
in	<i>ch</i>	Выбор шины (А/Б). Привязка к шине влияет на интерпретацию флагов в параметре <i>gap_flags</i> , а также на интерпретацию флагов <i>gap_flags</i> в следующем сообщении в кадре. В случаях, когда это не важно или выбор неочевиден, рекомендуется указывать UEM_CH_A .

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_bc_cseg_overlay (UEM_DEVHANDLE bc, UEM_OBJHANDLE * cseg_o, UEM_OBJHANDLE cseg_1, UEM_WORD gap, UEM_OBJHANDLE cseg_2)

Создание сообщения с наложением.

Сообщение с наложением - это ситуация, когда два сообщения передаются в шины А и Б одновременно или почти одновременно.

Такие ситуации необходимы для проверки функции вытеснения сообщений в ОУ.

В УЭМ для выполнения такой передачи создается комбинированный командный сегмент, состоящий из двух командных сегментов (по шинам А и Б).

Данная функция создает сегмент с наложением из двух исходных командных сегментов.

Исходные командные сегменты должны быть привязаны к разным шинам.

Создание командного сегмента с наложением обладает следующими особенностями:

Исходные командные сегменты никак не изменяются.

Внесенные ошибки кодирования и смены синхроимпульса переносятся из исходных командных сегментов в новый без изменений.

Паузы, заданные для исходных командных сегментов, игнорируются; в новом командном сегменте паузы устанавливаются в значения по умолчанию.

После создания командного сегмента с наложением исходные командные сегменты можно использовать независимо от нового, или - если они больше не нужны - уничтожить.

Ошибки кодирования и синхроимпульса можно вносить как до построения нового сегмента, так и после - в новый сегмент. Но удобнее делать это до, так как соответствие слов исходных и нового сегмента рассчитывается по специальному алгоритму и не очевидно.

Командный сегмент с наложением нельзя прочитать функцией **uem_cseg_read()**.

Аргументы:

in	<i>bc</i>	Дескриптор виртуального КШ.
out	<i>cseg_o</i>	Дескриптор объекта нового командного сегмента в ОЗУ КШ.
in	<i>cseg_1</i>	Дескриптор 1-го командного сегмента.
in	<i>gap</i>	Пауза между началом передачи 1-го командного сегмента и началом передачи 2-го командного сегмента. В единицах по 0,25 мкс.
in	<i>cseg_2</i>	Дескриптор 2-го командного сегмента.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_type (UEM_OBJHANDLE cseg, UEM_CSEG_TYPE * type)

Запрос типа командного сегмента.

При помощи этой функции можно отличить нормальный командный сегмент от специального сегмента - паузы или сегмента с наложением.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента.
out	<i>type</i>	Тип командного сегмента.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_cseg_desrtoy (UEM_OBJHANDLE cseg)

Уничтожение командного сегмента.

Аргументы:

in	<i>cseg</i>	Дескриптор объекта командного сегмента в ОЗУ КШ.
----	-------------	--

Возвращает:

Код завершения. См. **Коды завершения**.

Создание и настройка кадров и программы КШ

Функции создания и настройки кадров и программ КШ.

Функции

- ViStatus **uem_bcp_create** (UEM_OBJHANDLE *bcprog, UEM_DWORD max_size, UEM_DEVHANDLE bc)
Создание программы КШ.
- ViStatus **uem_bcp_append_frame** (UEM_OBJHANDLE bcprog, UEM_WORD repeat_count, UEM_WORD frame_flags, int *frameindex)
Добавление кадра в конец программы КШ.
- ViStatus **uem_bcp_append_cseg** (UEM_OBJHANDLE bcprog, UEM_OBJHANDLE cseg, int *csegindex)
Добавление командного сегмента в конец кадра.
- ViStatus **uem_bcp_discover_cseg** (UEM_OBJHANDLE bcprog, int frameindex, int csegindex, UEM_OBJHANDLE *cseg)
Выяснение командного сегмента.
- ViStatus **uem_bcp_replace_cseg** (UEM_OBJHANDLE bcprog, int frameindex, int csegindex, UEM_OBJHANDLE cseg)
Замена командного сегмента в кадре.
- ViStatus **uem_bcp_dimension** (UEM_OBJHANDLE bcprog, int frameindex, int *dim)
Запрос размерностей программы КШ.
- ViStatus **uem_bcp_inspect_frame** (UEM_OBJHANDLE bcprog, int frameindex, UEM_WORD *repeat_count, UEM_WORD *frame_flags)
Запрос характеристик кадра.
- ViStatus **uem_bcp_install** (UEM_OBJHANDLE bcprog)
Установка программы КШ в качестве исполняемой.
- ViStatus **uem_bcp_desrtoy** (UEM_OBJHANDLE bcprog)
Уничтожение объекта "программа КШ" в ОЗУ КШ.
- ViStatus **uem_bcp_set_standard_gaps** (UEM_OBJHANDLE bcprog)
Расчет и установка стандартных пауз между сообщениями (необязательно).

Константы для числа повторов кадра

Данные константы могут использоваться в аргументе **repeat_count** функции **uem_bcp_append_frame()**.

- #define **UEM_FRAME_REPEAT_UNLIM** 0
Неограниченное число повторов кадра.
- #define **UEM_FRAME_REPEAT_MIN** 1
Минимальное число повторов кадра.
- #define **UEM_FRAME_REPEAT_MAX** 1023
Максимальное число повторов кадра.
- #define **UEM_FRAME_REPEAT_DEFAULT** (UEM_FRAME_REPEAT_MIN)
Число повторов кадра по умолчанию (1).

Флаги кадра

Данные константы могут использоваться в аргументе **frame_flags** функции **uem_bcp_append_frame()**.

- #define **UEM_FRAME_STOP** 0x0001
Остановка КШ.

- **#define UEM_FRAME_ALLRPT 0x0002**
Защивание программы КШ.
- **#define UEM_FRAME_NONE 0**
Нет указаний.
- **#define UEM_FRAME_CONT (UEM_FRAME_NONE)**
Продолжение программы КШ.
- **#define UEM_FRAME_DEFAULT (UEM_FRAME_STOP)**
Флаги кадра по умолчанию: остановка КШ.

Виды размерностей программы КШ

- **#define UEM_VCP_NFRAMES (-1)**
Число кадров.
- **#define UEM_VCP_CUR_SIZE (-2)**
Текущий размер.
- **#define UEM_VCP_MAX_SIZE (-3)**
Максимальный размер.

Подробное описание

Функции создания и настройки кадров и программ КШ.

Программа КШ (vcp) состоит из кадров, а кадры - из командных сегментов (cseg).

Программа определяет состав и порядок передаваемых сообщений, в том числе - повторы последовательностей сообщений.

В данном разделе описаны функции создания и настройки кадров и программ КШ.

Макросы

#define UEM_FRAME_REPEAT_UNLIM 0

Неограниченное число повторов кадра.

#define UEM_FRAME_REPEAT_MIN 1

Минимальное число повторов кадра.

#define UEM_FRAME_REPEAT_MAX 1023

Максимальное число повторов кадра.

#define UEM_FRAME_REPEAT_DEFAULT (UEM_FRAME_REPEAT_MIN)

Число повторов кадра по умолчанию (1).

#define UEM_FRAME_STOP 0x0001

Остановка КШ.

Единичное значение этого флага соответствует указанию остановки работы КШ после передачи текущего кадра (включая выполнение заданного ограниченного (не «бесконечного») количества повторений). Если задано неограниченное число повторений, то при `bc_stop = 1` при первом чтении данного описателя аппаратурой блок обработан не будет и КШ завершит работу. Таким образом, данный флаг не должен указываться, если требуется неограниченное число повторов кадра.

#define UEM_FRAME_ALLRPT 0x0002

Зацикливание программы КШ.

Единичное значение этого флага, если флаг **UEM_FRAME_STOP** установлен в «0», соответствует указанию после завершения передачи текущего кадра (включая выполнение заданного ограниченного (не «бесконечного») количества повторений), перейти к выполнению первого кадра данной программы КШ. Если флаг **UEM_FRAME_STOP** установлен в «1», будут выполнены оба действия: остановка КШ (приостановка) и переход к выполнению первого кадра программы, - такая комбинация может быть полезна для организации периодического повторения программы КШ по внешнему синхросигналу.

#define UEM_FRAME_NONE 0

Нет указаний.

#define UEM_FRAME_CONT (UEM_FRAME_NONE)

Продолжение программы КШ.

Идентификатор введен как более осмысленный синоним идентификатора **UEM_FRAME_NONE**, поскольку отсутствие указаний означает как раз продолжение программы КШ, переход к следующему кадру после выполнения данного кадра, с учетом заданных повторов.

#define UEM_FRAME_DEFAULT (UEM_FRAME_STOP)

Флаги кадра по умолчанию: остановка КШ.

#define UEM_BCP_NFRAMES (-1)

Число кадров.

#define UEM_BCP_CUR_SIZE (-2)

Текущий размер.

#define UEM_BCP_MAX_SIZE (-3)

Максимальный размер.

Функции

ViStatus uem_bcp_create (UEM_OBJHANDLE * *bcp prog*, UEM_DWORD *max_size*, UEM_DEVHANDLE *bc*)

Создание программы КШ.

Аргументы:

out	<i>bcp prog</i>	Дескриптор объекта программы КШ в ОЗУ КШ.
in	<i>max_size</i>	Максимальный размер (число вложенных кадров и сообщений).
in	<i>bc</i>	Дескриптор виртуального КШ.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bcp_append_frame (UEM_OBJHANDLE *bcp prog*, UEM_WORD *repeat_count*, UEM_WORD *frame_flags*, int * *frameindex*)

Добавление кадра в конец программы КШ.

Аргументы:

in	<i>bcp prog</i>	Дескриптор программы КШ в ОЗУ КШ.
in	<i>repeat_count</i>	Число повторов кадра. Значения: 0-1023; значение 0 (UEM_FRAME_REPEAT_UNLIM) означает неограниченное число повторов; см. также константы UEM_FRAME_REPEAT_XXXX .
in	<i>frame_flags</i>	Управляющие флаги кадра, константы UEM_FRAME_XXXX .
out	<i>frameindex</i>	Если этот указатель не равен NULL, в переменную, на которую он указывает, будет записан номер (позиция) нового кадра в программе КШ.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bcp_append_cseg (UEM_OBJHANDLE *bcp prog*, UEM_OBJHANDLE *cseg*, int * *csegindex*)

Добавление командного сегмента в конец кадра.

Добавление всегда выполняется в конец последнего кадра указанной программы КШ.

Аргументы:

in	<i>bcp prog</i>	Дескриптор программы КШ в ОЗУ КШ.
in	<i>cseg</i>	Дескриптор командного сегмента в ОЗУ КШ.
out	<i>csegindex</i>	Если этот указатель не равен NULL, в переменную, на которую он указывает, будет записан номер (позиция) сегмента в кадре.

Возвращает:

Код завершения. См. **Коды завершения**.

Если для добавляемого командного сегмента не запрограммирована пауза перед ним (при помощи **uem_cseg_gap_set()** или другим способом), при добавлении будет рассчитана и установлена стандартная пауза (за исключением командных сегментов, являющихся первыми в кадре: для них стандартная пауза будет установлена функцией **uem_bcp_install()**; см. также **uem_bcp_set_standard_gaps()**).

Допускается добавлять один командный сегмент несколько раз в один и тот же кадр или в разные кадры одной и той же программы КШ или в разные программы КШ в пределах одного виртуального КШ. Однако следует иметь в виду, что установленная или автоматически рассчитанная пауза перед сообщением будет отсчитываться перед каждым вхождением этого сообщения.

ViStatus uem_bcp_discover_cseg (UEM_OBJHANDLE *bcprog*, int *frameindex*, int *csegindex*, UEM_OBJHANDLE * *cseg*)

Выяснение командного сегмента.

Функция определяет, какой командный сегмент установлен в указанной позиции указанной программы КШ.

Аргументы:

in	<i>bcprog</i>	Дескриптор программы КШ в ОЗУ КШ.
in	<i>frameindex</i>	Номер (позиция) кадра в программе КШ, нумерация с 0.
in	<i>csegindex</i>	Номер (позиция) командного сегмента в кадре, нумерация с 0.
out	<i>cseg</i>	Дескриптор командного сегмента в ОЗУ КШ.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bcp_replace_cseg (UEM_OBJHANDLE *bcprog*, int *frameindex*, int *csegindex*, UEM_OBJHANDLE *cseg*)

Замена командного сегмента в кадре.

Аргументы:

in	<i>bcprog</i>	Дескриптор программы КШ в ОЗУ КШ.
in	<i>frameindex</i>	Номер (позиция) кадра в программе КШ, нумерация с 0.
in	<i>csegindex</i>	Номер (позиция) командного сегмента в кадре, нумерация с 0.
in	<i>cseg</i>	Дескриптор командного сегмента в ОЗУ КШ.

Возвращает:

Код завершения. См. **Коды завершения**.

Если для вставляемого командного сегмента не запрограммирована пауза перед ним (при помощи **uem_cseg_gap_set()** или другим способом), при добавлении будет рассчитана и установлена стандартная пауза (за исключением командных сегментов, являющихся первыми в кадре: для них стандартная пауза будет установлена функцией **uem_bcp_install()**; см. также **uem_bcp_set_standard_gaps()**).

Допускается вставлять один командный сегмент несколько раз в один и тот же кадр или в разные кадры одной и той же программы КШ или в разные программы КШ в пределах одного виртуального КШ. Однако следует иметь в виду, что установленная или автоматически рассчитанная пауза перед сообщением будет отсчитываться перед каждым вхождением этого сообщения.

ViStatus uem_bcp_dimension (UEM_OBJHANDLE *bcprog*, int *frameindex*, int * *dim*)

Запрос размерностей программы КШ.

Функция возвращает число командных сегментов в указанном кадре программы, а также - показатели размерности самой программы.

Аргументы:

in	<i>bcprog</i>	Дескриптор программы КШ в ОЗУ КШ.
in	<i>frameindex</i>	Значения ≥ 0 : Номер (позиция) кадра в программы КШ, нумерация с 0; значения < 0 : запрос размерностей самой программы КШ, допускается указать одну из констант UEM_BCP_XXX .
out	<i>dim</i>	В эту переменную записывается значение запрошенной размерности: при <i>frameindex</i> ≥ 0 возвращается число командных сегментов в кадре с номером <i>frameindex</i> , при <i>frameindex</i> < 0 возвращается число кадров в программе КШ, текущий размер программы или максимальный размер программы, в зависимости от значения <i>frameindex</i> (см. UEM_BCP_XXX).

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bcp_inspect_frame (UEM_OBJHANDLE *bcprog*, int *frameindex*, UEM_WORD * *repeat_count*, UEM_WORD * *frame_flags*)

Запрос характеристик кадра.

Аргументы:

in	<i>bcprog</i>	Дескриптор программы КШ в ОЗУ КШ.
in	<i>frameindex</i>	Номер (позиция) кадра в программе КШ, нумерация с 0.
out	<i>repeat_count</i>	Число повторов кадра. Значения: 0-1023; значение 0 (UEM_FRAME_REPEAT_UNLIM) означает неограниченное число повторов; см. также константы UEM_FRAME_REPEAT_XXXX .
out	<i>frame_flags</i>	Управляющие флаги кадра, константы UEM_FRAME_XXXX .

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bcp_install (UEM_OBJHANDLE *bcprog*)

Установка программы КШ в качестве исполняемой.

Установка заключается в загрузке адреса этой программы в соответствующий управляющий регистр УЭМ.

Перед установкой программы функция убеждается, что для всех командных сегментов в программе КШ запрограммированы паузы перед ними, и устанавливает стандартные паузы для тех сегментов, для которых они не запрограммированы, путем обращения к функции **uem_bcp_set_standard_gaps()** (см.).

Аргументы:

in	<i>bcprog</i>	Дескриптор программы КШ в ОЗУ КШ.
----	---------------	-----------------------------------

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bcp_desrtoy (UEM_OBJHANDLE *bcprog*)

Уничтожение объекта "программа КШ" в ОЗУ КШ.

Уничтожение программы КШ не уничтожает включенные в нее командные сегменты.

Аргументы:

in	<i>bcprog</i>	Дескриптор программы КШ в ОЗУ КШ.
----	---------------	-----------------------------------

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_bcp_set_standard_gaps (UEM_OBJHANDLE *bcprog*)

Расчет и установка стандартных пауз между сообщениями (необязательно).

Данная функция рассчитывает и устанавливает стандартные паузы между сообщениями, включенными в программу КШ, за исключением тех сообщений, для которых паузы уже были установлены ранее при помощи **uem_cseg_gap_set()**, данной функции, или другим способом.

Обращение к данной функции необязательно, так как она вызывается автоматически из функции **uem_bcp_install()**.

Стандартная пауза устанавливается как интервал времени между началом передачи предыдущего командного сегмента и началом передачи данного командного сегмента. При расчете паузы учитывается продолжительность командного и ответных сегментов предыдущего сообщения, таймаут (таймауты) ответа **UEM_RTMO** и время минимальной паузы между сообщениями **UEM_MIN_T2**. Таким образом, устанавливается минимальная пауза, обеспечивающая передачу данных без нарушений протокола, при условии, что ОУ также не нарушает протокол.

Для первого сообщения зацикленного кадра учитывается как сообщение, предшествующее по порядку сообщений в программе КШ, так и сообщение, предшествующее при повторах.

Возможен конфликт выбора шины для отсчета паузы в первом сообщения зацикленного кадра. Данная функция в этом случае отдает предпочтение сообщению, предшествующему при повторах. Для разрешения конфликта рекомендуется в этом случае задавать паузу и правила ее отсчета самостоятельно при помощи **uem_cseg_gap_set()** или **uem_bc_gap_create()**.

Аргументы:

in	<i>bcprog</i>	Дескриптор программы КШ в ОЗУ КШ.
----	---------------	-----------------------------------

Возвращает:

Код завершения. См. Коды завершения.

Запуск и остановка КШ

Порядок запуска, остановки и контроля активности КШ.

Запуск виртуального КШ выполняется функциями **uem_start()** или **uem_bc_start()**, а остановка - функциями **uem_stop()** или **uem_bc_stop()**.

В зависимости от программы КШ, КШ также может останавливаться самостоятельно, при исчерпании заданной последовательности сообщений. Для проверки состояния активности КШ используются функции **uem_is_running()** или **uem_bc_state_ex()**.

Переходы расширенного состояния КШ приведены в следующей таблице:

Исходное состояние	Событие	Состояние после события
UEM_BC_STOPPED	uem_start()	UEM_BC_RUNNING
UEM_BC_STOPPED	uem_bc_start() с параметром flags = UEM_BC_START_NOW	UEM_BC_RUNNING
UEM_BC_STOPPED	uem_bc_start() с параметром flags =	UEM_BC_WAITING

Исходное состояние	Событие	Состояние после события
	UEM_BC_START_WAITING	
UEM_BC_WAITING	uem_start()	UEM_BC_RUNNING
UEM_BC_WAITING	uem_bc_start() с параметром flags = UEM_BC_START_NOW	UEM_BC_RUNNING
UEM_BC_WAITING	Сигнал sync_in_2 или его внутренняя имитация	UEM_BC_RUNNING
UEM_BC_WAITING	Получение ОУ команды управления "Принять управление интерфейсом", когда в ответном сегменте установлен флаг UEM_RTDES_DBCA_BCS TART	UEM_BC_RUNNING
UEM_BC_WAITING	uem_stop()	UEM_BC_STOPPED
UEM_BC_WAITING	uem_bc_stop() с параметром flags = UEM_BC_STOP_NOW	UEM_BC_STOPPED
UEM_BC_RUNNING	uem_stop()	UEM_BC_STOPPED
UEM_BC_RUNNING	uem_bc_stop() с параметром flags = UEM_BC_STOP_NOW	UEM_BC_STOPPED
UEM_BC_RUNNING	uem_bc_stop() с параметром flags = UEM_BC_STOP_ON_FRAME	UEM_BC_RUNNING , ожидание завершения кадра
UEM_BC_RUNNING	Ожидаемое завершение кадра	UEM_BC_WAITING
UEM_BC_RUNNING	Завершение программы КИИ (отработка флага кадра UEM_FRAME_STOP)	UEM_BC_WAITING

Передача сообщений

Описание функции передачи отдельного сообщения.

Функции

- ViStatus **uem_bc_send_receive** (**UEM_DEVHANDLE** bc, **UEM_CMD_SEG** *cseg_data, **UEM_VM_MESSAGE** **msg_and_resp)
Передача отдельного сообщения и получение ответа на него.

Подробное описание

Описание функции передачи отдельного сообщения.

Функции

ViStatus uem_bc_send_receive (UEM_DEVHANDLE *bc*, UEM_CMD_SEG * *cseg_data*, UEM_VM_MESSAGE ** *msg_and_resp*)

Передача отдельного сообщения и получение ответа на него.

Данная функция не требует предварительного конфигурирования КШ путем создания командных сегментов, кадров и программы КШ.

Для выполнения этого действия используются виртуальные КШ и МШ в составе УЭМ. Следует иметь в виду, что функция их переконфигурирует. Открывать МШ для выполнения данной функции не обязательно.

В сообщениях, отправляемых таким образом, нельзя регулировать паузы и вносить ошибки кодирования.

Аргументы:

in	<i>bc</i>	Дескриптор виртуального КШ.
in	<i>cseg_data</i>	Образ командного сегмента, который надо передать.
out	<i>msg_and_resp</i>	Переданный командный сегмент и ответ на него, по данным МШ. В эту переменную записывается указатель на структуру UEM_VM_MESSAGE . Память для этой структуры выделяется динамически, и приложение ответственно за ее освобождение.

Возвращает:

Код завершения. См. **Коды завершения**.

Функции ОУ

Описания функций оконечного устройства.

Функции

- ViStatus **uem_response_create** (UEM_OBJHANDLE *resp, UEM_DEVHANDLE rt, UEM_DWORD rtdes, UEM_DWORD illeg_mask, UEM_WORD status, UEM_WORD ndatawords, UEM_WORD *data)
Создание ответного сегмента.
- ViStatus **uem_response_read** (UEM_OBJHANDLE resp, UEM_DWORD *rtdes, UEM_DWORD *illeg_mask, UEM_WORD *status, UEM_WORD *ndatawords, UEM_WORD *data)
Считывание ответного сегмента.
- ViStatus **uem_response_gap_set** (UEM_OBJHANDLE resp, UEM_WORD gap)
Установка паузы перед передачей ответного сегмента.
- ViStatus **uem_response_gap_get** (UEM_OBJHANDLE resp, UEM_WORD *gap)
Считывание паузы перед передачей ответного сегмента.
- ViStatus **uem_response_word_gap_set** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_WORD gap)
Установка паузы перед передачей слова ответного сегмента.
- ViStatus **uem_response_word_gap_get** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_WORD *gap)
Считывание паузы перед передачей слова ответного сегмента.
- ViStatus **uem_response_error_set** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_ERROR_TYPE error_type, ViInt32 error_pos, ViInt32 error_param)
Внесение ошибок кодирования в ответный сегмент.
- ViStatus **uem_response_error_get** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_ERROR_TYPE *error_type, ViInt32 *error_pos, ViInt32 *error_param)
Считывание ошибок кодирования из ответного сегмента.
- ViStatus **uem_response_sync_set** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_SYNC sync)
Установка типа синхроимпульса.
- ViStatus **uem_response_sync_get** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_SYNC *sync)
Считывание типа синхроимпульса.
- ViStatus **uem_rt_install_response** (UEM_DEVHANDLE rt, UEM_BOOL transmit, UEM_WORD sa, UEM_OBJHANDLE resp)
Установка ответного сегмента как ответа на команду передачи данных.
- ViStatus **uem_rt_install_response_MODE** (UEM_DEVHANDLE rt, UEM_BOOL transmit, UEM_WORD mode, UEM_WORD modecode, UEM_OBJHANDLE resp)
Установка ответного сегмента как ответа на команду управления.
- ViStatus **uem_rt_discover_response** (UEM_DEVHANDLE rt, UEM_BOOL transmit, UEM_WORD sa, UEM_OBJHANDLE *resp)
Выяснение ответа на команду передачи данных.
- ViStatus **uem_rt_discover_response_MODE** (UEM_DEVHANDLE rt, UEM_BOOL transmit, UEM_WORD mode, UEM_WORD modecode, UEM_OBJHANDLE *resp)
Выяснение ответа на команду управления.
- ViStatus **uem_response_destroy** (UEM_OBJHANDLE resp)
Уничтожение объекта ответного сегмента в ОЗУ ОУ.

Признаки обработки командного слова в ОУ

Из объединения (по |) этих констант составляется аргумент `rtdes` в функциях `uem_response_create()`, `uem_response_read()`.

- `#define UEM_RTDES_SW_DIS (1<<15)`
Не отвечать.
- `#define UEM_RTDES_DBCA (1<<6)`
Принять управление интерфейсом.
- `#define UEM_RTDES_DBCA_BCSTART (1<<7)`
Запустить КШ.
- `#define UEM_RTDES_COM_ILLEGAL (1<<17)`
Недопустимая команда.
- `#define UEM_RTDES_LCMD_DW (1<<13)`
Передать последнюю команду.
- `#define UEM_RTDES_SWB_SAV (1<<12)`
Автоматическое формирование флагов ОС.
- `#define UEM_RTDES_WRONG_CH (1<<14)`
Отвечать по другой шине.
- `#define UEM_RTDES_WA (1<<8)`
Циркулярный возврат данных.
- `#define UEM_RTDES_WA_BRCST (1<<31)`
Циркулярный возврат в групповых командах.
- `#define UEM_RTDES_ILLEG_MASK (1<<16)`
Задать маску допустимых команд в зависимости от количества слов данных.
- `#define UEM_RTDES_DEFAULT (UEM_RTDES_SWB_SAV)`
Значение параметра `rtdes` по умолчанию.

Подробное описание

Описания функций оконечного устройства.

Для использования этих функций необходимо получить дескриптор виртуального ОУ при помощи функции `uem_rt_init()`.

Конфигурирование ОУ заключается в создании и настройке ответных сегментов и установке этих сегментов в качестве ответов на заданные командные слова МКПД. (См. **Командные и ответные сегменты.**)

Для того чтобы ОУ отвечал на командные слова в соответствии с установленными ответами, его необходимо запустить в работу при помощи функции `uem_start()`.

Макросы

`#define UEM_RTDES_SW_DIS (1<<15)`

Не отвечать.

Если флаг установлен (`UEM_RTDES_SW_DIS=1`), то ОС и СД (если они есть) не передаются в МКПД. При установке данного флага остальные флаги игнорируются.

#define UEM_RTDES_DBCA (1<<6)

Принять управление интерфейсом.

Если флаг установлен (UEM_RTDES_DBCA=1), то вне зависимости от кода КС и соответствующего формата сообщения, в ОС устанавливается признак **Принято управление интерфейсом** (если установлен флаг **UEM_RTDES_SWB_SAV**), СД не передаются, после завершения передачи ОС данное виртуальное устройство ОУ отключается. Возможное использование данного флага – поддержка возможности обработки КУ «Принять управление интерфейсом», если данная КУ применима для выбранного адреса ОУ.

#define UEM_RTDES_DBCA_BCSTART (1<<7)

Запустить КШ.

Флаг значим только при установленном флаге **UEM_RTDES_DBCA**. Если флаг установлен (UEM_RTDES_DBCA_BCSTART=1), и если КШ в момент передачи текущего ОС находится в состоянии ожидания, то КШ запускается. Возможное использование данного флага – поддержка возможности обработки КУ «Принять управление интерфейсом» с передачей управления собственному КШ, если данная КУ применима для выбранного адреса ОУ.

#define UEM_RTDES_COM_ILLEGAL (1<<17)

Недопустимая команда.

Флаг устанавливает признак недопустимости ОС для всех КС, для которых установлен данный ответ. В этом случае в ОС устанавливается бит статуса **ОШС**, слова данных в ответе не передаются. При сброшенном флаге КС считаются допустимыми. Допустимость/недопустимость в зависимости от числа слов данных можно установить в параметре **illeg_mask** функции **uem_response_create()**.

#define UEM_RTDES_LCMD_DW (1<<13)

Передать последнюю команду.

При установленном флаге (UEM_RTDES_LCMD_DW=1) и не установленных флагах **UEM_RTDES_SW_DIS**, **UEM_RTDES_DBCA** после передачи ОС, вне зависимости от значения параметра **ndatawords** функции **uem_response_create()**, передается одно СД, содержащее код последней предшествующей достоверной команды к данному адресу ОУ. Исключением является код команды управления «Передать последнюю команду». Возможное использование данного флага – поддержка возможности обработки КУ «Передать последнюю команду».

Примечание: в данном случае будут изменены только информационные разряды первого СД. Все остальные параметры должны заполняться корректно по общим правилам. В частности, ответный сегмент должен содержать как минимум одно слово данных. Внесенные в это слово ошибки кодирования, синхроимпульса, паузы будут отработаны оборудованием.

#define UEM_RTDES_SWB_SAV (1<<12)

Автоматическое формирование флагов ОС.

Флаг значим только при не установленном флаге **UEM_RTDES_SW_DIS** и определяет правила формирования флагов ОС, которые приведены в **таблице 3**.

Таблица 3. Правила формирования признаков ОС

Флаги ОС	UEM_RTDES_SWB_SAV=0 (прямое задание значений разрядов ОС)	UEM_RTDES_SWB_SAV=1 (автоматическое формирование значений разрядов ОС в зависимости от контекста предыдущих сообщений в МКПД)
«Адрес ОУ»	Определяется разрядами [15:11] параметра status функции uem_response_create()	Определяется адресом ОУ в КС
«Ошибка в сообщении»	Определяется разрядом [10] параметра status функции uem_response_create()	Определяется исходя из значения флагов UEM_RTDES_COM_ILLEGAL , параметра illeg_mask функции uem_response_create() и таблицы допустимости (настройка допустимости имеет приоритет), состояния достоверности и допустимости предыдущего КС, с учетом правил ГОСТ Р 52070 для КУ «Передать ОС» и «Передать последнюю команду»
«Передача ОС»	Определяется разрядом [9] параметра status функции uem_response_create()	Нулевое значение
«Запрос на обслуживание»	Определяется разрядом [8] параметра status функции uem_response_create()	Определяется разрядом [8] параметра status функции uem_response_create()
Резервные	Определяются разрядами [7:5] параметра status функции uem_response_create()	Нулевое значение
«Принята групповая команда»	Определяется разрядом [3] параметра status функции uem_response_create()	Определяется исходя из состояния и значений текущего и предыдущего КС, устанавливается в «1» в ответ на КУ «Передать ОС» и «Передать последнюю команду»
«Абонент занят»	Определяется разрядом [4] параметра status функции uem_response_create()	Определяется разрядом [4] параметра status функции uem_response_create()
«Неисправность абонента»	Определяется разрядом [2] параметра status функции uem_response_create()	Определяется разрядом [2] параметра status функции uem_response_create()

Флаги ОС	UEM_RTDES_SWB_SAV=0 (прямое задание значений разрядов ОС)	UEM_RTDES_SWB_SAV=1 (автоматическое формирование значений разрядов ОС в зависимости от контекста предыдущих сообщений в МКПД)
«Принято управление интерфейсом»	Определяется разрядом [1] параметра status функции uem_response_create()	Определяется значением флага UEM_RTDES_DBCA
«Неисправность ОУ»	Определяется разрядом [0] параметра status функции uem_response_create()	Определяется разрядом [0] параметра status функции uem_response_create() и текущим состоянием блокировки признака «Неисправность ОУ» по соответствующим КУ, если нет установки запрета обработки таких КУ флагом UEM_BRTF_DIS

#define UEM_RTDES_WRONG_CH (1<<14)

Отвечать по другой шине.

Если флаг установлен (**UEM_RTDES_WRONG_CH=1**), то ОС (если его передача предусмотрена) и СД (если они есть) передаются по резервной шине МКПД по отношению к той шине, по которой поступило КС.

#define UEM_RTDES_WA (1<<8)

Циркулярный возврат данных.

Флаг использования циркулярного возврата данных, значим только при нулевых значениях флагов **UEM_RTDES_SW_DIS**, **UEM_DBCA**, **UEM_RTDES_LCMD_DW**, и только в описателях КС информационного обмена (подадреса от 1 до 30 включительно, а также если установлен запрет команд управления, подадреса 0 и 31).

Установка данного флага (**UEM_RTDES_WA=1**) для КС на прием означает, что будут сохранены поступающие после данного КС СД (если они фактически есть) в отдельном (для данного адреса ОУ) буфере данных циркулярного возврата. Установка данного флага (**UEM_RTDES_WA=1**) для КС на передачу означает, что в передаваемых СД (если их передача происходит) будут находиться ранее сохраненные данные из буфера циркулярного возврата.

Рекомендация по включению режима циркулярного возврата для выбранного адреса ОУ.

Установить флаг **UEM_RTDES_WA** для одного из КС приема (с нужным значением подадреса приема циркулярного возврата данных) и одного из КС передачи (с нужным значением подадреса передачи циркулярного возврата данных), при необходимости использовать данные групповых сообщений – также для данного адреса ОУ включить разрешение приема слов данных в групповых командах.

ГОСТ Р 52070 рекомендует использовать только подадрес 30 для приема и передачи циркулярного возврата. В общем случае, если флаг **UEM_RTDES_WA** установлен для нескольких кодов КС,

обрабатываемых данным ОУ, принимаемые данные будут направляться в один и тот же буфер приема, и могут быть переданы из этого буфера в ответ на КС с несколькими подадресами.

#define UEM_RTDES_WA_BRCST (1<<31)

Циркулярный возврат в групповых командах.

Флаг обеспечивает запоминание в буфере слов данных, поступающих в групповых командах (аналогично флагу **UEM_RTDES_WA** для негрупповых команд).

#define UEM_RTDES_ILLEG_MASK (1<<16)

Задать маску допустимых команд в зависимости от количества слов данных.

При установленном флаге (**UEM_RTDES_ILLEG_MASK = 1**) допустимость команд в зависимости указанного в них количества слов данных определяется параметром **illeg_mask** в функции **uem_response_create()**. При сброшенном флаге (**UEM_RTDES_ILLEG_MASK = 0**) команды считаются допустимыми при любом числе слов данных, параметр **illeg_mask** игнорируется. Флаг не должен устанавливаться в ответах на команды управления.

#define UEM_RTDES_DEFAULT (UEM_RTDES_SWB_SAV)

Значение параметра **rtdes** по умолчанию.

Функции

ViStatus uem_response_create (UEM_OBJHANDLE * resp, UEM_DEVHANDLE rt, UEM_DWORD rtdes, UEM_DWORD illeg_mask, UEM_WORD status, UEM_WORD ndatawords, UEM_WORD * data)

Создание ответного сегмента.

Аргументы:

out	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
in	<i>rt</i>	Дескриптор ОУ.
in	<i>rtdes</i>	Битовая строка признаков правил обработки командного слова в ОУ. Составляется как объединение (по) констант UEM_RTDES_XXXX . По умолчанию выставляется UEM_RTDES_DEFAULT .
in	<i>illeg_mask</i>	Битовая маска недопустимости команд в зависимости от числа СД. Данный аргумент используется, если в аргументе rtdes установлен бит UEM_RTDES_ILLEG_MASK и не установлены биты UEM_RTDES_COM_ILLEGAL , UEM_RTDES_SW_DIS , в противном случае - игнорируется. Бит с номером n - признак недопустимости командного слова с числом слов данных n, 1 <= n <= 31. Бит с номером 0 - признак недопустимости командного слова с числом слов данных 32. Значение 1 в бите означает, что команда недопустима, значение 0 - допустима.
in	<i>status</i>	Ответное слово.

in	<i>ndatawords</i>	Число слов данных. Допустимые значения: 0-63. Аппаратура УЭМ способна формировать до 62 слов данных в ответном сегменте. Специальная константа 63 (UEM_NDATA_BY_CW) указывает, что число слов данных в ответном сегменте определяется полученным командным словом. Фактическое число слов данных в массиве data в этом случае должно быть 32.
in	<i>data</i>	Массив значений слов данных. Длина определяется аргументом ndatawords .

Возвращает:

Код завершения. См. **Коды завершения**.

Ответный сегмент создается без внесенных ошибок. Пауза перед ответным сегментом определяется временным параметром **UEM_MIN_T1**, заданным для данного виртуального ОУ.

ViStatus uem_response_read (UEM_OBJHANDLE resp, UEM_DWORD * rtdes, UEM_DWORD * illeg_mask, UEM_WORD * status, UEM_WORD * ndatawords, UEM_WORD * data)

Считывание ответного сегмента.

Функция позволяет прочитать данные ответного сегмента, созданного функцией **uem_response_create()**, обратно в ОЗУ управляющей ПЭВМ.

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
out	<i>rtdes</i>	Битовая строка признаков правил обработки командного слова в ОУ. Объединение констант UEM_RTDES_XXXX .
out	<i>illeg_mask</i>	Битовая маска недопустимости команд в зависимости от числа СД. См. illeg_mask в uem_response_create() .
out	<i>status</i>	Ответное слово.
out	<i>ndatawords</i>	Число слов данных.
out	<i>data</i>	Массив значений слов данных. Длина должна быть не меньше 62 слов.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_response_gap_set (UEM_OBJHANDLE resp, UEM_WORD gap)

Установка паузы перед передачей ответного сегмента.

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
in	<i>gap</i>	Значение паузы. Задается в единицах по 0,25 мкс. Допустимо: 0-65535. (См. Диапазоны значений).

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_response_gap_get (UEM_OBJHANDLE resp, UEM_WORD * gap)

Считывание паузы перед передачей ответного сегмента.

Функция считывает значение паузы, установленное функцией **uem_response_gap_set()**, либо установленное автоматически в **uem_response_create()**.

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
out	<i>gap</i>	Значение паузы.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_response_word_gap_set (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_WORD gap)

Установка паузы перед передачей слова ответного сегмента.

Функция полностью аналогична функции **uem_response_gap_set()**, но позволяет установить паузу не перед первым, а перед любым словом ответного сегмента. Единственная причина использовать эту функцию - для внесения ошибки типа "разрыв сегмента".

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
in	<i>wordnumber</i>	Номер слова. Нумерация с 0, сквозная, сначала ответное слово, потом слова данных.
in	<i>gap</i>	Значение паузы. Задается в единицах по 0,25 мкс. Допустимо: 0-65535. (См. Диапазоны значений).

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_response_word_gap_get (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_WORD * gap)

Считывание паузы перед передачей слова ответного сегмента.

Функция считывает значение паузы, установленное функцией **uem_response_word_gap_set()**.

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
in	<i>wordnumber</i>	Номер слова. Нумерация с 0, сквозная, сначала ответное слово, потом слова данных.
out	<i>gap</i>	Значение паузы.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_response_error_set (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_ERROR_TYPE error_type, Vilnt32 error_pos, Vilnt32 error_param)

Внесение ошибок кодирования в ответный сегмент.

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
in	<i>wordnumber</i>	Номер слова. Нумерация с 0, сквозная, сначала ответное слово, потом слова данных.
in	<i>error_type</i>	Тип вносимой ошибки. См. Типы вносимых ошибок кодирования .
in	<i>error_pos</i>	Позиция ошибки. Интерпретируется в зависимости от типа ошибки.

in	<i>error_param</i>	Дополнительный параметр. Интерпретируется в зависимости от типа ошибки.
----	--------------------	---

Возвращает:

Код завершения. См. **Коды завершения**.

Допускается внести ошибки кодирования в несколько слов ответного сегмента.

ViStatus uem_response_error_get (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_ERROR_TYPE * error_type, Vilnt32 * error_pos, Vilnt32 * error_param)

Считывание ошибок кодирования из ответного сегмента.

Функция позволяет прочитать данные ошибок кодирования, внесенных функцией **uem_response_error_set()**, обратно в ОЗУ управляющей ПЭВМ.

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
in	<i>wordnumber</i>	Номер слова. Нумерация с 0, сквозная, сначала ответное слово, потом слова данных.
out	<i>error_type</i>	Тип вносимой ошибки.
out	<i>error_pos</i>	Позиция ошибки. Интерпретируется в зависимости от типа ошибки.
out	<i>error_param</i>	Дополнительный параметр. Интерпретируется в зависимости от типа ошибки.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_response_sync_set (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_SYNC sync)

Установка типа синхроимпульса.

При создании сегмента тип синхроимпульса для каждого слова уже установлен корректно. Программная установка типа синхроимпульса может использоваться для следующих целей:

- внесение ошибок типа "неверный синхроимпульс",
- в сочетании с внесением ошибок кодирования **UEM_ERRT_BAD_SYNCHRO** расширяет номенклатуру искаженных форм синхроимпульса.

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
in	<i>wordnumber</i>	Номер слова, в котором устанавливается синхроимпульс. Нумерация с 0, сквозная, сначала ответное слово, потом слова данных.
in	<i>sync</i>	Тип синхроимпульса.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_response_sync_get (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_SYNC * sync)

Считывание типа синхроимпульса.

Функция читает тип синхроимпульса в слове сегмента, установленный при создании сегмента или функцией **uem_response_sync_set()**.

Аргументы:

in	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
in	<i>wordnumber</i>	Номер слова. Нумерация с 0, сквозная, сначала ответное слово, потом слова данных.
out	<i>sync</i>	Тип синхроимпульса.

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_rt_install_response (UEM_DEVHANDLE *rt*, UEM_BOOL *transmit*, UEM_WORD *sa*, UEM_OBJHANDLE *resp*)

Установка ответного сегмента как ответа на команду передачи данных.

Данная функция устанавливает указанный ответный сегмент в качестве ответа на команду передачи данных.

Аргументы:

in	<i>rt</i>	Дескриптор виртуального ОУ.
in	<i>transmit</i>	Признак Передачи(1)/Приема(0) в командном слове.
in	<i>sa</i>	Подадрес
in	<i>resp</i>	<p>Дескриптор ответного сегмента в ОЗУ ОУ. Указанный ответный сегмент будет установлен в качестве ответа на указанное командное слово.</p> <p>Этот аргумент также может принимать нулевое значение (NULL), которое означает указание деинсталлировать ответ на указанное командное слово, ничем его не заменяя.</p> <p>В ответ на командные слова, для которых не задан ответный сегмент, автоматически генерируется ответ в виде одного ответного слова с признаком "Ошибка в сообщении".</p> <p>Допускается устанавливать один и тот же ответный сегмент в качестве ответа на разные команды.</p>

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_rt_install_response_MODE (UEM_DEVHANDLE *rt*, UEM_BOOL *transmit*, UEM_WORD *mode*, UEM_WORD *modecode*, UEM_OBJHANDLE *resp*)

Установка ответного сегмента как ответа на команду управления.

Данная функция устанавливает указанный ответный сегмент в качестве ответа на команду управления.

Аргументы:

in	<i>rt</i>	Дескриптор виртуального ОУ.
in	<i>transmit</i>	Признак Передачи(1)/Приема(0) в командном слове.
in	<i>mode</i>	Код режима управления. Допускаются значения 0 (UEM_MODE_0) и 31 (UEM_MODE_31).
in	<i>modecode</i>	Код команды управления.

in	<i>resp</i>	<p>Дескриптор ответного сегмента в ОЗУ ОУ. Указанный ответный сегмент будет установлен в качестве ответа на указанное командное слово.</p> <p>Этот аргумент также может принимать нулевое значение (NULL), которое означает указание деинсталлировать ответ на указанное командное слово, ничем его не заменяя.</p> <p>В ответ на командные слова, для которых не задан ответный сегмент, автоматически генерируется ответ в виде одного ответного слова с признаком "Ошибка в сообщении".</p> <p>Допускается устанавливать один и тот же ответный сегмент в качестве ответа на разные команды.</p>
----	-------------	---

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_rt_discover_response (UEM_DEVHANDLE *rt*, UEM_BOOL *transmit*, UEM_WORD *sa*, UEM_OBJHANDLE * *resp*)

Выяснение ответа на команду передачи данных.

Данная функция позволяет выяснить, какой ответный сегмент установлен функцией **uem_rt_install_response()** в качестве ответа на указанную команду. Аргументы полностью аналогичны параметрам **uem_rt_install_response()**.

Аргументы:

in	<i>rt</i>	Дескриптор виртуального ОУ.
in	<i>transmit</i>	Признак Передачи(1)/Приема(0) в командном слове.
in	<i>sa</i>	Подадрес.
out	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ. Если ответ на указанную команду не установлен, в качестве значения в этом аргументе будет возвращен NULL.

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_rt_discover_response_MODE (UEM_DEVHANDLE *rt*, UEM_BOOL *transmit*, UEM_WORD *mode*, UEM_WORD *modecode*, UEM_OBJHANDLE * *resp*)

Выяснение ответа на команду управления.

Данная функция позволяет выяснить, какой ответный сегмент установлен функцией **uem_rt_install_response_MODE()** в качестве ответа на указанную команду. Аргументы полностью аналогичны параметрам **uem_rt_install_response_MODE()**.

Аргументы:

in	<i>rt</i>	Дескриптор виртуального ОУ.
in	<i>transmit</i>	Признак Передачи(1)/Приема(0) в командном слове.
in	<i>mode</i>	Код режима управления. Допускаются значения 0 (UEM_MODE_0) и 31 (UEM_MODE_31).
in	<i>modecode</i>	Код команды управления.
out	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ. Если ответ на указанную команду не установлен, в качестве значения в этом аргументе будет возвращен NULL.

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_response_destroy (UEM_OBJHANDLE *resp*)

Уничтожение объекта ответного сегмента в ОЗУ ОУ.

Аргументы:

<i>in</i>	<i>resp</i>	Дескриптор ответного сегмента в ОЗУ ОУ.
-----------	-------------	---

Возвращает:

Код завершения. См. **Коды завершения**.

Функции МШ

Описания функций монитора шины.

Определения типов

- typedef void(* **uem_bm_handler**) (UEM_DEVHANDLE bm, void *userdata)
Обработчик события МШ.

Функции

- ViStatus **uem_bm_receive** (UEM_DEVHANDLE bm, UEM_BM_MESSAGE **message_data)
Считывание очередного сообщения, принятого монитором шины.
- ViStatus **uem_bm_queue_count** (UEM_DEVHANDLE bm, UEM_DWORD *count)
Запрос размера очереди сообщений, принятых монитором.
- ViStatus **uem_bm_install_handler** (UEM_DEVHANDLE bm, **uem_bm_handler** handler, void *userdata)
Установка обработчика события МШ.

Подробное описание

Описания функций монитора шины.

Для использования этих функций необходимо получить дескриптор виртуального МШ при помощи функции **uem_bm_init**().

Запуск МШ в работу выполняется функцией **uem_start**(). При этом МШ начинает принимать с МКПД и записывать в свою внутреннюю очередь сообщения МКПД. Приложение считывает эти сообщения из очереди при помощи функции **uem_bm_receive**(). Остановка МШ выполняется функцией **uem_stop**(). При этом прием данных МКПД прекращается. Следует иметь в виду, что после остановки МШ в его внутренней очереди могут оставаться сообщения, еще не считанные приложением, которые следует дочитать при помощи **uem_bm_receive**().

Типы

typedef void(* **uem_bm_handler**) (UEM_DEVHANDLE bm, void *userdata)

Обработчик события МШ.

Тип указателя на функцию-обработчика события, вызываемую монитором шины при приходе нового сообщения (сообщений) МКПД.

Аргументы:

in	<i>bm</i>	Дескриптор виртуального МШ.
in	<i>userdata</i>	Указатель на произвольные данные приложения, установленный при установке обработчика.

Функции

ViStatus **uem_bm_receive** (UEM_DEVHANDLE *bm*, UEM_BM_MESSAGE ** *message_data*)

Считывание очередного сообщения, принятого монитором шины.

Функция извлекает из буфера МШ очередное сообщение. Ситуация, когда в буфере МШ нет принятых и несчитанных сообщений, индицируется кодом завершения **UEM_WARN_NO_NEXT_MESSAGE**.

Внутренняя очередь принятых сообщений может вместить до 210000 сообщений; при максимальном темпе передачи сообщений МКПД очереди хватает на 5 сек., при более низком темпе - на больший интервал времени. Приложение должно считывать все сообщения из очереди не реже 1 раза в 5 сек, рекомендуемый темп - 20 раз в секунду.

Аргументы:

in	<i>bm</i>	Дескриптор виртуального МШ.
out	<i>message_data</i>	Через эту переменную передается указатель на ячейку памяти, в которую МШ записывает указатель на извлеченное сообщение. В случае отсутствия очередного сообщения или ошибки в работе функция записывает в эту переменную значение NULL. Память для извлеченного сообщения выделяется при помощи malloc(), и приложение ответственно за последующее освобождение этой памяти при помощи free().

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bm_queue_count (UEM_DEVHANDLE *bm*, UEM_DWORD * *count*)

Запрос размера очереди сообщений, принятых монитором.

Функция сообщает приложению количество сообщений МКПД, принятых МШ и ожидающих считывания.

Аргументы:

in	<i>bm</i>	Дескриптор виртуального МШ.
out	<i>count</i>	Число сообщений в очереди.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bm_install_handler (UEM_DEVHANDLE *bm*, uem_bm_handler *handler*, void * *userdata*)

Установка обработчика события МШ.

Указанная функция будет вызываться МШ при получении очередного сообщения или группы сообщений МКПД.

Аргументы:

in	<i>bm</i>	Дескриптор виртуального МШ.
in	<i>handler</i>	Функция-обработчик события. Этот аргумент также может быть NULL, что означает отмену функции-обработчика события.
in	<i>userdata</i>	Указатель на произвольные данные приложения. Этот указатель будет передаваться в функцию-обработчик события handler при каждом вызове. Если приложению такой указатель не требуется, следует указать NULL.

Возвращает:

Код завершения. См. **Коды завершения**.

Запуск и остановка

Функции запуска, остановки и проверки активности виртуальных устройств.

Функции

- ViStatus **uem_start** (UEM_DEVHANDLE anydev)
Запуск любого виртуального устройства в составе УЭМ.
- ViStatus **uem_stop** (UEM_DEVHANDLE anydev)
Остановка любого виртуального устройства в составе УЭМ.
- ViStatus **uem_is_running** (UEM_DEVHANDLE anydev, UEM_BOOL *running)
Проверка активности виртуального устройства.

Запуск виртуального КШ с дополнительными параметрами

- enum **UEM_BC_STATE_EX** { UEM_BC_STOPPED, UEM_BC_WAITING, UEM_BC_RUNNING }
Расширенное состояние КШ.
- ViStatus **uem_bc_start** (UEM_DEVHANDLE bc, UEM_DWORD flags)
Запуск виртуального КШ в составе УЭМ с дополнительными параметрами.
- ViStatus **uem_bc_state_ex** (UEM_DEVHANDLE bc, UEM_BC_STATE_EX *state)
Запрос расширенного состояния КШ.
- #define **UEM_BC_START_NOW** 0
Нормальный (немедленный) старт.
- #define **UEM_BC_START_WAITING** 1
Переход в режим ожидания, старт по внешнему сигналу или команде.
- #define **UEM_BC_START_DEFAULT** (UEM_BC_START_NOW)
Стандартный способ старта.

Остановка виртуального КШ с дополнительными параметрами

- ViStatus **uem_bc_stop** (UEM_DEVHANDLE bc, UEM_DWORD flags)
Остановка виртуального КШ в составе УЭМ с дополнительными параметрами.
- #define **UEM_BC_STOP_NOW** 0
Нормальная (немедленная) остановка.
- #define **UEM_BC_STOP_ON_FRAME** 1
Остановка по завершению текущего кадра.
- #define **UEM_BC_STOP_DEFAULT** (UEM_BC_STOP_NOW)
Стандартный способ остановки.

Подробное описание

Функции запуска, остановки и проверки активности виртуальных устройств.

Данный раздел содержит описание функций запуска, остановки и проверки активности виртуальных устройств (КШ/ОУ/МШ) в составе УЭМ.

Это универсальные функции, не зависящие от типа виртуального устройства. Для виртуального КШ, при необходимости запуска и остановки с дополнительными параметрами, имеются также специальные функции для КШ.

Перед запуском виртуальное устройство должно быть сконфигурировано при помощи функций соответствующего раздела.

Макросы

#define UEM_BC_START_NOW 0

Нормальный (немедленный) старт.

#define UEM_BC_START_WAITING 1

Переход в режим ожидания, старт по внешнему сигналу или команде.

В режиме ожидания старт выполняется в следующих случаях:

При получении внешнего сигнала синхронизации **sync_in_2** [2, 3].

При срабатывании внутреннего имитатора внешнего сигнала синхронизации **sync_in_2** (см. **UEM_SYNC_IN_2_INTGEN**, **UEM_IST2**, а также и **UEM_SYNC_IN_2_SET**).

При получении виртуальным ОУ в составе УЭМ команды управления "Принять управления интерфейсом" (см. **uem_response_create()**, **UEM_RTDES_DBCA_BCSTART**).

Для выполнения такого запуска хотя бы один из указанных механизмов должен быть разрешен и сконфигурирован. В противном случае КШ остается в состоянии ожидания до выполнения действия **uem_stop()**.

Для определения состояния КШ, запущенного в режиме ожидания, можно использовать функцию **uem_bc_state_ex()**.

#define UEM_BC_START_DEFAULT (UEM_BC_START_NOW)

Стандартный способ старта.

#define UEM_BC_STOP_NOW 0

Нормальная (немедленная) остановка.

#define UEM_BC_STOP_ON_FRAME 1

Остановка по завершению текущего кадра.

#define UEM_BC_STOP_DEFAULT (UEM_BC_STOP_NOW)

Стандартный способ остановки.

Перечисления

enum UEM_BC_STATE_EX

Расширенное состояние КШ.

Элементы перечислений:

UEM_BC_STOPPED Остановлен.

UEM_BC_WAITING Находится в режиме ожидания.

UEM_BC_RUNNING Работает.

Функции

ViStatus uem_start (UEM_DEVHANDLE anydev)

Запуск любого виртуального устройства в составе УЭМ.

См. также **uem_bc_start()** для виртуального КШ.

Аргументы:

in	anydev	Дескриптор виртуального устройства (КШ, ОУ или МШ).
----	--------	---

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_stop (UEM_DEVHANDLE anydev)

Остановка любого виртуального устройства в составе УЭМ.

См. также **uem_bc_stop()** для виртуального КШ.

Аргументы:

in	anydev	Дескриптор виртуального устройства (КШ, ОУ или МШ).
----	--------	---

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_is_running (UEM_DEVHANDLE anydev, UEM_BOOL * running)

Проверка активности виртуального устройства.

Функция проверяет, работает ли виртуальное устройство в данный момент. Необходимость такой проверки связана с тем, что КШ и ОУ могут останавливаться самостоятельно в зависимости от заданной конфигурации и происходящих событий.

Аргументы:

in	anydev	Дескриптор виртуального устройства.
out	running	В эту переменную записывается результат проверки: 1 - устройство работает, 0 - устройство остановлено.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bc_start (UEM_DEVHANDLE bc, UEM_DWORD flags)

Запуск виртуального КШ в составе УЭМ с дополнительными параметрами.

Аргументы:

in	bc	Дескриптор виртуального КШ.
in	flags	Дополнительные параметры. Возможные значения UEM_BC_START_NOW , UEM_BC_START_WAITING .

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bc_state_ex (UEM_DEVHANDLE *bc*, UEM_BC_STATE_EX * *state*)

Запрос расширенного состояния КШ.

Данная функция может использоваться в дополнение или вместо функции **uem_is_running()** для КШ, запущенного в режиме ожидания (**UEM_BC_START_WAITING**).

Аргументы:

in	<i>bc</i>	Дескриптор виртуального КШ.
out	<i>state</i>	Расширенное состояние КШ.

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_bc_stop (UEM_DEVHANDLE *bc*, UEM_DWORD *flags*)

Остановка виртуального КШ в составе УЭМ с дополнительными параметрами.

Аргументы:

in	<i>bc</i>	Дескриптор виртуального КШ.
in	<i>flags</i>	Дополнительные параметры. Возможные значения UEM_BC_STOP_NOW , UEM_BC_STOP_ON_FRAME .

Возвращает:

Код завершения. См. **Коды завершения**.

При остановке по завершению текущего кадра (**flags == UEM_BC_STOP_ON_FRAME**) функция выполняется как неблокирующая. Остановка КШ произойдет по завершению текущего кадра, что зависит от состава и параметров кадра. Для определения фактического состояния КШ можно использовать функции **uem_is_running()** и **uem_bc_state_ex()**. Для немедленной остановки КШ, не дожидаясь завершения текущего кадра, допускается повторно вызвать данную функцию с параметром **flags == UEM_BC_STOP_NOW**.

Служебные функции

Описания служебных функций, стандартных для драйвера инструмента.

Макросы

- `#define UEM_LIB_REV 0x0100`
Номер версии библиотеки.

Функции

- `ViStatus uem_error_message (UEM_DEVHANDLE uem, ViStatus status, ViChar msg[])`
Запрос сообщения об ошибке.
- `ViStatus uem_error_query (ViSession uem, ViInt32 *status, ViChar msg[])`
Запрос последней ошибки.
- `ViStatus uem_reset (UEM_DEVHANDLE anydev)`
Сброс УЭМ или любого виртуального устройства в составе УЭМ.
- `ViStatus uem_revision_query (UEM_DEVHANDLE uem, ViChar dv[], ViChar iv[])`
Запрос версии.
- `ViStatus uem_self_test (UEM_DEVHANDLE uem, ViInt16 *res, ViChar msg[])`
Самоконтроль.

Подробное описание

Описания служебных функций, стандартных для драйвера инструмента.

Макросы

`#define UEM_LIB_REV 0x0100`

Номер версии библиотеки.

В формате: ((старшая часть номера) << 8) | (младшая часть номера).

Функции

`ViStatus uem_error_message (UEM_DEVHANDLE uem, ViStatus status, ViChar msg[])`

Запрос сообщения об ошибке.

Данная функция получает и интерпретирует код завершения, возвращенный какой-либо функцией драйвера, и возвращает строку сообщения об ошибке. Если код завершения сгенерирован не данным драйвером, функция обращается для получения строки сообщения к аналогичной функции нижележащего драйвера `umtmem`.

Аргументы:

in	<i>uem</i>	Дескриптор УЭМ. Примечание 1: В данной функции в этом аргументе допускается передавать VI_NULL. Примечание 2: В данной функции в этом аргументе допускается вместо дескриптора УЭМ передавать дескриптор любого виртуального устройства в составе УЭМ. Функция выполняет такой вызов, как если бы был передан дескриптор УЭМ.
in	<i>status</i>	Код завершения, возвращенный какой-либо функцией драйвера.
out	<i>msg</i>	Сообщение об ошибке. В данной строке возвращается сообщение, соответствующее переданному коду состояния. Примечание: Строка должна содержать 256 элементов ViChar.

Возвращает:

Код завершения. Возможные значения:

- VI_SUCCESS - код ошибки интерпретирован успешно.
- VI_WARN_UNKNOWN_STATUS - код ошибки неизвестен.

ViStatus uem_error_query (ViSession *uem*, Vilnt32 * *status*, ViChar *msg*[])

Запрос последней ошибки.

Возвращает код ошибки последней операции и соответствующее текстовое сообщение. Функция не реализована.

Аргументы:

in	<i>uem</i>	Дескриптор УЭМ.
out	<i>status</i>	В данной переменной возвращается код ошибки, считанный из очереди ошибок инструмента.
out	<i>msg</i>	В данной строке возвращается сообщение об ошибке, соответствующее коду ошибки. Примечание: Строка должна содержать 256 элементов ViChar.

Возвращает:

Код завершения. Всегда возвращается VI_ERROR_NIMPL_OPER.

ViStatus uem_reset (UEM_DEVHANDLE *anydev*)

Сброс УЭМ или любого виртуального устройства в составе УЭМ.

Для УЭМ в целом или для указанного виртуального устройства выполняется аппаратный сброс. Кроме этого, для виртуальных КШ и ОУ уничтожаются все объекты в ОЗУ. При указании дескриптора УЭМ все виртуальные устройства в составе этого УЭМ также будут сброшены. Если виртуальное устройство было запущено в работу, оно будет остановлено.

Аргументы:

in	<i>anydev</i>	Дескриптор УЭМ или виртуального устройства в составе УЭМ.
----	---------------	---

Возвращает:

Код завершения. См. Коды завершения.

ViStatus uem_revision_query (UEM_DEVHANDLE *uem*, ViChar *dv*[], ViChar *iv*[])

Запрос версии.

Данная функция возвращает версию драйвера и микропрограммы инструмента.

Аргументы:

in	<i>uem</i>	Дескриптор УЭМ. Примечание: В данной функции в этом аргументе допускается передавать VI_NULL.
out	<i>dv</i>	Версия драйвера. Примечание: Строка должна содержать 256 элементов ViChar.
out	<i>iv</i>	Версия инструмента. Примечание: Строка должна содержать 256 элементов ViChar. Примечание: Если в функцию передан сеанс NULL, будет возвращена строка "n/a".

Возвращает:

Код завершения. См. **Коды завершения**.

ViStatus uem_self_test (UEM_DEVHANDLE *uem*, ViInt16 * *res*, ViChar *msg*[])

Самоконтроль.

Данная функция производит самоконтроль инструмента и возвращает его результат.

Во время проведения самоконтроля все виртуальные устройства должны быть закрыты.

Аргументы:

in	<i>uem</i>	Дескриптор УЭМ.
out	<i>res</i>	Результат самоконтроля. Нулевое значение означает успешное прохождение самоконтроля. Прочие коды свидетельствуют об ошибке.
out	<i>msg</i>	Текстовое описание результата самоконтроля. Строка должна содержать 256 элементов ViChar.

Возвращает:

Код завершения. См. **Коды завершения**.

Возможные результаты (*res*) и соответствующие сообщения (*msg*):

- 0 - Тест самоконтроля успешно выполнен.
- -1 - Отказ при выполнении теста самоконтроля.
- -2 - Тест самоконтроля выявил ошибки.

Формирование и разбор командных и ответных слов

Функции формирования и разбора командных и ответных слов.

Функции

- **UEM_WORD uem_command_word** (**UEM_WORD** rt, **UEM_BOOL** tx, **UEM_WORD** sa, **UEM_WORD** ndatawords)
Формирование командного слова.
- **void uem_command_word_parse** (**UEM_WORD** cw, **UEM_WORD** *rt, **UEM_BOOL** *tx, **UEM_WORD** *sa, **UEM_WORD** *ndatawords)
Разбор командного слова.
- **UEM_WORD uem_status_word** (**UEM_WORD** rt, **UEM_WORD** status_bits)
Формирование ответного слова.
- **void uem_status_word_parse** (**UEM_WORD** sw, **UEM_WORD** *rt, **UEM_WORD** *status_bits)
Разбор ответного слова.

Вспомогательные константы для полей командных слов

В соответствии с [1].

- **#define UEM_RTADDR_MIN 0**
Минимальное значение адреса ОУ.
- **#define UEM_RTADDR_MAX 30**
Максимальное значение адреса ОУ.
- **#define UEM_RTADDR_BRCST 31**
Адрес для групповых сообщений.
- **#define UEM_RTADDR_MAX_EXT 31**
Максимальное значение адреса ОУ в сетях с запретом групповых сообщений.
- **#define UEM_RT_RX 0**
Признак передача/прием: прием.
- **#define UEM_RT_TX 1**
Признак передача/прием: передача.
- **#define UEM_SADDR_MIN 1**
Минимальное значение подадреса.
- **#define UEM_SADDR_MAX 30**
Максимальное значение подадреса.
- **#define UEM_SADDR_CONV_LOOPBACK 30**
Традиционный подадрес для тестовой петли.
- **#define UEM_SADDR_MIN_EXT 0**
Минимальное значение подадреса в сетях с запретом команд управления.
- **#define UEM_SADDR_MAX_EXT 31**
Максимальное значение подадреса в сетях с запретом команд управления.
- **#define UEM_MODE_0 0**
Код режима 0.
- **#define UEM_MODE_31 31**
Код режима 31.

Коды команд управления

В соответствии с [1].

- #define UEM_MCODE_ADVC 0
Принять управление интерфейсом.
- #define UEM_MCODE_SYNCHRO 1
Синхронизация.
- #define UEM_MCODE_TXSTATUS 2
Передать ОС.
- #define UEM_MCODE_BSELFTEST 3
Начать самоконтроль.
- #define UEM_MCODE_BTMT 4
Блокировать передатчик.
- #define UEM_MCODE_UBTMT 5
Разблокировать передатчик.
- #define UEM_MCODE_BRTF 6
Блокировать признак неисправности ОУ.
- #define UEM_MCODE_UBRTF 7
Разблокировать признак неисправности ОУ.
- #define UEM_MCODE_RESETRT 8
Установить ОУ в исходное состояние.
- #define UEM_MCODE_TXVECT 16
Передать векторное слово.
- #define UEM_MCODE_SYNCHRO_D 17
Синхронизация (с СД).
- #define UEM_MCODE_TXLCMD 18
Передать последнюю команду.
- #define UEM_MCODE_TXBIT 19
Передать слово ВСК ОУ.
- #define UEM_MCODE_BTMT_I 20
Блокировать i-й передатчик.
- #define UEM_MCODE_UBTMT_I 21
Разблокировать i-й передатчик.

Признаки ответного слова

В соответствии с [1].

- #define UEM_RTFAIL 0x0001
НОУ - Неисправность оконечного устройства.
- #define UEM_DBCA 0x0002
ПУИ - Принято управление интерфейсом.
- #define UEM_ABFAIL 0x0004
НА - Неисправность абонента.
- #define UEM_ABBUSY 0x0008
АЗ - Абонент занят.
- #define UEM_BCCA 0x0010
ПГК - Принята групповая команда.

- **#define UEM_RSV14** 0x0020
Резервный бит 14.
 - **#define UEM_RSV13** 0x0040
Резервный бит 13.
 - **#define UEM_RSV12** 0x0080
Резервный бит 12.
 - **#define UEM_SERVRQ** 0x0100
30 - Запрос обслуживания.
 - **#define UEM_ZERO** 0x0200
Передача ОС.
 - **#define UEM_MSGERR** 0x0400
ОШС - Ошибка в сообщении.
-

Подробное описание

Функции формирования и разбора командных и ответных слов.

Функции данного раздела служат для сборки командных и ответных слов из полей и разбора на поля.

Для этих целей также можно использовать структуры данных из unmuem_struct.h [5, 6]

Макросы

#define UEM_RTADDR_MIN 0

Минимальное значение адреса ОУ.

#define UEM_RTADDR_MAX 30

Максимальное значение адреса ОУ.

#define UEM_RTADDR_BRCST 31

Адрес для групповых сообщений.

#define UEM_RTADDR_MAX_EXT 31

Максимальное значение адреса ОУ в сетях с запретом групповых сообщений.

#define UEM_RT_RX 0

Признак передача/прием: прием.

#define UEM_RT_TX 1

Признак передача/прием: передача.

#define UEM_SADDR_MIN 1

Минимальное значение подадреса.

#define UEM_SADDR_MAX 30

Максимальное значение подадреса.

#define UEM_SADDR_CONV_LOOPBACK 30

Традиционный подадрес для тестовой петли.

#define UEM_SADDR_MIN_EXT 0

Минимальное значение подадреса в сетях с запретом команд управления.

#define UEM_SADDR_MAX_EXT 31

Максимальное значение подадреса в сетях с запретом команд управления.

#define UEM_MODE_0 0

Код режима 0.

#define UEM_MODE_31 31

Код режима 31.

#define UEM_MCODE_ADBC 0

Принять управление интерфейсом.

#define UEM_MCODE_SYNCHRO 1

Синхронизация.

#define UEM_MCODE_TXSTATUS 2

Передать ОС.

#define UEM_MCODE_BSELFTEST 3

Начать самоконтроль.

#define UEM_MCODE_BTMT 4

Блокировать передатчик.

#define UEM_MCODE_UBTMT 5

Разблокировать передатчик.

#define UEM_MCODE_BRTF 6

Блокировать признак неисправности ОУ.

#define UEM_MCODE_UBRTF 7

Разблокировать признак неисправности ОУ.

#define UEM_MCODE_RESETRT 8

Установить ОУ в исходное состояние.

#define UEM_MCODE_TXVECT 16

Передать векторное слово.

#define UEM_MCODE_SYNCHRO_D 17

Синхронизация (с СД).

#define UEM_MCODE_TXLCMD 18

Передать последнюю команду.

#define UEM_MCODE_TXBIT 19

Передать слово ВСК ОУ.

#define UEM_MCODE_BTMT_I 20

Блокировать i-й передатчик.

#define UEM_MCODE_UBTMT_I 21

Разблокировать i-й передатчик.

#define UEM_RTFAIL 0x0001

НОУ - Неисправность оконечного устройства.

#define UEM_DBCA 0x0002

ПУИ - Принято управление интерфейсом.

#define UEM_ABFAIL 0x0004

НА - Неисправность абонента.

#define UEM_ABBUSY 0x0008

АЗ - Абонент занят.

#define UEM_BCCA 0x0010

ПГК - Принята групповая команда.

#define UEM_RSV14 0x0020

Резервный бит 14.

#define UEM_RSV13 0x0040

Резервный бит 13.

#define UEM_RSV12 0x0080

Резервный бит 12.

#define UEM_SERVRQ 0x0100

ЗО - Запрос обслуживания.

#define UEM_ZERO 0x0200

Передача ОС.

#define UEM_MSGERR 0x0400

ОШС - Ошибка в сообщении.

Функции

UEM_WORD uem_command_word (UEM_WORD *rt*, UEM_BOOL *tx*, UEM_WORD *sa*, UEM_WORD *ndatawords*)

Формирование командного слова

Аргументы:

in	<i>rt</i>	Адрес ОУ.
in	<i>tx</i>	Разряд Передача(1)/Прием(0).
in	<i>sa</i>	Подадрес.
in	<i>ndatawords</i>	Число слов данных.

Возвращает:

Командное слово.

Для команд управления в **sa** следует указывать режим управления, а в **ndatawords** - код команды управления.

См. также:

Вспомогательные константы, Коды команд управления.

void uem_command_word_parse (UEM_WORD *cw*, UEM_WORD * *rt*, UEM_BOOL * *tx*, UEM_WORD * *sa*, UEM_WORD * *ndatawords*)

Разбор командного слова

Аргументы:

in	<i>cw</i>	Командное слово.
out	<i>rt</i>	Адрес ОУ.
out	<i>tx</i>	Разряд Передача(1)/Прием(0).
out	<i>sa</i>	Подадрес.
out	<i>ndatawords</i>	Число слов данных.

Возвращает:

Функция не возвращает значения.

UEM_WORD uem_status_word (UEM_WORD *rt*, UEM_WORD *status_bits*)

Формирование ответного слова.

Аргументы:

in	<i>rt</i>	Адрес ОУ.
in	<i>status_bits</i>	Признаки ответного слова. Объединение (по) констант Признаков ответного слова.

Возвращает:

Ответное слово.

void uem_status_word_parse (UEM_WORD *sw*, UEM_WORD * *rt*, UEM_WORD * *status_bits*)

Разбор ответного слова.

Аргументы:

in	<i>sw</i>	Ответное слово.
out	<i>rt</i>	Адрес ОУ.
out	<i>status_bits</i>	Признаки ответного слова. Объединение (по) констант Признаков ответного слова.

Возвращает:

Функция не возвращает значения.

Структуры данных

Структура UEM_BM_MESSAGE

Разобранное сообщение МШ.
#include <uem.h>

Поля данных

- **UEM_CMD_SEG cs**
Командный сегмент, включая формат сообщения и селектор шины.
- **UEM_BOOL rs1_pr**
Признак наличия ответного сегмента 1.
- **UEM_BOOL rs2_pr**
Признак наличия ответного сегмента 2.
- **UEM_RESP_SEG rs1**
Ответный сегмент 1.
- **UEM_RESP_SEG rs2**
Ответный сегмент 2.
- **UEM_SEGMENT_DESCR msg_d**
Описатель сообщения в целом.
- **UEM_SEGMENT_DESCR cs_d**
Описатель командного сегмента.
- **UEM_SEGMENT_DESCR rs1_d**
Описатель ответного сегмента 1.
- **UEM_SEGMENT_DESCR rs2_d**
Описатель ответного сегмента 2.
- **UEM_ERROR_FLAGS errors**
Строка бит - признаков ошибок распознавания сообщения.
- **UEM_BOOL p_pr**
Признак наличия предыдущего сообщения.
- **UEM_BOOL overlay**
Признак наложения на предыдущее сообщение.
- **UEM_TIME_TAG_LIN gap**
Пауза между предыдущим и данным сообщением.
- **UEM_DWORD lostp**
Количество пропущенных предыдущих сообщений.
- **UEM_RAW_BM_MESSAGE raw**
Неразобранное сообщение аппаратного формата.

Подробное описание

Разобранное сообщение МШ, представление в ОЗУ управляющей ПЭВМ, генерируется программным обеспечением виртуального МШ и содержит

- идентифицированные части (сегменты) сообщения,
- описатели сегментов сообщения, включая

- - время начала и конца,
- - признаки ошибок распознавания,
- - расположение сегмента в неразобранном сообщении в аппаратном формате,
- аналогичный описатель для сообщения в целом,
- признаки ошибок распознавания,
- пауза между предыдущим и данным сообщением,
- количество пропущенных предыдущих сообщений,
- неразобранное сообщение МШ в аппаратном формате.

Примечание: даже при сброшенных **rs1_pr** и **rs2_pr** признаки ошибки **rs1_d.errors** и **rs2_d.errors** могут содержать признак ошибки "Отсутствие ответа" (если формат сообщения предполагает ответ).

Поля

UEM_CMD_SEG cs

Командный сегмент, включая формат сообщения и селектор шины.

UEM_BOOL rs1_pr

Признак наличия ответного сегмента 1.

UEM_BOOL rs2_pr

Признак наличия ответного сегмента 2.

UEM_RESP_SEG rs1

Ответный сегмент 1.

Заполнено только если **rs1_p** выставлен в 1.

UEM_RESP_SEG rs2

Ответный сегмент 2.

Заполнено только если **rs2_p** выставлен в 1.

UEM_SEGMENT_DESCR msg_d

Описатель сообщения в целом.

UEM_SEGMENT_DESCR cs_d

Описатель командного сегмента.

UEM_SEGMENT_DESCR rs1_d

Описатель ответного сегмента 1.

Поле **errors** заполнено всегда, остальные поля - только если **rs1_p** выставлен в 1.

UEM_SEGMENT_DESCR rs2_d

Описатель ответного сегмента 2.

Поле **errors** заполнено всегда, остальные поля - только если **rs2_p** выставлен в 1.

UEM_ERROR_FLAGS errors

Строка бит - признаков ошибок распознавания сообщения.

UEM_BOOL p_pr

Признак наличия предыдущего сообщения.

UEM_BOOL overlay

Признак наложения на предыдущее сообщение.

UEM_TIME_TAG_LIN gap

Пауза между предыдущим и данным сообщением.

Заполняется только при $p_pr == 1$. При $overlay == 1$ это сдвиг начала данного сообщения по отношению к началу предыдущего. При $overlay == 0$ это пауза между концом предыдущего и началом данного сообщения, измеренная по ГОСТ [1], т.е. от последнего перепада предыдущего сообщения до первого перепада данного сообщения (+2 мкс по сравнению со "временем тишины", получаемой вычитанием моментов времени исчезновения и появления сигнала).

UEM_DWORD lostp

Количество пропущенных предыдущих сообщений.

Пропуск возможен при переполнении очереди сообщений или недостатка ОЗУ управляющей ПЭВМ. Для исключения пропуска приложение должно быстрее выбирать сообщения из очереди (см. `uem_bm_receive()`).

UEM_RAW_BM_MESSAGE raw

Неразобранное сообщение аппаратного формата.

Объявления и описания членов структуры находятся в файле:

`uem.h`

Структура UEM_CMD_SEG

Образ командного сегмента.
`#include <uem.h>`

Поля данных

- **UEM_CHANNEL ch**
Шина.
 - **UEM_FORMAT format**
Формат сообщения.
 - **UEM_WORD command1**
Командное слово.
 - **UEM_WORD command2**
Второе командное слово в сообщениях форматов 3 и 8.
 - **UEM_DATA data**
Слова данных.
-

Подробное описание

Образ командного сегмента.

Представление в ОЗУ управляющей ПЭВМ.

Использование полей **command1** и **command2** определяется форматом сообщения. Наличие, количество и состав слов данных определяется полем **data**.

В неформатных сообщениях (**UEM_UNF**) поля **command1** и **command2** не используются, сегмент состоит только из слов данных. Если необходимо, чтобы в сегменте были командные слова, следует установить в этих словах тип синхроимпульса командного слова (**UEM_SYNC_C**) при помощи функции **uem_cseg_sync_set()**, после создания командного сегмента.

Поля

UEM_CHANNEL ch

Шина.

UEM_FORMAT format

Формат сообщения.

UEM_WORD command1

Командное слово.

UEM_WORD command2

Второе командное слово в сообщениях форматов 3 и 8.

UEM_DATA data

Слова данных.

Объявления и описания членов структуры находятся в файле:

uem.h

Структура UEM_DATA

Блок слов данных.

```
#include <uem.h>
```

Поля данных

- **UEM_WORD ndata**
Число слов данных.
- **UEM_WORD data [62]**
Слова данных.

Подробное описание

Блок слов данных.

Поля

UEM_WORD ndata

Число слов данных.

UEM_WORD data[62]

Слова данных.

Примечание: Аппаратно УЭМ может формировать командные и ответные сегменты, содержащие до 62 слов данных.

Объявления и описания членов структуры находятся в файле:

uem.h

Структура UEM_RAW_BM_MESSAGE

Принятое сообщение в аппаратном формате.

```
#include <uem.h>
```

Поля данных

- **UEM_DWORD size**
Размер массива записей трассы.
- **unmuem_mt_data_t data [0]**
Массив записей трассы.

Подробное описание

Принятое сообщение в аппаратном формате генерируется аппаратурой монитора шины и состоит из последовательности записей трассы `unmuem_mt_data_t`. Определение этой структуры данных имеется в файле `unmuem_struct.h` [5, 6], а описание ее полей - в [2, 3].

Поля

UEM_DWORD size

Размер массива записей трассы.

unmuem_mt_data_t data[0]

Массив записей трассы.

Объявления и описания членов структуры находятся в файле:

`uem.h`

Структура UEM_RESP_SEG

Ответный сегмент.

```
#include <uem.h>
```

Поля данных

- **UEM_WORD status**
Ответное слово.
- **UEM_DATA data**
Слова данных.

Подробное описание

Ответный сегмент.

Представление в ОЗУ управляющей ПЭВМ.

Поля

UEM_WORD status

Ответное слово.

UEM_DATA data

Слова данных.

Объявления и описания членов структуры находятся в файле:

uem.h

Структура UEM_SEGMENT_DESCR

Описатель сегмента в мониторе шины.
#include <uem.h>

Поля данных

- **UEM_TIME_TAG start_time**
Время начала сегмента (время появления сигнала).
- **UEM_TIME_TAG end_time**
Время конца сегмента (время исчезновения сигнала).
- **UEM_ERROR_FLAGS errors**
Признаки обнаруженных ошибок.
- **ViUInt16 offset**
Смещение сегмента в UEM_RAW_BM_MESSAGE.
- **ViUInt16 dwoffset**
Смещение первого слова данных в UEM_RAW_BM_MESSAGE.
- **ViUInt16 endoffset**
Смещение первого слова за концом данного сегмента.
- **ViUInt16 size**
Размер сегмента в словах.

Подробное описание

Описатель сегмента в мониторе шины.

Поля

UEM_TIME_TAG start_time

Время начала сегмента (время появления сигнала).
Точность измерений $\pm 0,25$ мкс.

UEM_TIME_TAG end_time

Время конца сегмента (время исчезновения сигнала).
Точность измерений $\pm 0,25$ мкс.

UEM_ERROR_FLAGS errors

Признаки обнаруженных ошибок.

ViUInt16 offset

Смещение сегмента в UEM_RAW_BM_MESSAGE.

ViUInt16 dwoffset

Смещение первого слова данных в **UEM_RAW_BM_MESSAGE**.

ViUInt16 endoffset

Смещение первого слова за концом данного сегмента.

ViUInt16 size

Размер сегмента в словах.

Объявления и описания членов структуры находятся в файле:
uem.h

Объединение UEM_TIME_TAG

Формат метки времени.

```
#include <uem.h>
```

Поля данных

```
struct {
```

- **ViUInt64 quartas:22**
Число единицы метки времени, так называемых четвертей ; 0,25 мкс на единицу младшего разряда; диапазон 0 - 3 999 999 (охватывает 1 сек).
 - **ViUInt64 secs:17**
Число секунд; 1 сек на единицу младшего разряда; диапазон 0 - 86 399 (охватывает 1 сутки).
 - **ViUInt64 days:9**
Число суток; 1 сутки на единицу младшего разряда; диапазон 0 - 365 (охватывает 366 суток).
 - **ViUInt64 reserved:16**
Не используется и всегда содержит 0.
 - **} b**
Этот элемент объединения обеспечивает доступ к элементам метки времени - перечисленным полям.
 - **ViUInt64 i**
Этот элемент объединения обеспечивает манипулирование значением как единым целым.
-

Подробное описание

Формат метки времени.

Поля

ViUInt64 quartas

Число единиц метки времени, так называемых *четвертей* ; 0,25 мкс на единицу младшего разряда; диапазон 0 - 3 999 999 (охватывает 1 сек).

ViUInt64 secs

Число секунд; 1 сек на единицу младшего разряда; диапазон 0 - 86 399 (охватывает 1 сутки).

ViUInt64 days

Число суток; 1 сутки на единицу младшего разряда; диапазон 0 - 365 (охватывает 366 суток).

ViUInt64 reserved

Не используется и всегда содержит 0.

struct { ... } b

Этот элемент объединения обеспечивает доступ к элементам метки времени - перечисленным полям.

ViUInt64 i

Этот элемент объединения обеспечивает манипулирование значением как единым целым.

Объявления и описания членов объединения находятся в файле:

uem.h

Файлы

Файл uem.h

Универсальные электронные модули УЭМ-МК, МВ98.03. Расширенная библиотека функций. Файл заголовков функций.

```
#include "unmuem.h"  
#include "unmuem_struct.h"
```

Структуры данных

- union **UEM_TIME_TAG**
Формат метки времени.
- struct **UEM_DATA**
Блок слов данных.
- struct **UEM_CMD_SEG**
Образ командного сегмента.
- struct **UEM_RESP_SEG**
Ответный сегмент.
- struct **UEM_RAW_BM_MESSAGE**
Принятое сообщение в аппаратном формате.
- struct **UEM_SEGMENT_DESCR**
Описатель сегмента в мониторе шины.
- struct **UEM_BM_MESSAGE**
Разобранное сообщение МШ.

Макросы

- #define **UEM_WARN_OFFSET** (0x3FFC0B00L)
Начальный номер кодов предупреждений.
- #define **UEM_ERROR_OFFSET** (_VI_ERROR + UEM_WARN_OFFSET)
Начальный номер кодов ошибок.
- #define **UEM_ERROR_BAD_PARAM_VALUE** (UEM_ERROR_OFFSET + 0)
Недопустимое значение параметра.
- #define **UEM_ERROR_BAD_PARAM_VALUE_1** (UEM_ERROR_OFFSET + 1)
Недопустимое значение в параметре 1.
- #define **UEM_ERROR_BAD_PARAM_VALUE_2** (UEM_ERROR_OFFSET + 2)
Недопустимое значение в параметре 2.
- #define **UEM_ERROR_BAD_PARAM_VALUE_3** (UEM_ERROR_OFFSET + 3)
Недопустимое значение в параметре 3.
- #define **UEM_ERROR_BAD_PARAM_VALUE_4** (UEM_ERROR_OFFSET + 4)
Недопустимое значение в параметре 4.
- #define **UEM_ERROR_BAD_PARAM_VALUE_5** (UEM_ERROR_OFFSET + 5)
Недопустимое значение в параметре 5.
- #define **UEM_ERROR_BAD_PARAM_VALUE_6** (UEM_ERROR_OFFSET + 6)
Недопустимое значение в параметре 6.
- #define **UEM_ERROR_BAD_PARAM_VALUE_7** (UEM_ERROR_OFFSET + 7)
Недопустимое значение в параметре 7.
- #define **UEM_ERROR_BAD_PARAM_VALUE_8** (UEM_ERROR_OFFSET + 8)
Недопустимое значение в параметре 8.

- #define UEM_ERROR_BAD_PARAM_VALUE_9 (UEM_ERROR_OFFSET + 9)
Недопустимое значение в параметре 9.
- #define UEM_ERROR_BAD_PARAM_VALUE_10 (UEM_ERROR_OFFSET + 10)
Недопустимое значение в параметре 10.
- #define UEM_ERROR_INV_HANDLE (UEM_ERROR_OFFSET + 11)
Недействительный дескриптор.
- #define UEM_ERROR_INV_HANDLE_TYPE (UEM_ERROR_OFFSET + 12)
Неподходящий тип дескриптора.
- #define UEM_ERROR_NO_FREE_RAM (UEM_ERROR_OFFSET + 13)
Недостаточно ОЗУ УЭМ.
- #define UEM_ERROR_NO_HOST_MEM (UEM_ERROR_OFFSET + 14)
Недостаточно ОЗУ управляющей ПЭВМ.
- #define UEM_ERROR_NOT_CONNECTED (UEM_ERROR_OFFSET + 15)
Нет связи с устройством.
- #define UEM_ERROR_INPOOL (UEM_ERROR_OFFSET + 16)
Внутренняя ошибка менеджера памяти.
- #define UEM_ERROR_BCP_NINST (UEM_ERROR_OFFSET + 17)
Не установлена программа КИИ.
- #define UEM_ERROR_FORMAT_DISABLED (UEM_ERROR_OFFSET + 18)
Формат сообщения запрещен конфигурацией УЭМ.
- #define UEM_ERROR_FORMAT_X_MCODE (UEM_ERROR_OFFSET + 19)
Формат сообщения несовместим с командой управления.
- #define UEM_ERROR_NOT_APPLICABLE (UEM_ERROR_OFFSET + 20)
Действие не применимо к объекту.
- #define UEM_ERROR_ADDRESS_OUT_OF_RANGE (UEM_ERROR_OFFSET + 21)
Адрес вне допустимого диапазона.
- #define UEM_ERROR_NUMBER_OUT_OF_RANGE (UEM_ERROR_OFFSET + 23)
Номер вне допустимого диапазона.
- #define UEM_ERROR_BAD_TIMEOUT (UEM_ERROR_OFFSET + 24)
Недопустимое значение таймаута.
- #define UEM_ERROR_BAD_OVERLAY_SOURCE (UEM_ERROR_OFFSET + 25)
Недопустимые исходные сегменты для наложения.
- #define UEM_ERROR_WRONG_LOCATION (UEM_ERROR_OFFSET + 26)
Объект расположен не в том устройстве.
- #define UEM_ERROR_TOO_MANY_DATAWORDS (UEM_ERROR_OFFSET + 27)
Слишком много слов данных.
- #define UEM_ERROR_MAX_SIZE_EXCEED (UEM_ERROR_OFFSET + 28)
Превышен максимальный размер.
- #define UEM_ERROR_NO_FRAME_APPEND (UEM_ERROR_OFFSET + 29)
Не добавлен кадр.
- #define UEM_ERROR_IN_USE (UEM_ERROR_OFFSET + 30)
Устройство или объект используются.
- #define UEM_ERROR_THREAD_FAULT (UEM_ERROR_OFFSET + 31)
Ошибки в работе служебной нити.
- #define UEM_ERROR_BM_INTERNAL_BUFFER_OVERFLOW (UEM_ERROR_OFFSET + 32)
Переполнение внутреннего буфера МИИ.

- #define **UEM_ERROR_INC_RESP** (UEM_ERROR_OFFSET + 33)
Несовместимый ответный сегмент.
- #define **UEM_WARN_NO_NEXT_MESSAGE** (UEM_WARN_OFFSET + 0)
Нет следующего сообщения (в буфере MII).
- #define **UEM_WARN_JUST_IN_STATE** (UEM_WARN_OFFSET + 1)
Устройство уже в нужном состоянии; никаких действий не производится.
- #define **UEM_TMTA_DIS** UEMi_MAKE_PARAMID (0, 2, 2)
Запрет работы передатчика шины А.
- #define **UEM_TMTB_DIS** UEMi_MAKE_PARAMID (0, 3, 3)
Запрет работы передатчика шины Б.
- #define **UEM_RCVA_DIS** UEMi_MAKE_PARAMID (0, 4, 4)
Запрет работы приемника шины А.
- #define **UEM_RCVB_DIS** UEMi_MAKE_PARAMID (0, 5, 5)
Запрет работы приемника шины Б.
- #define **UEM_SYNC_IN_1_ENA** UEMi_MAKE_PARAMID (0, 8, 8)
Разрешение входной синхронизации 1.
- #define **UEM_SYNC_IN_2_ENA** UEMi_MAKE_PARAMID (0, 9, 9)
Разрешение входной синхронизации 2.
- #define **UEM_SYNC_OUT_1_ENA** UEMi_MAKE_PARAMID (0, 10, 10)
Разрешение выходной синхронизации 1.
- #define **UEM_SYNC_OUT_2_ENA** UEMi_MAKE_PARAMID (0, 11, 11)
Разрешение выходной синхронизации 2.
- #define **UEM_BRCST_DIS** UEMi_MAKE_PARAMID(0, 12, 12)
Запрет групповых сообщений.
- #define **UEM_SYNC_IN_1_INTGEN** UEMi_MAKE_PARAMID(0, 13, 13)
Разрешение внутренней эмуляции сигнала входной синхронизации 1.
- #define **UEM_ERR_INJ_DIS** UEMi_MAKE_PARAMID (0, 14, 14)
Запрет внесения ошибок кодирования в передаваемую в МКПД информацию для КШ и ОУ.
- #define **UEM_SYNC_IN_2_INTGEN** UEMi_MAKE_PARAMID(0, 29, 29)
Разрешение внутренней эмуляции сигнала входной синхронизации 2.
- #define **UEM_TMT_RES** UEMi_MAKE_PARAMID (5, 4, 4)
Сброс настроек передатчиков (только запись).
- #define **UEM_SYNC_IN_1_SET** UEMi_MAKE_PARAMID (5, 10, 10)
Программная генерация сигнала входной синхронизации 1 (только запись).
- #define **UEM_SYNC_IN_2_SET** UEMi_MAKE_PARAMID (5, 11, 11)
Программная генерация сигнала входной синхронизации 2 (только запись).
- #define **UEM_SYNC_OUT_1_SET** UEMi_MAKE_PARAMID (5, 12, 12)
Программная генерация сигнала выходной синхронизации 1 (только запись).
- #define **UEM_SYNC_OUT_2_SET** UEMi_MAKE_PARAMID (5, 13, 13)
Программная генерация сигнала выходной синхронизации 2 (только запись).
- #define **UEM_DB_ACT** UEMi_MAKE_PARAMID (5, 31, 31)
Обнаружена передача данных по шине (только чтение).
- #define **UEM_TXA_RFT** UEMi_MAKE_PARAMID (6, 31, 16)
Управление длительностью фронта и среза при передаче в шину А.
- #define **UEM_TXA_VPP** UEMi_MAKE_PARAMID (6, 15, 0)
Управление размахом сигнала при передаче в шину А.

- #define **UEM_TXB_RFT** UEMi_MAKE_PARAMID (7, 31, 16)
Управление длительностью фронта и среза при передаче в шину Б.
- #define **UEM_TXB_VPP** UEMi_MAKE_PARAMID (7, 15, 0)
Управление размахом сигнала при передаче в шину Б.
- #define **UEM_MC_DIS** UEMi_MAKE_PARAMID (0x0C, 0, 0)
Запрет команд управления.
- #define **UEM_BTMT_DIS** UEMi_MAKE_PARAMID (0x0C, 1, 1)
Запрет блокирования и разблокирования передатчиков ОУ.
- #define **UEM_BRTF_DIS** UEMi_MAKE_PARAMID (0x0C, 2, 2)
Запрет блокирования и разблокирования признака неисправности ОУ.
- #define **UEM_SYNC2_VRTA** UEMi_MAKE_PARAMID (0x0C, 7, 3)
Номер ОУ – условие выработки сигнала выходной синхронизации 2.
- #define **UEM_SYNC1_C_D_** UEMi_MAKE_PARAMID (0x0C, 8, 8)
Командное/ответное (1) слово или слово данных (0) – условие выработки сигнала выходной синхронизации 1.
- #define **UEM_SYNC1_ERR** UEMi_MAKE_PARAMID (0x0C, 9, 9)
Наличие ошибок в слове – условие выработки сигнала выходной синхронизации 1.
- #define **UEM_SYNC1_GAPB** UEMi_MAKE_PARAMID (0x0C, 10, 10)
Наличие паузы перед словом - условие выработки сигнала выходной синхронизации 1.
- #define **UEM_SYNC1_CH** UEMi_MAKE_PARAMID (0x0C, 12, 11)
Слово передается по указанной шине - условие выработки сигнала выходной синхронизации 1.
- #define **UEM_IST2** UEMi_MAKE_PARAMID (0x0C, 31, 16)
Период внутренней генерации сигнала входной синхронизации 2.
- #define **UEM_IST1** UEMi_MAKE_PARAMID (8, 31, 16)
Период внутренней генерации сигнала входной синхронизации 1.
- #define **UEM_BM_WORD_PATTERN** UEMi_MAKE_PARAMID (0x0F, 15, 0)
Шаблон (значение) слова - условие выработки сигнала выходной синхронизации 1.
- #define **UEM_BM_WORD_MASK** UEMi_MAKE_PARAMID (0x0F, 31, 16)
Маска побитного сравнения слова с шаблоном - условие выработки сигнала выходной синхронизации 1.
- #define **UEM_MIN_T1_DEFAULT** (4*4)
Минимальная пауза между командным и ответным сегментами (min t1), значение по умолчанию.
- #define **UEM_MIN_T2_DEFAULT** (4*4)
Минимальная пауза между сообщениями (min t2), значение по умолчанию.
- #define **UEM_RTMO_DEFAULT** (14*4)
Таймаут ответа ОУ, значение по умолчанию.
- #define **UEM_LIB_REV** 0x0100
Номер версии библиотеки.

Граничные значения изменения количества разрядов.

Эти константы задают граничные значения аргумента **error_pos** в функциях **uem_cseg_error_set()**, **uem_response_error_set()**, когда в аргументе **error_type** указано **UEM_ERRT_WRONG_BITCOUNT**.

#define UEM_BITCOUNT_CHANGE_MIN (-3)

Минимальное приращение количества разрядов.

#define UEM_BITCOUNT_CHANGE_MAX (+3)

Максимальное приращение количества разрядов.

Граничные значения позиции ошибки.

Эти константы задают граничные значения аргумента **error_pos** для некоторых значений аргумента **error_type** в функциях **uem_cseg_error_set()**, **uem_response_error_set()**.

```
#define UEM_BIPHASE_POS_MIN 4
Минимальная позиция при error_type, равном UEM_ERRT_BAD_BIPHASE_ZERO,
UEM_ERRT_BAD_BIPHASE_POS или UEM_ERRT_BAD_BIPHASE_NEG.
#define UEM_BIPHASE_POS_MAX 20
Максимальная позиция при error_type, равном UEM_ERRT_BAD_BIPHASE_ZERO,
UEM_ERRT_BAD_BIPHASE_POS или UEM_ERRT_BAD_BIPHASE_NEG.
#define UEM_SHIFT_POS_MIN 0
Минимальная позиция при error_type, равном UEM_ERRT_SHIFT_EDGE.
#define UEM_SHIFT_POS_MAX 40
Максимальная позиция при error_type, равном UEM_ERRT_SHIFT_EDGE.
```

Граничные значения величины сдвига.

Эти константы задают граничные значения для аргумента **error_param** в функциях **uem_cseg_error_set()**, **uem_response_error_set()**, когда аргумент **error_type** равен **UEM_ERRT_SHIFT_EDGE**. Величина сдвига указывается в единицах по 10 нс. Отрицательные значения обозначают сдвиг влево, положительные - вправо.

```
#define UEM_SHIFT_LENGTH_MIN (-25)
Максимальная величина сдвига влево.
#define UEM_SHIFT_LENGTH_MAX (+25)
Максимальная величина сдвига вправо.
```

Значения аргументов по умолчанию.

Эти константы задают значения аргументов функций **uem_cseg_error_set()**, **uem_response_error_set()** по умолчанию, соответствуют отсутствию внесения ошибок.

```
#define UEM_ERROR_TYPE_DEFAULT 0
Значение по умолчанию для error_type.
#define UEM_ERROR_POS_DEFAULT 0
Значение по умолчанию для error_pos.
#define UEM_ERROR_PARAM_DEFAULT 0
Значение по умолчанию для error_param.
```

Селектор сеанса.

```
#define UEM_SEL_UNMUEM 1
Сеанс низкоуровневого драйвера УЭМ иптиет.
#define UEM_SEL_UNMBASE 2
Сеанс драйвера носителя мезонинов иптbase.
#define UEM_SEL_UNBASE_INT 3
"Внутренний" сеанс драйвера носителя мезонинов иптbase.
```

Значения параметра UEM_TMTA_DIS.

```
#define UEM_TMTA_DISABLED 1
Передатчик шины А отключен.
#define UEM_TMTA_ENABLED 0
Передатчик шины А включен.
#define UEM_TMTA_DEFAULT (UEM_TMTA_ENABLED)
По умолчанию: Передатчик шины А включен.
```

Значения параметра UEM_TMTB_DIS.

```
#define UEM_TMTB_DISABLED 1
Передатчик шины Б отключен.
#define UEM_TMTB_ENABLED 0
Передатчик шины Б включен.
```

```
#define UEM_TMTB_DEFAULT (UEM_TMTB_ENABLED)
```

По умолчанию: Передатчик шины Б включен.

Значения параметра UEM_RCVA_DIS.

```
#define UEM_RCVA_DISABLED 1
```

Приемник шины А отключен.

```
#define UEM_RCVA_ENABLED 0
```

Приемник шины А включен.

```
#define UEM_RCVA_DEFAULT (UEM_RCVA_ENABLED)
```

По умолчанию: Приемник шины А включен.

Значения параметра UEM_RCVB_DIS.

```
#define UEM_RCVB_DISABLED 1
```

Приемник шины Б отключен.

```
#define UEM_RCVB_ENABLED 0
```

Приемник шины Б включен.

```
#define UEM_RCVB_DEFAULT (UEM_RCVB_ENABLED)
```

По умолчанию: Приемник шины Б включен.

Значения параметра UEM_SYNC_IN_1_ENA.

```
#define UEM_SYNC_IN_1_DISABLED 0
```

Обработка входного сигнала синхронизации 1 запрещена.

```
#define UEM_SYNC_IN_1_ENABLED 1
```

Обработка входного сигнала синхронизации 1 разрешена.

```
#define UEM_SYNC_IN_1_DEFAULT (UEM_SYNC_IN_1_DISABLED)
```

По умолчанию: Обработка входного сигнала синхронизации 1 запрещена.

Значения параметра UEM_SYNC_IN_2_ENA.

```
#define UEM_SYNC_IN_2_DISABLED 0
```

Обработка входного сигнала синхронизации 2 запрещена.

```
#define UEM_SYNC_IN_2_ENABLED 1
```

Обработка входного сигнала синхронизации 2 разрешена.

```
#define UEM_SYNC_IN_2_DEFAULT (UEM_SYNC_IN_2_DISABLED)
```

По умолчанию: Обработка входного сигнала синхронизации 2 запрещена.

Значения параметра UEM_SYNC_OUT_1_ENA.

```
#define UEM_SYNC_OUT_1_DISABLED 0
```

Формирование выходного сигнала синхронизации 1 запрещено.

```
#define UEM_SYNC_OUT_1_ENABLED 1
```

Формирование выходного сигнала синхронизации 1 разрешено.

```
#define UEM_SYNC_OUT_1_DEFAULT (UEM_SYNC_OUT_1_DISABLED)
```

По умолчанию: Формирование выходного сигнала синхронизации 1 запрещено.

Значения параметра UEM_SYNC_OUT_2_ENA.

```
#define UEM_SYNC_OUT_2_DISABLED 0
```

Формирование выходного сигнала синхронизации 2 запрещено.

```
#define UEM_SYNC_OUT_2_ENABLED 1
```

Формирование выходного сигнала синхронизации 2 разрешено.

```
#define UEM_SYNC_OUT_2_DEFAULT (UEM_SYNC_OUT_2_DISABLED)
```

По умолчанию: Формирование выходного сигнала синхронизации 2 запрещено.

Значения параметра UEM_BRCST_DIS.

```
#define UEM_BRCST_DISABLED 1
```

Групповые сообщения запрещены.

```
#define UEM_BRCST_ENABLED 0
Групповые сообщения разрешены.
#define UEM_BRCST_DEFAULT (UEM_BRCST_ENABLED)
По умолчанию: Групповые сообщения разрешены.
```

Значения параметра UEM_SYNC_IN_1_INTGEN.

```
#define UEM_SYNC_IN_1_INTGEN_DISABLED 0
Внутренняя генерация запрещена.
#define UEM_SYNC_IN_1_INTGEN_ENABLED 1
Внутренняя генерация разрешена.
#define UEM_SYNC_IN_1_INTGEN_DEFAULT (UEM_SYNC_IN_1_INTGEN_DISABLED)
По умолчанию: запрещена.
```

Значения параметра UEM_SYNC_IN_2_INTGEN.

```
#define UEM_SYNC_IN_2_INTGEN_DISABLED 0
Внутренняя генерация запрещена.
#define UEM_SYNC_IN_2_INTGEN_ENABLED 1
Внутренняя генерация разрешена.
#define UEM_SYNC_IN_2_INTGEN_DEFAULT (UEM_SYNC_IN_2_INTGEN_DISABLED)
По умолчанию: запрещена.
```

Значения параметра UEM_ERR_INJ_DIS.

```
#define UEM_ERR_INJ_DISABLED 1
Внесение ошибок кодирования запрещено.
#define UEM_ERR_INJ_ENABLED 0
Внесение ошибок кодирования разрешено.
#define UEM_ERR_INJ_DEFAULT (UEM_ERR_INJ_ENABLED)
По умолчанию: Внесение ошибок кодирования разрешено.
```

Значение UEM_SET.

Значение **UEM_SET**, тождественно равно 1, используется с параметрами, фактически являющимися командами, вызывающими определенные действия аппаратуры УЭМ. Такие параметры доступны только по записи, причем действенной является именно запись значения 1. Такие параметры не имеют и не требуют установки значений по умолчанию.

```
#define UEM_SET 1
Установка параметра в 1.
```

Значения параметра UEM_DB_ACT.

Данный параметр доступен только по чтению, он не имеет и не требует установки значения по умолчанию.

```
#define UEM_DB_INACTIVE 0
Нет активности на шине данных.
#define UEM_DB_ACTIVE 1
Обнаружена активность на шине данных.
```

Значения параметров UEM_TXA_RFT, UEM_TXB_RFT.

```
#define UEM_RFT_MIN 0
Минимальное значение.
#define UEM_RFT_MAX 31
Максимальное значение.
#define UEM_RFT_SINE 32
Специальное значение. Включает формирование синусоидального сигнала.
```



```
#define UEM_RFT_DEFAULT 18
```

Значение по умолчанию (18).

Значения параметров UEM_TXA_VPP, UEM_TXB_VPP.

```
#define UEM_VPP_MIN 0x00E
```

Минимальное значение.

```
#define UEM_VPP_MAX 0x01FF
```

Максимальное значение.

```
#define UEM_VPP_DEFAULT 0x0136
```

Значение по умолчанию (0x0136).

Значения параметра UEM_MC_DIS.

```
#define UEM_MC_DISABLED 1
```

Команды управления запрещены.

```
#define UEM_MC_ENABLED 0
```

Команды управления разрешены.

```
#define UEM_MC_DEFAULT (UEM_MC_ENABLED)
```

По умолчанию: Команды управления разрешены.

Значения параметра UEM_BTMT_DIS.

```
#define UEM_BTMT_DISABLED 1
```

Блокирование и разблокирование передатчиков ОУ запрещено.

```
#define UEM_BTMT_ENABLED 0
```

Блокирование и разблокирование передатчиков ОУ разрешено.

```
#define UEM_BTMT_DEFAULT (UEM_BTMT_ENABLED)
```

По умолчанию: Блокирование и разблокирование передатчиков ОУ разрешено.

Значения параметра UEM_BRTF_DIS.

```
#define UEM_BRTF_DISABLED 1
```

Блокирование и разблокирование признака неисправности ОУ запрещено.

```
#define UEM_BRTF_ENABLED 0
```

Блокирование и разблокирование признака неисправности ОУ разрешено.

```
#define UEM_BRTF_DEFAULT (UEM_BRTF_ENABLED)
```

По умолчанию: Блокирование и разблокирование признака неисправности ОУ разрешено.

Значения параметра UEM_SYNC2_VRTA.

```
#define UEM_SYNC2_VRTA_MIN 0
```

Минимальное значение.

```
#define UEM_SYNC2_VRTA_MAX 31
```

Максимальное значение.

```
#define UEM_SYNC2_VRTA_DEFAULT 0
```

Значение по умолчанию.

Значения параметра UEM_SYNC1_C_D_.

```
#define UEM_SYNC1_ON_COMMAND 1
```

Синхроимпульс командного/ответного слова.

```
#define UEM_SYNC1_ON_DATA 0
```

Синхроимпульс слова данных.

```
#define UEM_SYNC1_C_D_DEFAULT (UEM_SYNC1_ON_DATA)
```

По умолчанию: Синхроимпульс слова данных.

Значения параметра UEM_SYNC1_ERR.

```
#define UEM_SYNC1_ON_ERROR 1
```

При наличии ошибки в слове.

```
#define UEM_SYNC1_ON_NO_ERROR 0
При отсутствии ошибки в слове.
#define UEM_SYNC1_ERR_DEFAULT (UEM_SYNC1_ON_NO_ERROR)
По умолчанию: При отсутствии ошибки в слове.
```

Значения параметра UEM_SYNC1_GAPB.

```
#define UEM_SYNC1_ON_GAPB 1
При наличии паузы перед словом.
#define UEM_SYNC1_ON_NO_GAPB 0
При отсутствии паузы перед словом.
#define UEM_SYNC1_GAPB_DEFAULT (UEM_SYNC1_ON_NO_GAPB)
По умолчанию: При отсутствии паузы перед словом.
```

Значения параметра UEM_SYNC1_CH.

```
#define UEM_SYNC1_CH_A 0
Слово передается по шине А.
#define UEM_SYNC1_CH_B 2
Слово передается по шине Б.
#define UEM_SYNC1_ACH 1
Слово передается по любой шине.
#define UEM_SYNC1_CH_DEFAULT (UEM_SYNC1_CH_A)
Значение UEM_SYNC1_CH по умолчанию: по шине А.
```

Значения параметров UEM_IST1, UEM_IST2.

```
#define UEM_IST_MIN 1
Минимальное значение.
#define UEM_IST_MAX 65536
Максимальное значение.
#define UEM_IST_DEFAULT (UEM_IST_MAX)
Значение по умолчанию.
```

Значения параметров UEM_BM_WORD_PATTERN, UEM_BM_WORD_MASK.

```
#define UEM_BM_WORD_MIN 0x0000
Минимальное значение.
#define UEM_BM_WORD_MAX 0xFFFF
Максимальное значение.
#define UEM_BM_WORD_DEFAULT (UEM_BM_WORD_MIN)
Значение по умолчанию.
```

Более mnemonic обозначения форматов сообщений

```
#define UEM_BCRT UEM_F1
КШОУ.
#define UEM_RTBC UEM_F2
ОУКШ.
#define UEM_RTRT UEM_F3
ОУОУ.
#define UEM_MC UEM_F4
Команда управления.
#define UEM_MCRTBC UEM_F5
Команда управления со словом данных, передаваемым от ОУ к КШ.
#define UEM_MCBCRT UEM_F6
Команда управления со словом данных, передаваемым от КШ к ОУ.
```

```
#define UEM_BCRTb UEM_F7
КШОУ ГРУППОВОЕ.
#define UEM_RTRTb UEM_F8
ОУОУ ГРУППОВОЕ.
#define UEM_MCb UEM_F9
Команда управления групповая.
#define UEM_MCBCRTb UEM_F10
Команда управления групповая со словом данных, передаваемым от КШ к ОУ.
Число слов данных
#define UEM_NDATA_MIN 0
Число слов данных - минимальное значение.
#define UEM_NDATA_MAX 62
Число слова данных - максимальное значение.
#define UEM_NDATA_BY_CW 63
Число слов данных определяется командным словом.
```

Признаки ошибок распознавания сообщения.

*Комбинации перечисленных констант (по |) определяют содержание элементов типа **UEM_ERROR_FLAGS**.*

```
#define UEM_ERRF_ERROR 1
Наличие любой ошибки (суммарный флаг).
#define UEM_ERRF_ENCODING (1<<1)
Наличие ошибки кодирования слов (суммарный флаг).
#define UEM_ERRF_FORMAT (1<<2)
Наличие нарушения формата, ошибки состава сообщения (суммарный флаг).
#define UEM_ERRF_MINGAP (1<<3)
Временной интервал меньше допустимого.
#define UEM_ERRF_NO_RESPONSE (1<<4)
Отсутствие ответа.
#define UEM_ERRF_SYNC_TYPE (1<<5)
Неверный тип синхроимпульса.
#define UEM_ERRF_MISSING_CWSW (1<<6)
Отсутствует командное или ответное слово.
#define UEM_ERRF_EXTRA_CWSW (1<<7)
Лишнее командное или ответное слово.
#define UEM_ERRF_MISSING_DW (1<<8)
Недостаточно слов данных.
#define UEM_ERRF_EXTRA_DW (1<<9)
Лишние слова данных.
#define UEM_ERRF_INCORRECT_RTN (1<<10)
Некорректный адрес ОУ.
#define UEM_ERRF_RTRT_FORMAT (1<<11)
Ошибка формата ОУОУ (одинаковые адреса ОУ, несовпадение числа СД).
#define UEM_ERRF_INC_MODE_CODE (1<<12)
Некорректная команда управления.
#define UEM_ERRF_FORMAT_MC (1<<13)
Ошибка формата команды управления.
#define UEM_ERRF_GAPN (1<<20)
Недостовверная информация (сигнал) во время паузы перед словом.
```

```
#define UEM_ERRF_PARITY (1<<21)
Ошибка четности.
#define UEM_ERRF_LESS_BITS (1<<22)
Укороченное слово.
#define UEM_ERRF_MORE_BITS (1<<23)
Удлиненное слово.
#define UEM_ERRF_ENC (1<<24)
Ошибка бифазного кодирования.
#define UEM_ERRF_DT (1<<30)
Несоблюдение минимальной паузы перед словом, по данным аппаратного декодера.
#define UEM_ERRF_ENCODING2 (1<<31)
Наличие любой ошибки кодирования слов (суммарный флаг, по данным аппаратного декодера).
```

Флаги отсчета паузы.

Пауза отсчитывается от одного из определенных событий.

Флаги отсчета паузы определяют выбор события.

*Пауза может отсчитываться от начала либо конца предшествующего сегмента, переданного КШ либо другим абонентом (ОУ), по той же либо по альтернативной шине. Выбор каждой из альтернатив управляется отдельным флагом. Для начала отсчета паузы также может использоваться сигнал внешней синхронизации **sync_in_1**. Поступление этого сигнала используется вместо события начала или конца сегмента, переданного КШ. Если же задан отсчет паузы от начала (или конца) сегмента, переданного другим абонентом, то учитываются оба события - начало (или конец) сегмента и поступление сигнала. При задании отсчета паузы от начала (или конца) сегмента другого абонента, или от поступления сигнала, или обоих, то есть - от внешних событий, которые теоретически могут и не наступить, используется таймаут, который задается в функции **uem_cseg_gap_set()**, и по истечении которого отсчет паузы начинается без дальнейшего ожидания внешних событий.*

```
#define UEM_CSEG_GAP_FROM_START 0x20
Отсчет паузы от начала предыдущего сегмента.
#define UEM_CSEG_GAP_FROM_END 0
Отсчет паузы от конца предыдущего сегмента.
#define UEM_CSEG_GAP_ESYNC 0x10
Отсчет паузы после сигнала внешней синхронизации (только для КШ).
#define UEM_CSEG_GAP_ALT_BUS 0x08
Отсчет паузы от сегмента по альтернативной шине (только для КШ).
#define UEM_CSEG_GAP_THIS_BUS 0
Отсчет паузы от сегмента по этой же шине (только для КШ).
#define UEM_CSEG_GAP_ALT_AB 0x04
Отсчет паузы от сегмента другого абонента (ОУ) (только для КШ).
#define UEM_CSEG_GAP_THIS_AB 0
Отсчет паузы от собственного сегмента (только для КШ).
#define UEM_CSEG_GAP_DEFAULT_FLAGS (UEM_CSEG_GAP_FROM_START)
Стандартный набор флагов отсчета паузы.
```

Диапазоны значений параметров отсчета паузы.

```
#define UEM_CSEG_GAP_MIN 0
Минимальное значение паузы.
#define UEM_CSEG_GAP_MAX 65535
Максимальное значение паузы.
```

```
#define UEM_CSEG_GAP_DEFAULT_VALUE 0
Значение паузы по умолчанию.
#define UEM_CSEG_GAP_TIMEOUT_MIN 0
Минимальное значение таймаута отсчета паузы.
#define UEM_CSEG_GAP_TIMEOUT_MAX 1023
Максимальное значение таймаута отсчета паузы.
#define UEM_CSEG_GAP_DEFAULT_TIMEOUT 0
Таймаут отсчета паузы по умолчанию.
```

Константы для числа повторов кадра.

*Данные константы могут использоваться в аргументе **repeat_count** функции **uem_bcp_append_frame()**.*

```
#define UEM_FRAME_REPEAT_UNLIM 0
Неограниченное число повторов кадра.
#define UEM_FRAME_REPEAT_MIN 1
Минимальное число повторов кадра.
#define UEM_FRAME_REPEAT_MAX 1023
Максимальное число повторов кадра.
#define UEM_FRAME_REPEAT_DEFAULT (UEM_FRAME_REPEAT_MIN)
Число повторов кадра по умолчанию (1).
```

Флаги кадра.

*Данные константы могут использоваться в аргументе **frame_flags** функции **uem_bcp_append_frame()**.*

```
#define UEM_FRAME_STOP 0x0001
Остановка КИШ.
#define UEM_FRAME_ALLRPT 0x0002
Защелкивание программы КИШ.
#define UEM_FRAME_NONE 0
Нет указаний.
#define UEM_FRAME_CONT (UEM_FRAME_NONE)
Продолжение программы КИШ.
#define UEM_FRAME_DEFAULT (UEM_FRAME_STOP)
Флаги кадра по умолчанию: остановка КИШ.
```

Виды размерностей программы КИШ.

```
#define UEM_BCP_NFRAMES (-1)
Число кадров.
#define UEM_BCP_CUR_SIZE (-2)
Текущий размер.
#define UEM_BCP_MAX_SIZE (-3)
Максимальный размер.
```

Признаки обработки командного слова в ОУ.

*Из объединения (по |) этих констант составляется аргумент **rtdes** в функциях **uem_response_create()**, **uem_response_read()**.*

```
#define UEM_RTDES_SW_DIS (1<<15)
Не отвечать.
#define UEM_RTDES_DBCA (1<<6)
Принять управление интерфейсом.
```

```
#define UEM_RTDES_DBCA_BCSTART (1<<7)
Запустить КШ.
#define UEM_RTDES_COM_ILLEGAL (1<<17)
Недопустимая команда.
#define UEM_RTDES_LCMD_DW (1<<13)
Передать последнюю команду.
#define UEM_RTDES_SWB_SAV (1<<12)
Автоматическое формирование флагов ОС.
#define UEM_RTDES_WRONG_CH (1<<14)
Отвечать по другой шине.
#define UEM_RTDES_WA (1<<8)
Циркулярный возврат данных.
#define UEM_RTDES_WA_BRCST (1<<31)
Циркулярный возврат в групповых командах.
#define UEM_RTDES_ILLEG_MASK (1<<16)
Задать маску допустимых команд в зависимости от количества слов данных.
#define UEM_RTDES_DEFAULT (UEM_RTDES_SWB_SAV)
Значение параметра rtdes по умолчанию.
```

Вспомогательные константы для полей командных слов.

В соответствии с [1].

```
#define UEM_RTADDR_MIN 0
Минимальное значение адреса ОУ.
#define UEM_RTADDR_MAX 30
Максимальное значение адреса ОУ.
#define UEM_RTADDR_BRCST 31
Адрес для групповых сообщений.
#define UEM_RTADDR_MAX_EXT 31
Максимальное значение адреса ОУ в сетях с запретом групповых сообщений.
#define UEM_RT_RX 0
Признак передача/прием: прием.
#define UEM_RT_TX 1
Признак передача/прием: передача.
#define UEM_SADDR_MIN 1
Минимальное значение подадреса.
#define UEM_SADDR_MAX 30
Максимальное значение подадреса.
#define UEM_SADDR_CONV_LOOPBACK 30
Традиционный подадрес для тестовой петли.
#define UEM_SADDR_MIN_EXT 0
Минимальное значение подадреса в сетях с запретом команд управления.
#define UEM_SADDR_MAX_EXT 31
Максимальное значение подадреса в сетях с запретом команд управления.
#define UEM_MODE_0 0
Код режима 0.
#define UEM_MODE_31 31
Код режима 31.
```

Коды команд управления.

В соответствии с [1].

```
#define UEM_MCODE_ADVC 0
Принять управление интерфейсом.
#define UEM_MCODE_SYNCHRO 1
Синхронизация.
#define UEM_MCODE_TXSTATUS 2
Передать ОС.
#define UEM_MCODE_BSELFTEST 3
Начать самоконтроль.
#define UEM_MCODE_BTMT 4
Блокировать передатчик.
#define UEM_MCODE_UBTMT 5
Разблокировать передатчик.
#define UEM_MCODE_BRTF 6
Блокировать признак неисправности ОУ.
#define UEM_MCODE_UBRTF 7
Разблокировать признак неисправности ОУ.
#define UEM_MCODE_RESETRT 8
Установить ОУ в исходное состояние.
#define UEM_MCODE_TXVECT 16
Передать векторное слово.
#define UEM_MCODE_SYNCHRO_D 17
Синхронизация (с СД).
#define UEM_MCODE_TXLCMD 18
Передать последнюю команду.
#define UEM_MCODE_TXBIT 19
Передать слово ВСК ОУ.
#define UEM_MCODE_BTMT_I 20
Блокировать i-й передатчик.
#define UEM_MCODE_UBTMT_I 21
Разблокировать i-й передатчик.
```

Признаки ответного слова.

В соответствии с [1].

```
#define UEM_RTFAIL 0x0001
НОУ - Неисправность оконечного устройства.
#define UEM_DBCA 0x0002
ПУИ - Принято управление интерфейсом.
#define UEM_ABFAIL 0x0004
НА - Неисправность абонента.
#define UEM_ABBUSY 0x0008
АЗ - Абонент занят.
#define UEM_BCCA 0x0010
ПГК - Принята групповая команда.
#define UEM_RSIV14 0x0020
Резервный бит 14.
```

```
#define UEM_RSV13 0x0040
Резервный бит 13.
#define UEM_RSV12 0x0080
Резервный бит 12.
#define UEM_SERVRQ 0x0100
ЗО - Запрос обслуживания.
#define UEM_ZERO 0x0200
Передача ОС.
#define UEM_MSGERR 0x0400
ОШС - Ошибка в сообщении.
```

Определения типов

- typedef ViSession **UEM_DEVHANDLE**
Дескриптор устройства УЭМ или виртуального устройства в составе УЭМ.
- typedef ViSession **UEM_OBJHANDLE**
Дескриптор объекта в ОЗУ УЭМ.
- typedef ViBoolean **UEM_BOOL**
Логическое значение.
- typedef ViUInt16 **UEM_PARAMID**
Идентификатор параметра.
- typedef ViUInt32 **UEM_DWORD**
32-битное целое без знака.
- typedef ViUInt16 **UEM_WORD**
16-битное целое без знака.
- typedef ViUInt64 **UEM_TIME_TAG_LIN**
Метка времени в линейном формате
- typedef ViUInt32 **UEM_ERROR_FLAGS**
Признаки ошибок распознавания сообщения в мониторе шины.
- typedef void(* **uem_bm_handler**)(UEM_DEVHANDLE bm, void *userdata)
Обработчик события МШ.

Перечисления

- enum **UEM_ERROR_TYPE** { UEM_ERRT_NONE = 0, UEM_ERRT_INV_PARITY = 1, UEM_ERRT_WRONG_BITCOUNT = 2, UEM_ERRT_BAD_SYNCHRO = 3, UEM_ERRT_BAD_BIPHASE_ZERO = 4, UEM_ERRT_BAD_BIPHASE_POS = 5, UEM_ERRT_BAD_BIPHASE_NEG = 6, UEM_ERRT_SHIFT_EDGE = 7 }
Тип вносимой ошибки кодирования.
 - enum **UEM_SYNCHRO_ERROR_POS** { UEM_BAD_SYNCHRO_NONE = 0, UEM_BAD_SYNCHRO_IIIIIE = 1, UEM_BAD_SYNCHRO_EIIIEE = 2, UEM_BAD_SYNCHRO_EEIEEE = 3, UEM_BAD_SYNCHRO_NONE2 = 4, UEM_BAD_SYNCHRO_EEEIEE = 5, UEM_BAD_SYNCHRO_EEEEIE = 6, UEM_BAD_SYNCHRO_EEEEEI = 7 }
Позиция ошибки кодирования синхроимпульса.
 - enum **UEM_HANDLE_TYPE** { UEM_INVH, UEM_UEM, UEM_BC, UEM_RT, UEM_BM, UEM_CSEG, UEM_BCP, UEM_RESP }
 - enum **UEM_TIME_PARAM** { UEM_MIN_T1, UEM_MIN_T2, UEM_RTMO }
- Идентификатор (селектор) параметра интервала времени*

- enum **UEM_FORMAT** { **UEM_UNF**, **UEM_F1**, **UEM_F2**, **UEM_F3**, **UEM_F4**, **UEM_F5**, **UEM_F6**, **UEM_F7**, **UEM_F8**, **UEM_F9**, **UEM_F10** }
Форматы сообщений (номера по ГОСТ [1]).
- enum **UEM_CHANNEL** { **UEM_CH_A**, **UEM_CH_B** }
Селектор шины (А/В).
- enum **UEM_SYNC** { **UEM_SYNC_D**, **UEM_SYNC_C** }
Селектор синхроимпульса.
- enum **UEM_CSEG_TYPE** { **UEM_CSEG_NORMAL**, **UEM_CSEG_OVERLAY**, **UEM_CSEG_GAP** }
Тип командного сегмента.

Функции

- ViStatus **uem_init** (ViRsrc idstr, ViBoolean idn, ViBoolean reset, ViSession *uem)
Инициализация объекта УЭМ.
- ViStatus **uem_connect** (ViSession uem, ViSession vi, ViUInt16 mezznum, ViBoolean idn, ViBoolean reset)
Привязка объекта УЭМ к сеансу носителя мезонина.
- ViStatus **uem_bc_init** (**UEM_DEVHANDLE** *bc, **UEM_DEVHANDLE** uem)
Открытие виртуального КШ в составе УЭМ.
- ViStatus **uem_rt_init** (**UEM_DEVHANDLE** *rt, **UEM_DEVHANDLE** uem, **UEM_WORD** rtaddr)
Открытие виртуального ОУ в составе УЭМ.
- ViStatus **uem_bm_init** (**UEM_DEVHANDLE** *bm, **UEM_DEVHANDLE** uem)
Открытие виртуального МШ в составе УЭМ.
- ViStatus **uem_close** (**UEM_DEVHANDLE** anydev)
Закрытие УЭМ или любого виртуального устройства в составе УЭМ.
- **UEM_HANDLE_TYPE** **uem_handle_type** (**UEM_DEVHANDLE** anyobject)
Тип дескриптора объекта библиотеки УЭМ.
- ViStatus **uem_parent_dev** (**UEM_DEVHANDLE** anyobject, **UEM_DEVHANDLE** *parentdev)
Родительское устройство.
- ViStatus **uem_root_dev** (**UEM_DEVHANDLE** anyobject, **UEM_DEVHANDLE** *uem)
Физическое устройство УЭМ.
- ViStatus **uem_layer_handle** (**UEM_DEVHANDLE** uem, ViUInt32 sel, ViSession *handle)
Связь с ПО нижележащих слоев.
- ViStatus **uem_param_get** (**UEM_DEVHANDLE** uem, **UEM_PARAMID** paramid, **UEM_DWORD** *value)
Считывание значения конфигурационного параметра.
- ViStatus **uem_param_set** (**UEM_DEVHANDLE** uem, **UEM_PARAMID** paramid, **UEM_DWORD** value)
Запись значения конфигурационного параметра.
- ViStatus **uem_timing_set** (**UEM_DEVHANDLE** anydev, **UEM_TIME_PARAM** param, **UEM_DWORD** value)
Установка параметра интервала времени.
- ViStatus **uem_timing_get** (**UEM_DEVHANDLE** anydev, **UEM_TIME_PARAM** param, **UEM_DWORD** *value)
Считывание параметра интервала времени.
- ViStatus **uem_time_tag_get** (**UEM_DEVHANDLE** uem, **UEM_TIME_TAG** *time_tag)
Считывание встроенного счетчика времени.
- ViStatus **uem_time_tag_set** (**UEM_DEVHANDLE** uem, **UEM_TIME_TAG** *time_tag)
Установка значения встроенного счетчика времени.
- ViStatus **uem_time_tag_reset** (**UEM_DEVHANDLE** uem)
Сброс встроенного счетчика времени.
- **UEM_TIME_TAG_LIN** **uem_time_tag_to_linear** (**UEM_TIME_TAG** *time_tag)
Перевод метки времени в линейный формат.

- void **uem_time_tag_to_struct** (UEM_TIME_TAG *time_tag, UEM_TIME_TAG_LIN linear)
Перевод метки времени из линейного в структурированный формат.
- ViStatus **uem_bc_cseg_format** (UEM_DEVHANDLE bc, UEM_CMD_SEG *cseg_data, UEM_CHANNEL ch, UEM_FORMAT format, UEM_WORD rt, UEM_WORD sa, UEM_WORD ndatawords, UEM_WORD *datawords)
Формирование образов командных сегментов для сообщений форматов 1,2,7 и неформатных сообщений.
- ViStatus **uem_bc_cseg_format_RTRT** (UEM_DEVHANDLE bc, UEM_CMD_SEG *cseg_data, UEM_CHANNEL ch, UEM_FORMAT format, UEM_WORD rtrt, UEM_WORD sarx, UEM_WORD rrtx, UEM_WORD satx, UEM_WORD ndatawords)
Формирование образов командных сегментов для сообщений форматов 3,8.
- ViStatus **uem_bc_cseg_format_MODE** (UEM_DEVHANDLE bc, UEM_CMD_SEG *cseg_data, UEM_CHANNEL ch, UEM_FORMAT format, UEM_WORD rt, UEM_WORD mode, UEM_WORD modecode, UEM_WORD dataword)
Формирование образов командных сегментов для сообщений форматов 4,5,6,9,10.
- ViStatus **uem_bc_cseg_create** (UEM_DEVHANDLE bc, UEM_OBJHANDLE *cseg, UEM_CMD_SEG *cseg_data)
Создание командного сегмента.
- ViStatus **uem_cseg_read** (UEM_OBJHANDLE cseg, UEM_CMD_SEG *cseg_data)
Чтение командного сегмента.
- ViStatus **uem_cseg_gap_set** (UEM_OBJHANDLE cseg, UEM_WORD gap, UEM_WORD gap_flags, UEM_WORD gap_timeout)
Программирование паузы перед сообщением.
- ViStatus **uem_cseg_gap_get** (UEM_OBJHANDLE cseg, UEM_WORD *gap, UEM_WORD *gap_flags, UEM_WORD *gap_timeout)
Считывание паузы перед сообщением.
- ViStatus **uem_cseg_gap_reset** (UEM_OBJHANDLE cseg)
Сброс паузы перед сообщением.
- ViStatus **uem_cseg_word_gap_set** (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_WORD gap, UEM_WORD gap_flags, UEM_WORD gap_timeout)
Программирование паузы между словами.
- ViStatus **uem_cseg_word_gap_get** (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_WORD *gap, UEM_WORD *gap_flags, UEM_WORD *gap_timeout)
Считывание паузы перед словом.
- ViStatus **uem_cseg_error_set** (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_ERROR_TYPE error_type, ViInt32 error_pos, ViInt32 error_param)
Внесение ошибок кодирования.
- ViStatus **uem_cseg_error_get** (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_ERROR_TYPE *error_type, ViInt32 *error_pos, ViInt32 *error_param)
Считывание внесенных ошибок кодирования.
- ViStatus **uem_cseg_sync_set** (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_SYNC sync)
Установка типа синхроимпульса.
- ViStatus **uem_cseg_sync_get** (UEM_OBJHANDLE cseg, UEM_WORD wordnumber, UEM_SYNC *sync)
Считывание типа синхроимпульса.
- ViStatus **uem_bc_gap_create** (UEM_DEVHANDLE bc, UEM_OBJHANDLE *cseg, UEM_WORD gap, UEM_WORD gap_flags, UEM_WORD gap_timeout, UEM_CHANNEL ch)
Создание паузы.
- ViStatus **uem_bc_cseg_overlay** (UEM_DEVHANDLE bc, UEM_OBJHANDLE *cseg_o, UEM_OBJHANDLE cseg_1, UEM_WORD gap, UEM_OBJHANDLE cseg_2)
Создание сообщения с наложением.

- ViStatus **uem_cseg_type** (UEM_OBJHANDLE cseg, UEM_CSEG_TYPE *type)
Запрос типа командного сегмента.
- ViStatus **uem_cseg_desrtoy** (UEM_OBJHANDLE cseg)
Уничтожение командного сегмента.
- ViStatus **uem_bcp_create** (UEM_OBJHANDLE *bcprog, UEM_DWORD max_size, UEM_DEVHANDLE bc)
Создание программы КИИ.
- ViStatus **uem_bcp_append_frame** (UEM_OBJHANDLE bcprog, UEM_WORD repeat_count, UEM_WORD frame_flags, int *frameindex)
Добавление кадра в конец программы КИИ.
- ViStatus **uem_bcp_append_cseg** (UEM_OBJHANDLE bcprog, UEM_OBJHANDLE cseg, int *csegindex)
Добавление командного сегмента в конец кадра.
- ViStatus **uem_bcp_discover_cseg** (UEM_OBJHANDLE bcprog, int frameindex, int csegindex, UEM_OBJHANDLE *cseg)
Выяснение командного сегмента.
- ViStatus **uem_bcp_replace_cseg** (UEM_OBJHANDLE bcprog, int frameindex, int csegindex, UEM_OBJHANDLE cseg)
Замена командного сегмента в кадре.
- ViStatus **uem_bcp_dimension** (UEM_OBJHANDLE bcprog, int frameindex, int *dim)
Запрос размерностей программы КИИ.
- ViStatus **uem_bcp_inspect_frame** (UEM_OBJHANDLE bcprog, int frameindex, UEM_WORD *repeat_count, UEM_WORD *frame_flags)
Запрос характеристик кадра.
- ViStatus **uem_bcp_install** (UEM_OBJHANDLE bcprog)
Установка программы КИИ в качестве исполняемой.
- ViStatus **uem_bcp_desrtoy** (UEM_OBJHANDLE bcprog)
Уничтожение объекта "программа КИИ" в ОЗУ КИИ.
- ViStatus **uem_bcp_set_standard_gaps** (UEM_OBJHANDLE bcprog)
Расчет и установка стандартных пауз между сообщениями (необязательно).
- ViStatus **uem_bc_send_receive** (UEM_DEVHANDLE bc, UEM_CMD_SEG *cseg_data, UEM_BM_MESSAGE **msg_and_resp)
Передача отдельного сообщения и получение ответа на него.
- ViStatus **uem_response_create** (UEM_OBJHANDLE *resp, UEM_DEVHANDLE rt, UEM_DWORD rtdes, UEM_DWORD illeg_mask, UEM_WORD status, UEM_WORD ndatawords, UEM_WORD *data)
Создание ответного сегмента.
- ViStatus **uem_response_read** (UEM_OBJHANDLE resp, UEM_DWORD *rtdes, UEM_DWORD *illeg_mask, UEM_WORD *status, UEM_WORD *ndatawords, UEM_WORD *data)
Считывание ответного сегмента.
- ViStatus **uem_response_gap_set** (UEM_OBJHANDLE resp, UEM_WORD gap)
Установка паузы перед передачей ответного сегмента.
- ViStatus **uem_response_gap_get** (UEM_OBJHANDLE resp, UEM_WORD *gap)
Считывание паузы перед передачей ответного сегмента.
- ViStatus **uem_response_word_gap_set** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_WORD gap)
Установка паузы перед передачей слова ответного сегмента.
- ViStatus **uem_response_word_gap_get** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_WORD *gap)
Считывание паузы перед передачей слова ответного сегмента.

- ViStatus **uem_response_error_set** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_ERROR_TYPE error_type, ViInt32 error_pos, ViInt32 error_param)
Внесение ошибок кодирования в ответный сегмент.
- ViStatus **uem_response_error_get** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_ERROR_TYPE *error_type, ViInt32 *error_pos, ViInt32 *error_param)
Считывание ошибок кодирования из ответного сегмента.
- ViStatus **uem_response_sync_set** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_SYNC sync)
Установка типа синхроимпульса.
- ViStatus **uem_response_sync_get** (UEM_OBJHANDLE resp, UEM_WORD wordnumber, UEM_SYNC *sync)
Считывание типа синхроимпульса.
- ViStatus **uem_rt_install_response** (UEM_DEVHANDLE rt, UEM_BOOL transmit, UEM_WORD sa, UEM_OBJHANDLE resp)
Установка ответного сегмента как ответа на команду передачи данных.
- ViStatus **uem_rt_install_response_MODE** (UEM_DEVHANDLE rt, UEM_BOOL transmit, UEM_WORD mode, UEM_WORD modecode, UEM_OBJHANDLE resp)
Установка ответного сегмента как ответа на команду управления.
- ViStatus **uem_rt_discover_response** (UEM_DEVHANDLE rt, UEM_BOOL transmit, UEM_WORD sa, UEM_OBJHANDLE *resp)
Выяснение ответа на команду передачи данных.
- ViStatus **uem_rt_discover_response_MODE** (UEM_DEVHANDLE rt, UEM_BOOL transmit, UEM_WORD mode, UEM_WORD modecode, UEM_OBJHANDLE *resp)
Выяснение ответа на команду управления.
- ViStatus **uem_response_destroy** (UEM_OBJHANDLE resp)
Уничтожение объекта ответного сегмента в ОЗУ ОУ.
- ViStatus **uem_bm_receive** (UEM_DEVHANDLE bm, UEM_BM_MESSAGE **message_data)
Считывание очередного сообщения, принятого монитором шины.
- ViStatus **uem_bm_queue_count** (UEM_DEVHANDLE bm, UEM_DWORD *count)
Запрос размера очереди сообщений, принятых монитором.
- ViStatus **uem_bm_install_handler** (UEM_DEVHANDLE bm, uem_bm_handler handler, void *userdata)
Установка обработчика события МШ.
- ViStatus **uem_start** (UEM_DEVHANDLE anydev)
Запуск любого виртуального устройства в составе УЭМ.
- ViStatus **uem_stop** (UEM_DEVHANDLE anydev)
Остановка любого виртуального устройства в составе УЭМ.
- ViStatus **uem_is_running** (UEM_DEVHANDLE anydev, UEM_BOOL *running)
Проверка активности виртуального устройства.
- ViStatus **uem_error_message** (UEM_DEVHANDLE uem, ViStatus status, ViChar msg[])
Запрос сообщения об ошибке.
- ViStatus **uem_error_query** (ViSession uem, ViInt32 *status, ViChar msg[])
Запрос последней ошибки.
- ViStatus **uem_reset** (UEM_DEVHANDLE anydev)
Сброс УЭМ или любого виртуального устройства в составе УЭМ.
- ViStatus **uem_revision_query** (UEM_DEVHANDLE uem, ViChar dv[], ViChar iv[])
Запрос версии.
- ViStatus **uem_self_test** (UEM_DEVHANDLE uem, ViInt16 *res, ViChar msg[])
Самоконтроль.

- **UEM_WORD uem_command_word** (UEM_WORD rt, UEM_BOOL tx, UEM_WORD sa, UEM_WORD ndatawords)
Формирование командного слова.
- **void uem_command_word_parse** (UEM_WORD cw, UEM_WORD *rt, UEM_BOOL *tx, UEM_WORD *sa, UEM_WORD *ndatawords)
Разбор командного слова.
- **UEM_WORD uem_status_word** (UEM_WORD rt, UEM_WORD status_bits)
Формирование ответного слова.
- **void uem_status_word_parse** (UEM_WORD sw, UEM_WORD *rt, UEM_WORD *status_bits)
Разбор ответного слова.

Запуск виртуального КШ с дополнительными параметрами

- **#define UEM_BC_START_NOW 0**
Нормальный (немедленный) старт.
- **#define UEM_BC_START_WAITING 1**
Переход в режим ожидания, старт по внешнему сигналу или команде.
- **#define UEM_BC_START_DEFAULT (UEM_BC_START_NOW)**
Стандартный способ старта.
- **enum UEM_BC_STATE_EX { UEM_BC_STOPPED, UEM_BC_WAITING, UEM_BC_RUNNING }**
Расширенное состояние КШ.
- **ViStatus uem_bc_start** (UEM_DEVHANDLE bc, UEM_DWORD flags)
Запуск виртуального КШ в составе УЭМ с дополнительными параметрами.
- **ViStatus uem_bc_state_ex** (UEM_DEVHANDLE bc, UEM_BC_STATE_EX *state)
Запрос расширенного состояния КШ.

Остановка виртуального КШ с дополнительными параметрами

- **#define UEM_BC_STOP_NOW 0**
Нормальная (немедленная) остановка.
- **#define UEM_BC_STOP_ON_FRAME 1**
Остановка по завершению текущего кадра.
- **#define UEM_BC_STOP_DEFAULT (UEM_BC_STOP_NOW)**
Стандартный способ остановки.
- **ViStatus uem_bc_stop** (UEM_DEVHANDLE bc, UEM_DWORD flags)
Остановка виртуального КШ в составе УЭМ с дополнительными параметрами.

Подробное описание

Универсальные электронные модули УЭМ-МК, МВ98.03. Расширенная библиотека функций. Файл заголовков функций.

Алфавитный указатель

- b
 - UEM_TIME_TAG, 129
- ch
 - UEM_CMD_SEG, 121
- command1
 - UEM_CMD_SEG, 121
- command2
 - UEM_CMD_SEG, 121
- cs
 - UEM_BM_MESSAGE, 119
- cs_d
 - UEM_BM_MESSAGE, 119
- data
 - UEM_CMD_SEG, 122
 - UEM_DATA, 123
 - UEM_RAW_BM_MESSAGE, 124
 - UEM_RESP_SEG, 125
- days
 - UEM_TIME_TAG, 128
- dwoffset
 - UEM_SEGMENT_DESCR, 127
- end_time
 - UEM_SEGMENT_DESCR, 126
- endoffset
 - UEM_SEGMENT_DESCR, 127
- errors
 - UEM_BM_MESSAGE, 120
 - UEM_SEGMENT_DESCR, 126
- format
 - UEM_CMD_SEG, 121
- gap
 - UEM_BM_MESSAGE, 120
- i
 - UEM_TIME_TAG, 129
- lostp
 - UEM_BM_MESSAGE, 120
- msg_d
 - UEM_BM_MESSAGE, 119
- ndata
 - UEM_DATA, 123
- offset
 - UEM_SEGMENT_DESCR, 126
- overlay
 - UEM_BM_MESSAGE, 120
- p_pr
 - UEM_BM_MESSAGE, 120
- quartas
 - UEM_TIME_TAG, 128
- raw
 - UEM_BM_MESSAGE, 120
- reserved
 - UEM_TIME_TAG, 128
- rs1
 - UEM_BM_MESSAGE, 119
- rs1_d
 - UEM_BM_MESSAGE, 119
- rs1_pr
 - UEM_BM_MESSAGE, 119
- rs2
 - UEM_BM_MESSAGE, 119
- rs2_d
 - UEM_BM_MESSAGE, 120
- rs2_pr
 - UEM_BM_MESSAGE, 119
- secs
 - UEM_TIME_TAG, 128
- size
 - UEM_RAW_BM_MESSAGE, 124
 - UEM_SEGMENT_DESCR, 127
- start_time
 - UEM_SEGMENT_DESCR, 126
- status
 - UEM_RESP_SEG, 125
- uem.h, 130
- UEM_ABBUSY
 - Формирование и разбор командных и ответных слов, 116
- UEM_ABFAIL
 - Формирование и разбор командных и ответных слов, 116
- UEM_BAD_SYNCHRO_EEEEEI
 - Типы вносимых ошибок кодирования, 21
- UEM_BAD_SYNCHRO_EEEEIE
 - Типы вносимых ошибок кодирования, 21
- UEM_BAD_SYNCHRO_EEEIEE
 - Типы вносимых ошибок кодирования, 21
- UEM_BAD_SYNCHRO_EEIEEE
 - Типы вносимых ошибок кодирования, 21
- UEM_BAD_SYNCHRO_EIEEEE
 - Типы вносимых ошибок кодирования, 21
- UEM_BAD_SYNCHRO_IEEEEE
 - Типы вносимых ошибок кодирования, 21
- UEM_BAD_SYNCHRO_NONE
 - Типы вносимых ошибок кодирования, 21
- UEM_BAD_SYNCHRO_NONE2
 - Типы вносимых ошибок кодирования, 21
- UEM_BC
 - Действия с дескрипторами, 29
- uem_bc_cseg_create
 - Создание и настройка командных сегментов, 75
- uem_bc_cseg_format

- Заполнение образа командного сегмента в ОЗУ ПЭВМ, 69
- uem_bc_cseg_format_MODE
 - Заполнение образа командного сегмента в ОЗУ ПЭВМ, 70
- uem_bc_cseg_format_RTRT
 - Заполнение образа командного сегмента в ОЗУ ПЭВМ, 70
- uem_bc_cseg_overlay
 - Создание и настройка командных сегментов, 79
- uem_bc_gap_create
 - Создание и настройка командных сегментов, 79
- uem_bc_init
 - Установление и разрыв связи с устройством, 26
- UEM_BC_RUNNING
 - Запуск и остановка, 105
- uem_bc_send_receive
 - Передача сообщений, 89
- uem_bc_start
 - Запуск и остановка, 106
- UEM_BC_START_DEFAULT
 - Запуск и остановка, 105
- UEM_BC_START_NOW
 - Запуск и остановка, 105
- UEM_BC_START_WAITING
 - Запуск и остановка, 105
- uem_bc_state_ex
 - Запуск и остановка, 107
- UEM_BC_STATE_EX
 - Запуск и остановка, 105
- uem_bc_stop
 - Запуск и остановка, 107
- UEM_BC_STOP_DEFAULT
 - Запуск и остановка, 105
- UEM_BC_STOP_NOW
 - Запуск и остановка, 105
- UEM_BC_STOP_ON_FRAME
 - Запуск и остановка, 105
- UEM_BC_STOPPED
 - Запуск и остановка, 105
- UEM_BC_WAITING
 - Запуск и остановка, 105
- UEM_BCCA
 - Формирование и разбор командных и ответных слов, 116
- UEM_BCP
 - Действия с дескрипторами, 29
- uem_bcp_append_cseg
 - Создание и настройка кадров и программы КШ, 84
- uem_bcp_append_frame
 - Создание и настройка кадров и программы КШ, 84
- uem_bcp_create
 - Создание и настройка кадров и программы КШ, 84
- UEM_BCP_CUR_SIZE
 - Создание и настройка кадров и программы КШ, 83
- uem_bcp_desrtoy
 - Создание и настройка кадров и программы КШ, 86
- uem_bcp_dimension
 - Создание и настройка кадров и программы КШ, 85
- uem_bcp_discover_cseg
 - Создание и настройка кадров и программы КШ, 85
- uem_bcp_inspect_frame
 - Создание и настройка кадров и программы КШ, 86
- uem_bcp_install
 - Создание и настройка кадров и программы КШ, 86
- UEM_BCP_MAX_SIZE
 - Создание и настройка кадров и программы КШ, 83
- UEM_BCP_NFRAMES
 - Создание и настройка кадров и программы КШ, 83
- uem_bcp_replace_cseg
 - Создание и настройка кадров и программы КШ, 85
- uem_bcp_set_standard_gaps
 - Создание и настройка кадров и программы КШ, 87
- UEM_BCRT
 - Определения типов данных для КШ, ОУ, МШ, 63
- UEM_BCRTЪ
 - Определения типов данных для КШ, ОУ, МШ, 63
- UEM_BIPHASE_POS_MAX
 - Типы вносимых ошибок кодирования, 18
- UEM_BIPHASE_POS_MIN
 - Типы вносимых ошибок кодирования, 18
- UEM_BITCOUNT_CHANGE_MAX
 - Типы вносимых ошибок кодирования, 18
- UEM_BITCOUNT_CHANGE_MIN
 - Типы вносимых ошибок кодирования, 18
- UEM_BM
 - Действия с дескрипторами, 29
- uem_bm_handler
 - Функции МШ, 102
- uem_bm_init
 - Установление и разрыв связи с устройством, 27
- uem_bm_install_handler
 - Функции МШ, 103
- UEM_BM_MESSAGE, 118
 - cs, 119
 - cs_d, 119
 - errors, 120
 - gap, 120
 - lostp, 120
 - msg_d, 119
 - overlay, 120

- p_pr, 120
- raw, 120
- rs1, 119
- rs1_d, 119
- rs1_pr, 119
- rs2, 119
- rs2_d, 120
- rs2_pr, 119
- uem_bm_queue_count
 - Функции МШ, 103
- uem_bm_receive
 - Функции МШ, 102
- UEM_BM_WORD_DEFAULT
 - Значения параметров, 54
- UEM_BM_WORD_MASK
 - Описание параметров, 41
- UEM_BM_WORD_MAX
 - Значения параметров, 54
- UEM_BM_WORD_MIN
 - Значения параметров, 54
- UEM_BM_WORD_PATTERN
 - Описание параметров, 41
- UEM_BOOL
 - Определения примитивных типов, 23
- UEM_BRCST_DEFAULT
 - Значения параметров, 49
- UEM_BRCST_DIS
 - Описание параметров, 36
- UEM_BRCST_DISABLED
 - Значения параметров, 49
- UEM_BRCST_ENABLED
 - Значения параметров, 49
- UEM_BRTF_DEFAULT
 - Значения параметров, 52
- UEM_BRTF_DIS
 - Описание параметров, 39
- UEM_BRTF_DISABLED
 - Значения параметров, 52
- UEM_BRTF_ENABLED
 - Значения параметров, 52
- UEM_BTMT_DEFAULT
 - Значения параметров, 52
- UEM_BTMT_DIS
 - Описание параметров, 39
- UEM_BTMT_DISABLED
 - Значения параметров, 52
- UEM_BTMT_ENABLED
 - Значения параметров, 52
- UEM_CH_A
 - Определения типов данных для КШ, ОУ, МШ, 66
- UEM_CH_B
 - Определения типов данных для КШ, ОУ, МШ, 66
- UEM_CHANNEL
 - Определения типов данных для КШ, ОУ, МШ, 66
- uem_close
 - Установление и разрыв связи с устройством, 27
- UEM_CMD_SEG, 121
- ch, 121
- command1, 121
- command2, 121
- data, 122
- format, 121
- uem_command_word
 - Формирование и разбор командных и ответных слов, 116
- uem_command_word_parse
 - Формирование и разбор командных и ответных слов, 117
- uem_connect
 - Установление и разрыв связи с устройством, 26
- UEM_CSEG
 - Действия с дескрипторами, 29
- uem_cseg_desrtoy
 - Создание и настройка командных сегментов, 80
- uem_cseg_error_get
 - Создание и настройка командных сегментов, 78
- uem_cseg_error_set
 - Создание и настройка командных сегментов, 77
- UEM_CSEG_GAP
 - Создание и настройка командных сегментов, 75
- UEM_CSEG_GAP_ALT_AB
 - Создание и настройка командных сегментов, 74
- UEM_CSEG_GAP_ALT_BUS
 - Создание и настройка командных сегментов, 74
- UEM_CSEG_GAP_DEFAULT_FLAGS
 - Создание и настройка командных сегментов, 74
- UEM_CSEG_GAP_DEFAULT_TIMEOUT
 - Создание и настройка командных сегментов, 75
- UEM_CSEG_GAP_DEFAULT_VALUE
 - Создание и настройка командных сегментов, 74
- UEM_CSEG_GAP_ESYNC
 - Создание и настройка командных сегментов, 73
- UEM_CSEG_GAP_FROM_END
 - Создание и настройка командных сегментов, 73
- UEM_CSEG_GAP_FROM_START
 - Создание и настройка командных сегментов, 73
- uem_cseg_gap_get
 - Создание и настройка командных сегментов, 76
- UEM_CSEG_GAP_MAX
 - Создание и настройка командных сегментов, 74
- UEM_CSEG_GAP_MIN
 - Создание и настройка командных сегментов, 74
- uem_cseg_gap_reset
 - Создание и настройка командных сегментов, 76
- uem_cseg_gap_set
 - Создание и настройка командных сегментов, 76
- UEM_CSEG_GAP_THIS_AB
 - Создание и настройка командных сегментов, 74
- UEM_CSEG_GAP_THIS_BUS
 - Создание и настройка командных сегментов, 74
- UEM_CSEG_GAP_TIMEOUT_MAX
 - Создание и настройка командных сегментов, 75
- UEM_CSEG_GAP_TIMEOUT_MIN
 - Создание и настройка командных сегментов, 74

UEM_CSEG_NORMAL	Определения типов данных для КИШ, ОУ, МИШ, 64
Создание и настройка командных сегментов, 75	UEM_ERRF_FORMAT_MC
UEM_CSEG_OVERLAY	Определения типов данных для КИШ, ОУ, МИШ, 65
Создание и настройка командных сегментов, 75	UEM_ERRF_GAPN
uem_cseg_read	Определения типов данных для КИШ, ОУ, МИШ, 65
Создание и настройка командных сегментов, 75	UEM_ERRF_INC_MODE_CODE
uem_cseg_sync_get	Определения типов данных для КИШ, ОУ, МИШ, 65
Создание и настройка командных сегментов, 78	UEM_ERRF_INCORRECT_RTN
uem_cseg_sync_set	Определения типов данных для КИШ, ОУ, МИШ, 65
Создание и настройка командных сегментов, 78	UEM_ERRF_LESS_BITS
uem_cseg_type	Определения типов данных для КИШ, ОУ, МИШ, 65
Создание и настройка командных сегментов, 80	UEM_ERRF_MINGAP
UEM_CSEG_TYPE	Определения типов данных для КИШ, ОУ, МИШ, 64
Создание и настройка командных сегментов, 75	UEM_ERRF_MISSING_CWSW
uem_cseg_word_gap_get	Определения типов данных для КИШ, ОУ, МИШ, 65
Создание и настройка командных сегментов, 77	UEM_ERRF_MISSING_DW
uem_cseg_word_gap_set	Определения типов данных для КИШ, ОУ, МИШ, 65
Создание и настройка командных сегментов, 77	UEM_ERRF_MORE_BITS
UEM_DATA, 123	Определения типов данных для КИШ, ОУ, МИШ, 65
data, 123	UEM_ERRF_NO_RESPONSE
ndata, 123	Определения типов данных для КИШ, ОУ, МИШ, 64
UEM_DB_ACT	UEM_ERRF_PARITY
Описание параметров, 37	Определения типов данных для КИШ, ОУ, МИШ, 65
UEM_DB_ACTIVE	UEM_ERRF_RTRT_FORMAT
Значения параметров, 50	Определения типов данных для КИШ, ОУ, МИШ, 65
UEM_DB_INACTIVE	UEM_ERRF_SYNC_TYPE
Значения параметров, 50	Определения типов данных для КИШ, ОУ, МИШ, 64
UEM_DBCA	UEM_ERROR_ADDRESS_OUT_OF_RANGE
Формирование и разбор командных и ответных слов, 115	Коды завершения, 12
UEM_DEVHANDLE	UEM_ERROR_BAD_OVERLAY_SOURCE
Определения примитивных типов, 23	Коды завершения, 12
UEM_DWORD	UEM_ERROR_BAD_PARAM_VALUE
Определения примитивных типов, 23	Коды завершения, 10
UEM_ERR_INJ_DEFAULT	UEM_ERROR_BAD_PARAM_VALUE_1
Значения параметров, 50	Коды завершения, 10
UEM_ERR_INJ_DIS	UEM_ERROR_BAD_PARAM_VALUE_10
Описание параметров, 36	Коды завершения, 11
UEM_ERR_INJ_DISABLED	UEM_ERROR_BAD_PARAM_VALUE_2
Значения параметров, 50	Коды завершения, 11
UEM_ERR_INJ_ENABLED	UEM_ERROR_BAD_PARAM_VALUE_3
Значения параметров, 50	Коды завершения, 11
UEM_ERRF_DT	UEM_ERROR_BAD_PARAM_VALUE_4
Определения типов данных для КИШ, ОУ, МИШ, 66	Коды завершения, 11
UEM_ERRF_ENC	UEM_ERROR_BAD_PARAM_VALUE_5
Определения типов данных для КИШ, ОУ, МИШ, 66	Коды завершения, 11
UEM_ERRF_ENCODING	UEM_ERROR_BAD_PARAM_VALUE_6
Определения типов данных для КИШ, ОУ, МИШ, 64	Коды завершения, 11
UEM_ERRF_ENCODING2	UEM_ERROR_BAD_PARAM_VALUE_7
Определения типов данных для КИШ, ОУ, МИШ, 66	Коды завершения, 11
UEM_ERRF_ERROR	UEM_ERROR_BAD_PARAM_VALUE_8
Определения типов данных для КИШ, ОУ, МИШ, 64	Коды завершения, 11
UEM_ERRF_EXTRA_CWSW	UEM_ERROR_BAD_PARAM_VALUE_9
Определения типов данных для КИШ, ОУ, МИШ, 65	Коды завершения, 11
UEM_ERRF_EXTRA_DW	UEM_ERROR_BAD_TIMEOUT
Определения типов данных для КИШ, ОУ, МИШ, 65	Коды завершения, 12
UEM_ERRF_FORMAT	UEM_ERROR_BCP_NINST
	Коды завершения, 12

UEM_ERROR_BM_INTERNAL_BUFFER_OVERFLOW	Коды завершения, 13
UEM_ERROR_FLAGS	Определения типов данных для КШ, ОУ, МШ, 66
UEM_ERROR_FORMAT_DISABLED	Коды завершения, 12
UEM_ERROR_FORMAT_X_MCODE	Коды завершения, 12
UEM_ERROR_IN_USE	Коды завершения, 13
UEM_ERROR_INC_RESP	Коды завершения, 13
UEM_ERROR_INPOOL	Коды завершения, 12
UEM_ERROR_INV_HANDLE	Коды завершения, 11
UEM_ERROR_INV_HANDLE_TYPE	Коды завершения, 11
UEM_ERROR_MAX_SIZE_EXCEED	Коды завершения, 13
uem_error_message	Служебные функции, 108
UEM_ERROR_NO_FRAME_APPEND	Коды завершения, 13
UEM_ERROR_NO_FREE_RAM	Коды завершения, 11
UEM_ERROR_NO_HOST_MEM	Коды завершения, 12
UEM_ERROR_NOT_APPLICABLE	Коды завершения, 12
UEM_ERROR_NOT_CONNECTED	Коды завершения, 12
UEM_ERROR_NUMBER_OUT_OF_RANGE	Коды завершения, 12
UEM_ERROR_OFFSET	Коды завершения, 10
UEM_ERROR_PARAM_DEFAULT	Типы вносимых ошибок кодирования, 19
UEM_ERROR_POS_DEFAULT	Типы вносимых ошибок кодирования, 19
uem_error_query	Служебные функции, 109
UEM_ERROR_THREAD_FAULT	Коды завершения, 13
UEM_ERROR_TOO_MANY_DATAWORDS	Коды завершения, 13
UEM_ERROR_TYPE	Типы вносимых ошибок кодирования, 19
UEM_ERROR_TYPE_DEFAULT	Типы вносимых ошибок кодирования, 19
UEM_ERROR_WRONG_LOCATION	Коды завершения, 12
UEM_ERRT_BAD_BIPHASE_NEG	Типы вносимых ошибок кодирования, 20
UEM_ERRT_BAD_BIPHASE_POS	Типы вносимых ошибок кодирования, 20
UEM_ERRT_BAD_BIPHASE_ZERO	Типы вносимых ошибок кодирования, 20
UEM_ERRT_BAD_SYNCHRO	Типы вносимых ошибок кодирования, 19
UEM_ERRT_INV_PARITY	Типы вносимых ошибок кодирования, 19
UEM_ERRT_NONE	Типы вносимых ошибок кодирования, 19
UEM_ERRT_SHIFT_EDGE	Типы вносимых ошибок кодирования, 20
UEM_ERRT_WRONG_BITCOUNT	Типы вносимых ошибок кодирования, 19
UEM_F1	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F10	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F2	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F3	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F4	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F5	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F6	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F7	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F8	Определения типов данных для КШ, ОУ, МШ, 66
UEM_F9	Определения типов данных для КШ, ОУ, МШ, 66
UEM_FORMAT	Определения типов данных для КШ, ОУ, МШ, 66
UEM_FRAME_ALLRPT	Создание и настройка кадров и программы КШ, 83
UEM_FRAME_CONT	Создание и настройка кадров и программы КШ, 83
UEM_FRAME_DEFAULT	Создание и настройка кадров и программы КШ, 83
UEM_FRAME_NONE	Создание и настройка кадров и программы КШ, 83
UEM_FRAME_REPEAT_DEFAULT	Создание и настройка кадров и программы КШ, 82
UEM_FRAME_REPEAT_MAX	Создание и настройка кадров и программы КШ, 82
UEM_FRAME_REPEAT_MIN	Создание и настройка кадров и программы КШ, 82
UEM_FRAME_REPEAT_UNLIM	

- Создание и настройка кадров и программы КШ, 82
- UEM_FRAME_STOP
 - Создание и настройка кадров и программы КШ, 83
- uem_handle_type
 - Действия с дескрипторами, 29
- UEM_HANDLE_TYPE
 - Действия с дескрипторами, 29
- uem_init
 - Установление и разрыв связи с устройством, 25
- UEM_INVH
 - Действия с дескрипторами, 29
- uem_is_running
 - Запуск и остановка, 106
- UEM_IST_DEFAULT
 - Значения параметров, 54
- UEM_IST_MAX
 - Значения параметров, 54
- UEM_IST_MIN
 - Значения параметров, 54
- UEM_IST1
 - Описание параметров, 40
- UEM_IST2
 - Описание параметров, 40
- uem_layer_handle
 - Действия с дескрипторами, 30
- UEM_LIB_REV
 - Служебные функции, 108
- UEM_MC
 - Определения типов данных для КШ, ОУ, МШ, 63
- UEM_MC_DEFAULT
 - Значения параметров, 51
- UEM_MC_DIS
 - Описание параметров, 38
- UEM_MC_DISABLED
 - Значения параметров, 51
- UEM_MC_ENABLED
 - Значения параметров, 51
- UEM_MCб
 - Определения типов данных для КШ, ОУ, МШ, 64
- UEM_MCBCRT
 - Определения типов данных для КШ, ОУ, МШ, 63
- UEM_MCBCRTб
 - Определения типов данных для КШ, ОУ, МШ, 64
- UEM_MCODE_ADVC
 - Формирование и разбор командных и ответных слов, 114
- UEM_MCODE_BRTF
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_BSELFTEST
 - Формирование и разбор командных и ответных слов, 114
- UEM_MCODE_BTMT
 - Формирование и разбор командных и ответных слов, 114
- UEM_MCODE_BTMT_I
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_RESETRT
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_SYNCHRO
 - Формирование и разбор командных и ответных слов, 114
- UEM_MCODE_SYNCHRO_D
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_TXBIT
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_TXLCMD
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_TXSTATUS
 - Формирование и разбор командных и ответных слов, 114
- UEM_MCODE_TXVECT
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_UBRTF
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_UBTMT
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCODE_UBTMT_I
 - Формирование и разбор командных и ответных слов, 115
- UEM_MCRTBC
 - Определения типов данных для КШ, ОУ, МШ, 63
- UEM_MIN_T1
 - Параметры интервалов времени, 56
- UEM_MIN_T1_DEFAULT
 - Параметры интервалов времени, 56
- UEM_MIN_T2
 - Параметры интервалов времени, 56
- UEM_MIN_T2_DEFAULT
 - Параметры интервалов времени, 56
- UEM_MODE_0
 - Формирование и разбор командных и ответных слов, 114
- UEM_MODE_31
 - Формирование и разбор командных и ответных слов, 114
- UEM_MSGERR
 - Формирование и разбор командных и ответных слов, 116
- UEM_NDATA_BY_CW
 - Определения типов данных для КШ, ОУ, МШ, 64
- UEM_NDATA_MAX
 - Определения типов данных для КШ, ОУ, МШ, 64
- UEM_NDATA_MIN

- Определения типов данных для КШ, ОУ, МШ, 64
- UEM_OBJHANDLE
 - Определения примитивных типов, 23
- uem_param_get
 - Параметры конфигурации УЭМ, 31
- uem_param_set
 - Параметры конфигурации УЭМ, 32
- UEM_PARAMID
 - Определения примитивных типов, 23
- uem_parent_dev
 - Действия с дескрипторами, 29
- UEM_RAW_BM_MESSAGE, 124
 - data, 124
 - size, 124
- UEM_RCVA_DEFAULT
 - Значения параметров, 47
- UEM_RCVA_DIS
 - Описание параметров, 34
- UEM_RCVA_DISABLED
 - Значения параметров, 47
- UEM_RCVA_ENABLED
 - Значения параметров, 47
- UEM_RCVB_DEFAULT
 - Значения параметров, 48
- UEM_RCVB_DIS
 - Описание параметров, 34
- UEM_RCVB_DISABLED
 - Значения параметров, 47
- UEM_RCVB_ENABLED
 - Значения параметров, 47
- uem_reset
 - Служебные функции, 109
- UEM_RESP
 - Действия с дескрипторами, 29
- UEM_RESP_SEG, 125
 - data, 125
 - status, 125
- uem_response_create
 - Функции ОУ, 95
- uem_response_destroy
 - Функции ОУ, 101
- uem_response_error_get
 - Функции ОУ, 98
- uem_response_error_set
 - Функции ОУ, 97
- uem_response_gap_get
 - Функции ОУ, 96
- uem_response_gap_set
 - Функции ОУ, 96
- uem_response_read
 - Функции ОУ, 96
- uem_response_sync_get
 - Функции ОУ, 98
- uem_response_sync_set
 - Функции ОУ, 98
- uem_response_word_gap_get
 - Функции ОУ, 97
- uem_response_word_gap_set
 - Функции ОУ, 97
- uem_revision_query
 - Служебные функции, 109
- UEM_RFT_DEFAULT
 - Значения параметров, 51
- UEM_RFT_MAX
 - Значения параметров, 51
- UEM_RFT_MIN
 - Значения параметров, 51
- UEM_RFT_SINE
 - Значения параметров, 51
- uem_root_dev
 - Действия с дескрипторами, 30
- UEM_RSV12
 - Формирование и разбор командных и ответных слов, 116
- UEM_RSV13
 - Формирование и разбор командных и ответных слов, 116
- UEM_RSV14
 - Формирование и разбор командных и ответных слов, 116
- UEM_RT
 - Действия с дескрипторами, 29
- uem_rt_discover_response
 - Функции ОУ, 100
- uem_rt_discover_response_MODE
 - Функции ОУ, 100
- uem_rt_init
 - Установление и разрыв связи с устройством, 27
- uem_rt_install_response
 - Функции ОУ, 99
- uem_rt_install_response_MODE
 - Функции ОУ, 99
- UEM_RT_RX
 - Формирование и разбор командных и ответных слов, 113
- UEM_RT_TX
 - Формирование и разбор командных и ответных слов, 113
- UEM_RTADDR_BRCST
 - Формирование и разбор командных и ответных слов, 113
- UEM_RTADDR_MAX_EXT
 - Формирование и разбор командных и ответных слов, 113
- UEM_RTADDR_MIN
 - Формирование и разбор командных и ответных слов, 113
- UEM_RTADDR_MAX
 - Формирование и разбор командных и ответных слов, 113
- UEM_RTBC
 - Определения типов данных для КШ, ОУ, МШ, 63
- UEM_RTDES_COM_ILLEGAL
 - Функции ОУ, 92

UEM_RTDES_DBCA
 Функции ОУ, 92

UEM_RTDES_DBCA_BCSTART
 Функции ОУ, 92

UEM_RTDES_DEFAULT
 Функции ОУ, 95

UEM_RTDES_ILLEG_MASK
 Функции ОУ, 95

UEM_RTDES_LCMD_DW
 Функции ОУ, 92

UEM_RTDES_SW_DIS
 Функции ОУ, 91

UEM_RTDES_SWB_SAV
 Функции ОУ, 92

UEM_RTDES_WA
 Функции ОУ, 94

UEM_RTDES_WA_BRCST
 Функции ОУ, 95

UEM_RTDES_WRONG_CH
 Функции ОУ, 94

UEM_RTFAIL
 Формирование и разбор командных и ответных слов, 115

UEM_RTMO
 Параметры интервалов времени, 56

UEM_RTMO_DEFAULT
 Параметры интервалов времени, 56

UEM_RTRT
 Определения типов данных для КШ, ОУ, МШ, 63

UEM_RTRTb
 Определения типов данных для КШ, ОУ, МШ, 64

UEM_SADDR_CONV_LOOPBACK
 Формирование и разбор командных и ответных слов, 114

UEM_SADDR_MAX
 Формирование и разбор командных и ответных слов, 114

UEM_SADDR_MAX_EXT
 Формирование и разбор командных и ответных слов, 114

UEM_SADDR_MIN
 Формирование и разбор командных и ответных слов, 114

UEM_SADDR_MIN_EXT
 Формирование и разбор командных и ответных слов, 114

UEM_SEGMENT_DESCR, 126
 dwoffset, 127
 end_time, 126
 endoffset, 127
 errors, 126
 offset, 126
 size, 127
 start_time, 126

UEM_SEL_UNBASE_INT
 Действия с дескрипторами, 28

UEM_SEL_UNMBASE
 Действия с дескрипторами, 28

UEM_SEL_UNMUEM
 Действия с дескрипторами, 28

uem_self_test
 Служебные функции, 110

UEM_SERVRQ
 Формирование и разбор командных и ответных слов, 116

UEM_SET
 Значения параметров, 50

UEM_SHIFT_LENGTH_MAX
 Типы вносимых ошибок кодирования, 19

UEM_SHIFT_LENGTH_MIN
 Типы вносимых ошибок кодирования, 19

UEM_SHIFT_POS_MAX
 Типы вносимых ошибок кодирования, 19

UEM_SHIFT_POS_MIN
 Типы вносимых ошибок кодирования, 18

uem_start
 Запуск и остановка, 106

uem_status_word
 Формирование и разбор командных и ответных слов, 117

uem_status_word_parse
 Формирование и разбор командных и ответных слов, 117

uem_stop
 Запуск и остановка, 106

UEM_SYNC
 Определения типов данных для КШ, ОУ, МШ, 67

UEM_SYNC_C
 Определения типов данных для КШ, ОУ, МШ, 67

UEM_SYNC_D
 Определения типов данных для КШ, ОУ, МШ, 67

UEM_SYNC_IN_1_DEFAULT
 Значения параметров, 48

UEM_SYNC_IN_1_DISABLED
 Значения параметров, 48

UEM_SYNC_IN_1_ENA
 Описание параметров, 35

UEM_SYNC_IN_1_ENABLED
 Значения параметров, 48

UEM_SYNC_IN_1_INTGEN
 Описание параметров, 36

UEM_SYNC_IN_1_INTGEN_DEFAULT
 Значения параметров, 49

UEM_SYNC_IN_1_INTGEN_DISABLED
 Значения параметров, 49

UEM_SYNC_IN_1_INTGEN_ENABLED
 Значения параметров, 49

UEM_SYNC_IN_1_SET
 Описание параметров, 37

UEM_SYNC_IN_2_DEFAULT
 Значения параметров, 48

UEM_SYNC_IN_2_DISABLED
 Значения параметров, 48

UEM_SYNC_IN_2_ENA

- Описание параметров, 35
- UEM_SYNC_IN_2_ENABLED
 - Значения параметров, 48
- UEM_SYNC_IN_2_INTGEN
 - Описание параметров, 36
- UEM_SYNC_IN_2_INTGEN_DEFAULT
 - Значения параметров, 50
- UEM_SYNC_IN_2_INTGEN_DISABLED
 - Значения параметров, 50
- UEM_SYNC_IN_2_INTGEN_ENABLED
 - Значения параметров, 50
- UEM_SYNC_IN_2_SET
 - Описание параметров, 37
- UEM_SYNC_OUT_1_DEFAULT
 - Значения параметров, 49
- UEM_SYNC_OUT_1_DISABLED
 - Значения параметров, 48
- UEM_SYNC_OUT_1_ENA
 - Описание параметров, 35
- UEM_SYNC_OUT_1_ENABLED
 - Значения параметров, 48
- UEM_SYNC_OUT_1_SET
 - Описание параметров, 37
- UEM_SYNC_OUT_2_DEFAULT
 - Значения параметров, 49
- UEM_SYNC_OUT_2_DISABLED
 - Значения параметров, 49
- UEM_SYNC_OUT_2_ENA
 - Описание параметров, 35
- UEM_SYNC_OUT_2_ENABLED
 - Значения параметров, 49
- UEM_SYNC_OUT_2_SET
 - Описание параметров, 37
- UEM_SYNC1_ACH
 - Значения параметров, 54
- UEM_SYNC1_C_D_
 - Описание параметров, 39
- UEM_SYNC1_C_D_DEFAULT
 - Значения параметров, 53
- UEM_SYNC1_CH
 - Описание параметров, 40
- UEM_SYNC1_CH_A
 - Значения параметров, 53
- UEM_SYNC1_CH_B
 - Значения параметров, 54
- UEM_SYNC1_CH_DEFAULT
 - Значения параметров, 54
- UEM_SYNC1_ERR
 - Описание параметров, 39
- UEM_SYNC1_ERR_DEFAULT
 - Значения параметров, 53
- UEM_SYNC1_GAPB
 - Описание параметров, 40
- UEM_SYNC1_GAPB_DEFAULT
 - Значения параметров, 53
- UEM_SYNC1_ON_COMMAND
 - Значения параметров, 53
- UEM_SYNC1_ON_DATA
 - Значения параметров, 53
- UEM_SYNC1_ON_ERROR
 - Значения параметров, 53
- UEM_SYNC1_ON_GAPB
 - Значения параметров, 53
- UEM_SYNC1_ON_NO_ERROR
 - Значения параметров, 53
- UEM_SYNC1_ON_NO_GAPB
 - Значения параметров, 53
- UEM_SYNC2_VRTA
 - Описание параметров, 39
- UEM_SYNC2_VRTA_DEFAULT
 - Значения параметров, 52
- UEM_SYNC2_VRTA_MAX
 - Значения параметров, 52
- UEM_SYNC2_VRTA_MIN
 - Значения параметров, 52
- UEM_SYNCHRO_ERROR_POS
 - Типы вносимых ошибок кодирования, 20
- UEM_TIME_PARAM
 - Параметры интервалов времени, 56
- UEM_TIME_TAG, 128
 - b, 129
 - days, 128
 - i, 129
 - quartas, 128
 - reserved, 128
 - secs, 128
- uem_time_tag_get
 - Встроенный счетчик времени, 59
- UEM_TIME_TAG_LIN
 - Встроенный счетчик времени, 59
- uem_time_tag_reset
 - Встроенный счетчик времени, 59
- uem_time_tag_set
 - Встроенный счетчик времени, 59
- uem_time_tag_to_linear
 - Встроенный счетчик времени, 60
- uem_time_tag_to_struct
 - Встроенный счетчик времени, 60
- uem_timing_get
 - Параметры интервалов времени, 57
- uem_timing_set
 - Параметры интервалов времени, 56
- UEM_TMT_RES
 - Описание параметров, 36
- UEM_TMTA_DEFAULT
 - Значения параметров, 47
- UEM_TMTA_DIS
 - Описание параметров, 34
- UEM_TMTA_DISABLED
 - Значения параметров, 46
- UEM_TMTA_ENABLED
 - Значения параметров, 47
- UEM_TMTB_DEFAULT
 - Значения параметров, 47

- UEM_TMTB_DIS
 - Описание параметров, 34
- UEM_TMTB_DISABLED
 - Значения параметров, 47
- UEM_TMTB_ENABLED
 - Значения параметров, 47
- UEM_TXA_RFT
 - Описание параметров, 38
- UEM_TXA_VPP
 - Описание параметров, 38
- UEM_TXB_RFT
 - Описание параметров, 38
- UEM_TXB_VPP
 - Описание параметров, 38
- UEM_UEM
 - Действия с дескрипторами, 29
- UEM_UNF
 - Определения типов данных для КИШ, ОУ, МИШ, 66
- UEM_VPP_DEFAULT
 - Значения параметров, 51
- UEM_VPP_MAX
 - Значения параметров, 51
- UEM_VPP_MIN
 - Значения параметров, 51
- UEM_WARN_JUST_IN_STATE
 - Коды завершения, 13
- UEM_WARN_NO_NEXT_MESSAGE
 - Коды завершения, 13
- UEM_WARN_OFFSET
 - Коды завершения, 10
- UEM_WORD
 - Определения примитивных типов, 24
- UEM_ZERO
 - Формирование и разбор командных и ответных слов, 116
- Адресная строка, 14
- Вводные и дополнительные сведения, 8
- Виртуальные устройства, 14
- Внесение ошибок состава сообщения, 70
- Встроенный счетчик времени, 58
 - uem_time_tag_get, 59
 - UEM_TIME_TAG_LIN, 59
 - uem_time_tag_reset, 59
 - uem_time_tag_set, 59
 - uem_time_tag_to_linear, 60
 - uem_time_tag_to_struct, 60
- Действия с дескрипторами, 28
 - UEM_BC, 29
 - UEM_BCP, 29
 - UEM_BM, 29
 - UEM_CSEG, 29
 - uem_handle_type, 29
 - UEM_HANDLE_TYPE, 29
 - UEM_INVH, 29
 - uem_layer_handle, 30
 - uem_parent_dev, 29
 - UEM_RESP, 29
 - uem_root_dev, 30
 - UEM_RT, 29
 - UEM_SEL_UNBASE_INT, 28
 - UEM_SEL_UNMBASE, 28
 - UEM_SEL_UNMUEM, 28
 - UEM_UEM, 29
- Заполнение образа командного сегмента в ОЗУ ПЭВМ, 68
 - uem_bc_cseg_format, 69
 - uem_bc_cseg_format_MODE, 70
 - uem_bc_cseg_format_RTRT, 70
- Запуск и остановка, 104
 - UEM_BC_RUNNING, 105
 - uem_bc_start, 106
 - UEM_BC_START_DEFAULT, 105
 - UEM_BC_START_NOW, 105
 - UEM_BC_START_WAITING, 105
 - uem_bc_state_ex, 107
 - UEM_BC_STATE_EX, 105
 - uem_bc_stop, 107
 - UEM_BC_STOP_DEFAULT, 105
 - UEM_BC_STOP_NOW, 105
 - UEM_BC_STOP_ON_FRAME, 105
 - UEM_BC_STOPPED, 105
 - UEM_BC_WAITING, 105
 - uem_is_running, 106
 - uem_start, 106
 - uem_stop, 106
- Запуск и остановка КИШ, 87
- Значения параметров, 41
 - UEM_BM_WORD_DEFAULT, 54
 - UEM_BM_WORD_MAX, 54
 - UEM_BM_WORD_MIN, 54
 - UEM_BRCST_DEFAULT, 49
 - UEM_BRCST_DISABLED, 49
 - UEM_BRCST_ENABLED, 49
 - UEM_BRTF_DEFAULT, 52
 - UEM_BRTF_DISABLED, 52
 - UEM_BRTF_ENABLED, 52
 - UEM_BTMT_DEFAULT, 52
 - UEM_BTMT_DISABLED, 52
 - UEM_BTMT_ENABLED, 52
 - UEM_DB_ACTIVE, 50
 - UEM_DB_INACTIVE, 50
 - UEM_ERR_INJ_DEFAULT, 50
 - UEM_ERR_INJ_DISABLED, 50
 - UEM_ERR_INJ_ENABLED, 50
 - UEM_IST_DEFAULT, 54
 - UEM_IST_MAX, 54
 - UEM_IST_MIN, 54
 - UEM_MC_DEFAULT, 51
 - UEM_MC_DISABLED, 51
 - UEM_MC_ENABLED, 51
 - UEM_RCVA_DEFAULT, 47
 - UEM_RCVA_DISABLED, 47
 - UEM_RCVA_ENABLED, 47
 - UEM_RCVB_DEFAULT, 48

UEM_RCVB_DISABLED, 47
UEM_RCVB_ENABLED, 47
UEM_RFT_DEFAULT, 51
UEM_RFT_MAX, 51
UEM_RFT_MIN, 51
UEM_RFT_SINE, 51
UEM_SET, 50
UEM_SYNC_IN_1_DEFAULT, 48
UEM_SYNC_IN_1_DISABLED, 48
UEM_SYNC_IN_1_ENABLED, 48
UEM_SYNC_IN_1_INTGEN_DEFAULT, 49
UEM_SYNC_IN_1_INTGEN_DISABLED, 49
UEM_SYNC_IN_1_INTGEN_ENABLED, 49
UEM_SYNC_IN_2_DEFAULT, 48
UEM_SYNC_IN_2_DISABLED, 48
UEM_SYNC_IN_2_ENABLED, 48
UEM_SYNC_IN_2_INTGEN_DEFAULT, 50
UEM_SYNC_IN_2_INTGEN_DISABLED, 50
UEM_SYNC_IN_2_INTGEN_ENABLED, 50
UEM_SYNC_OUT_1_DEFAULT, 49
UEM_SYNC_OUT_1_DISABLED, 48
UEM_SYNC_OUT_1_ENABLED, 48
UEM_SYNC_OUT_2_DEFAULT, 49
UEM_SYNC_OUT_2_DISABLED, 49
UEM_SYNC_OUT_2_ENABLED, 49
UEM_SYNC1_ACH, 54
UEM_SYNC1_C_D_DEFAULT, 53
UEM_SYNC1_CH_A, 53
UEM_SYNC1_CH_B, 54
UEM_SYNC1_CH_DEFAULT, 54
UEM_SYNC1_ERR_DEFAULT, 53
UEM_SYNC1_GAPB_DEFAULT, 53
UEM_SYNC1_ON_COMMAND, 53
UEM_SYNC1_ON_DATA, 53
UEM_SYNC1_ON_ERROR, 53
UEM_SYNC1_ON_GAPB, 53
UEM_SYNC1_ON_NO_ERROR, 53
UEM_SYNC1_ON_NO_GAPB, 53
UEM_SYNC2_VRTA_DEFAULT, 52
UEM_SYNC2_VRTA_MAX, 52
UEM_SYNC2_VRTA_MIN, 52
UEM_TMTA_DEFAULT, 47
UEM_TMTA_DISABLED, 46
UEM_TMTA_ENABLED, 47
UEM_TMTB_DEFAULT, 47
UEM_TMTB_DISABLED, 47
UEM_TMTB_ENABLED, 47
UEM_VPP_DEFAULT, 51
UEM_VPP_MAX, 51
UEM_VPP_MIN, 51

Коды завершения, 8
UEM_ERROR_ADDRESS_OUT_OF_RANGE, 12
UEM_ERROR_BAD_OVERLAY_SOURCE, 12
UEM_ERROR_BAD_PARAM_VALUE, 10
UEM_ERROR_BAD_PARAM_VALUE_1, 10
UEM_ERROR_BAD_PARAM_VALUE_10, 11
UEM_ERROR_BAD_PARAM_VALUE_2, 11
UEM_ERROR_BAD_PARAM_VALUE_3, 11
UEM_ERROR_BAD_PARAM_VALUE_4, 11
UEM_ERROR_BAD_PARAM_VALUE_5, 11
UEM_ERROR_BAD_PARAM_VALUE_6, 11
UEM_ERROR_BAD_PARAM_VALUE_7, 11
UEM_ERROR_BAD_PARAM_VALUE_8, 11
UEM_ERROR_BAD_PARAM_VALUE_9, 11
UEM_ERROR_BAD_TIMEOUT, 12
UEM_ERROR_BCP_NINST, 12
UEM_ERROR_BM_INTERNAL_BUFFER_OVERFLOW, 13
UEM_ERROR_FORMAT_DISABLED, 12
UEM_ERROR_FORMAT_X_MCODE, 12
UEM_ERROR_IN_USE, 13
UEM_ERROR_INC_RESP, 13
UEM_ERROR_INPOOL, 12
UEM_ERROR_INV_HANDLE, 11
UEM_ERROR_INV_HANDLE_TYPE, 11
UEM_ERROR_MAX_SIZE_EXCEED, 13
UEM_ERROR_NO_FRAME_APPEND, 13
UEM_ERROR_NO_FREE_RAM, 11
UEM_ERROR_NO_HOST_MEM, 12
UEM_ERROR_NOT_APPLICABLE, 12
UEM_ERROR_NOT_CONNECTED, 12
UEM_ERROR_NUMBER_OUT_OF_RANGE, 12
UEM_ERROR_OFFSET, 10
UEM_ERROR_THREAD_FAULT, 13
UEM_ERROR_TOO_MANY_DATAWORDS, 13
UEM_ERROR_WRONG_LOCATION, 12
UEM_WARN_JUST_IN_STATE, 13
UEM_WARN_NO_NEXT_MESSAGE, 13
UEM_WARN_OFFSET, 10

Командные и ответные сегменты, 14
Описание параметров, 32
UEM_BM_WORD_MASK, 41
UEM_BM_WORD_PATTERN, 41
UEM_BRCST_DIS, 36
UEM_BRTF_DIS, 39
UEM_BTMT_DIS, 39
UEM_DB_ACT, 37
UEM_ERR_INJ_DIS, 36
UEM_IST1, 40
UEM_IST2, 40
UEM_MC_DIS, 38
UEM_RCVA_DIS, 34
UEM_RCVB_DIS, 34
UEM_SYNC_IN_1_ENA, 35
UEM_SYNC_IN_1_INTGEN, 36
UEM_SYNC_IN_1_SET, 37
UEM_SYNC_IN_2_ENA, 35
UEM_SYNC_IN_2_INTGEN, 36
UEM_SYNC_IN_2_SET, 37
UEM_SYNC_OUT_1_ENA, 35
UEM_SYNC_OUT_1_SET, 37
UEM_SYNC_OUT_2_ENA, 35
UEM_SYNC_OUT_2_SET, 37
UEM_SYNC1_C_D_, 39

- UEM_SYNC1_CH, 40
- UEM_SYNC1_ERR, 39
- UEM_SYNC1_GAPB, 40
- UEM_SYNC2_VRTA, 39
- UEM_TMT_RES, 36
- UEM_TMTA_DIS, 34
- UEM_TMTB_DIS, 34
- UEM_TXA_RFT, 38
- UEM_TXA_VPP, 38
- UEM_TXB_RFT, 38
- UEM_TXB_VPP, 38
- Определения примитивных типов, 23
 - UEM_BOOL, 23
 - UEM_DEVHANDLE, 23
 - UEM_DWORD, 23
 - UEM_OBJHANDLE, 23
 - UEM_PARAMID, 23
 - UEM_WORD, 24
- Определения типов данных для КШ, ОУ, МШ, 61
 - UEM_BCRT, 63
 - UEM_BCRTb, 63
 - UEM_CH_A, 66
 - UEM_CH_B, 66
 - UEM_CHANNEL, 66
 - UEM_ERRF_DT, 66
 - UEM_ERRF_ENC, 66
 - UEM_ERRF_ENCODING, 64
 - UEM_ERRF_ENCODING2, 66
 - UEM_ERRF_ERROR, 64
 - UEM_ERRF_EXTRA_CWSW, 65
 - UEM_ERRF_EXTRA_DW, 65
 - UEM_ERRF_FORMAT, 64
 - UEM_ERRF_FORMAT_MC, 65
 - UEM_ERRF_GAPN, 65
 - UEM_ERRF_INC_MODE_CODE, 65
 - UEM_ERRF_INCORRECT_RTN, 65
 - UEM_ERRF_LESS_BITS, 65
 - UEM_ERRF_MINGAP, 64
 - UEM_ERRF_MISSING_CWSW, 65
 - UEM_ERRF_MISSING_DW, 65
 - UEM_ERRF_MORE_BITS, 65
 - UEM_ERRF_NO_RESPONSE, 64
 - UEM_ERRF_PARITY, 65
 - UEM_ERRF_RTRT_FORMAT, 65
 - UEM_ERRF_SYNC_TYPE, 64
 - UEM_ERROR_FLAGS, 66
 - UEM_F1, 66
 - UEM_F10, 66
 - UEM_F2, 66
 - UEM_F3, 66
 - UEM_F4, 66
 - UEM_F5, 66
 - UEM_F6, 66
 - UEM_F7, 66
 - UEM_F8, 66
 - UEM_F9, 66
 - UEM_FORMAT, 66
 - UEM_MC, 63
 - UEM_MCb, 64
 - UEM_MCBCRT, 63
 - UEM_MCBCRTb, 64
 - UEM_MCRTBC, 63
 - UEM_NDATA_BY_CW, 64
 - UEM_NDATA_MAX, 64
 - UEM_NDATA_MIN, 64
 - UEM_RTBC, 63
 - UEM_RTRT, 63
 - UEM_RTRTb, 64
 - UEM_SYNC, 67
 - UEM_SYNC_C, 67
 - UEM_SYNC_D, 67
 - UEM_UNF, 66
- Параметры интервалов времени, 55
 - UEM_MIN_T1, 56
 - UEM_MIN_T1_DEFAULT, 56
 - UEM_MIN_T2, 56
 - UEM_MIN_T2_DEFAULT, 56
 - UEM_RTMO, 56
 - UEM_RTMO_DEFAULT, 56
 - UEM_TIME_PARAM, 56
 - uem_timing_get, 57
 - uem_timing_set, 56
- Параметры конфигурации УЭМ, 31
 - uem_param_get, 31
 - uem_param_set, 32
- Параметры передатчиков и характеристики выходных сигналов, 15
- Передача сообщений, 88
 - uem_bc_send_receive, 89
- Порядок действий при установлении связи с устройством, 13
- Служебные функции, 108
 - uem_error_message, 108
 - uem_error_query, 109
 - UEM_LIB_REV, 108
 - uem_reset, 109
 - uem_revision_query, 109
 - uem_self_test, 110
- Создание и настройка кадров и программы КШ, 81
 - uem_bcp_append_cseg, 84
 - uem_bcp_append_frame, 84
 - uem_bcp_create, 84
 - UEM_BCP_CUR_SIZE, 83
 - uem_bcp_desrtoy, 86
 - uem_bcp_dimension, 85
 - uem_bcp_discover_cseg, 85
 - uem_bcp_inspect_frame, 86
 - uem_bcp_install, 86
 - UEM_BCP_MAX_SIZE, 83
 - UEM_BCP_NFRAMES, 83
 - uem_bcp_replace_cseg, 85
 - uem_bcp_set_standard_gaps, 87
 - UEM_FRAME_ALLRPT, 83
 - UEM_FRAME_CONT, 83

- UEM_FRAME_DEFAULT, 83
- UEM_FRAME_NONE, 83
- UEM_FRAME_REPEAT_DEFAULT, 82
- UEM_FRAME_REPEAT_MAX, 82
- UEM_FRAME_REPEAT_MIN, 82
- UEM_FRAME_REPEAT_UNLIM, 82
- UEM_FRAME_STOP, 83
- Создание и настройка командных сегментов, 71
 - uem_bc_cseg_create, 75
 - uem_bc_cseg_overlay, 79
 - uem_bc_gap_create, 79
 - uem_cseg_desrtoy, 80
 - uem_cseg_error_get, 78
 - uem_cseg_error_set, 77
 - UEM_CSEG_GAP, 75
 - UEM_CSEG_GAP_ALT_AB, 74
 - UEM_CSEG_GAP_ALT_BUS, 74
 - UEM_CSEG_GAP_DEFAULT_FLAGS, 74
 - UEM_CSEG_GAP_DEFAULT_TIMEOUT, 75
 - UEM_CSEG_GAP_DEFAULT_VALUE, 74
 - UEM_CSEG_GAP_ESYNC, 73
 - UEM_CSEG_GAP_FROM_END, 73
 - UEM_CSEG_GAP_FROM_START, 73
 - uem_cseg_gap_get, 76
 - UEM_CSEG_GAP_MAX, 74
 - UEM_CSEG_GAP_MIN, 74
 - uem_cseg_gap_reset, 76
 - uem_cseg_gap_set, 76
 - UEM_CSEG_GAP_THIS_AB, 74
 - UEM_CSEG_GAP_THIS_BUS, 74
 - UEM_CSEG_GAP_TIMEOUT_MAX, 75
 - UEM_CSEG_GAP_TIMEOUT_MIN, 74
 - UEM_CSEG_NORMAL, 75
 - UEM_CSEG_OVERLAY, 75
 - uem_cseg_read, 75
 - uem_cseg_sync_get, 78
 - uem_cseg_sync_set, 78
 - uem_cseg_type, 80
 - UEM_CSEG_TYPE, 75
 - uem_cseg_word_gap_get, 77
 - uem_cseg_word_gap_set, 77
- Типы вносимых ошибок кодирования, 17
 - UEM_BAD_SYNCHRO_EEEEEI, 21
 - UEM_BAD_SYNCHRO_EEEEEIE, 21
 - UEM_BAD_SYNCHRO_EEEIEE, 21
 - UEM_BAD_SYNCHRO_EEIEEE, 21
 - UEM_BAD_SYNCHRO_EIEEEE, 21
 - UEM_BAD_SYNCHRO_IEEEEE, 21
 - UEM_BAD_SYNCHRO_NONE, 21
 - UEM_BAD_SYNCHRO_NONE2, 21
 - UEM_BIPHASE_POS_MAX, 18
 - UEM_BIPHASE_POS_MIN, 18
 - UEM_BITCOUNT_CHANGE_MAX, 18
 - UEM_BITCOUNT_CHANGE_MIN, 18
 - UEM_ERROR_PARAM_DEFAULT, 19
 - UEM_ERROR_POS_DEFAULT, 19
 - UEM_ERROR_TYPE, 19
 - UEM_ERROR_TYPE_DEFAULT, 19
 - UEM_ERRT_BAD_BIPHASE_NEG, 20
 - UEM_ERRT_BAD_BIPHASE_POS, 20
 - UEM_ERRT_BAD_BIPHASE_ZERO, 20
 - UEM_ERRT_BAD_SYNCHRO, 19
 - UEM_ERRT_INV_PARITY, 19
 - UEM_ERRT_NONE, 19
 - UEM_ERRT_SHIFT_EDGE, 20
 - UEM_ERRT_WRONG_BITCOUNT, 19
 - UEM_SHIFT_LENGTH_MAX, 19
 - UEM_SHIFT_LENGTH_MIN, 19
 - UEM_SHIFT_POS_MAX, 19
 - UEM_SHIFT_POS_MIN, 18
 - UEM_SYNCHRO_ERROR_POS, 20
- Управление синхронизацией, 21
- Установление и разрыв связи с устройством, 25
 - uem_bc_init, 26
 - uem_bm_init, 27
 - uem_close, 27
 - uem_connect, 26
 - uem_init, 25
 - uem_rt_init, 27
- Формирование и разбор командных и ответных слов, 111
 - UEM_ABBUSY, 116
 - UEM_ABFAIL, 116
 - UEM_BCCA, 116
 - uem_command_word, 116
 - uem_command_word_parse, 117
 - UEM_DBCA, 115
 - UEM_MCODE_ADBC, 114
 - UEM_MCODE_BRTF, 115
 - UEM_MCODE_BSELFTEST, 114
 - UEM_MCODE_BTMT, 114
 - UEM_MCODE_BTMT_I, 115
 - UEM_MCODE_RESETRT, 115
 - UEM_MCODE_SYNCHRO, 114
 - UEM_MCODE_SYNCHRO_D, 115
 - UEM_MCODE_TXBIT, 115
 - UEM_MCODE_TXLCMD, 115
 - UEM_MCODE_TXSTATUS, 114
 - UEM_MCODE_TXVECT, 115
 - UEM_MCODE_UBRTF, 115
 - UEM_MCODE_UBTMT, 115
 - UEM_MCODE_UBTMT_I, 115
 - UEM_MODE_0, 114
 - UEM_MODE_31, 114
 - UEM_MSGERR, 116
 - UEM_RSV12, 116
 - UEM_RSV13, 116
 - UEM_RSV14, 116
 - UEM_RT_RX, 113
 - UEM_RT_TX, 113
 - UEM_RTADDR_BRCST, 113
 - UEM_RTADDR_MAX_EXT, 113
 - UEM_RTADDR_MIN, 113
 - UEM_RTADDR_MAX, 113

UEM_RTFAIL, 115
UEM_SADDR_CONV_LOOPBACK, 114
UEM_SADDR_MAX, 114
UEM_SADDR_MAX_EXT, 114
UEM_SADDR_MIN, 114
UEM_SADDR_MIN_EXT, 114
UEM_SERVRQ, 116
uem_status_word, 117
uem_status_word_parse, 117
UEM_ZERO, 116

Функции КШ, 68
Функции МШ, 102
uem_bm_handler, 102
uem_bm_install_handler, 103
uem_bm_queue_count, 103
uem_bm_receive, 102

Функции ОУ, 90
uem_response_create, 95
uem_response_destroy, 101
uem_response_error_get, 98
uem_response_error_set, 97
uem_response_gap_get, 96
uem_response_gap_set, 96
uem_response_read, 96
uem_response_sync_get, 98
uem_response_sync_set, 98
uem_response_word_gap_get, 97
uem_response_word_gap_set, 97
uem_rt_discover_response, 100
uem_rt_discover_response_MODE, 100
uem_rt_install_response, 99
uem_rt_install_response_MODE, 99
UEM_RTDES_COM_ILLEGAL, 92
UEM_RTDES_DBCA, 92
UEM_RTDES_DBCA_BCSTART, 92
UEM_RTDES_DEFAULT, 95
UEM_RTDES_ILLEG_MASK, 95
UEM_RTDES_LCMD_DW, 92
UEM_RTDES_SW_DIS, 91
UEM_RTDES_SWB_SAV, 92
UEM_RTDES_WA, 94
UEM_RTDES_WA_BRCST, 95
UEM_RTDES_WRONG_CH, 94

ПЕРЕЧЕНЬ ПРИНЯТЫХ СОКРАЩЕНИЙ И ОБОЗНАЧЕНИЙ

АЗ	Абонент занят
ВСК	Встроенный самоконтроль
ЗО	Запрос обслуживания
КС	Командное слово
КУ	Команда управления
КШ	Контроллер шины
КШОУ	Тип сообщения – от КШ к ОУ
МЗР	Младший значащий разряд
МК	Мультиплексный канал
МКПД	Мультиплексный канал передачи данных
МШ	Монитор шины
НА	Неисправность абонента
НОУ	Неисправность ОУ
НП	Непосредственное подключение
НС	Нормальное состояние
НФ	Неформатное сообщение
ОЗУ	Оперативное запоминающее устройство
ОО	Отсутствие ответа
ОС	Ответное слово
ОУ	Оконечное устройство
ОУКШ	Тип сообщения – от ОУ к КШ
ОУОУ	Тип сообщения – от ОУ к ОУ
ОШС	Ошибка в сообщении
ПА	Поадрес
ПГК	Принята групповая команда
ПО	Программное обеспечение
ПУИ	Принято управление интерфейсом
ПЭВМ	Персональная электронная вычислительная машина
РЕЗ	Резервные признаки ответного слова
СД	Слово данных
ТП	Трансформаторное подключение
УЭМ	Универсальный электронный модуль

ПЕРЕЧЕНЬ ССЫЛОЧНЫХ ДОКУМЕНТОВ

1. ГОСТ Р 52070-2003. Интерфейс магистральный последовательный системы электронных модулей. Общие требования.
2. ЮФКВ.469555.555РЭ. Универсальный электронный модуль УЭМ-МК. Руководство по эксплуатации.
3. ЮФКВ.469555.731РЭ Модуль МВ98.03. Руководство по эксплуатации.
4. ФТКС.76902-01 32 01. ДРАЙВЕР НМ. Руководство системного программиста.
5. ЮФКВ. 10149-01 32 01. Универсальный электронный модуль УЭМ-МК. Программное обеспечение системное. Руководство системного программиста.
6. ЮФКВ.10175-01 32 01. Модуль МВ98.03. Программное обеспечение системное. Руководство системного программиста.

