LedMX
Контроллер для LED-экранов
и
бегущих строк

Для F/W:4.0

Оглавление

Оглавление 2
1. Технические характеристики 3
1.1. Сводная таблица характеристик 3
1.2. Поддерживаемые драйвера LED-панелей 4
1.3. Параметры передачи данных: 4
2. Начало работы 5
2.1. Настройка 5
2.2. Обновление прошивки 5
2.2.1. Обновление через файл 6
2.2.2. Обновление через конфигуратор (виртуальный COM-порт) 6
2.2.3. Аварийный режим обновления 7
3. Конфигурационные файлы 8
3.1. Файл IO*.txt 9
3.2. Файл Driver*.drv 9
3.3. Файл User*.txt 10
3.3.1. Блок “ScreenSettings” 10
3.3.2. Блок “Defaults” 10
3.3.3. Блок “AreasCollection” 12
3.3.4. Блок “Screens” 13
3.3.5. Блок “Scenes” 16
3.3.6. Блок “Strings” 17
3.3.7. Блок “Images” 17
3.3.8. Блок “Vars” 18
3.3.9. Блок “Bridge_n” 20
3.3.10. Пример файла для демонстрации или рекламы 21
4. Регистры ModBus 23
4.1. Формат данных 24
4.2. Командный регистр 25
4.3. Описание регистров 28
4.4. Таблица регистров 45
5. Маркировка устройств / Информация для заказа 47
5.1. Контроллер LedMX 47
5.2. LED-панель 47
5.2.1. Короткая маркировка (для заказа) 47
5.2.2. Длинная маркировка (для подбора драйвера) 48
5.3. Панель оператора 49
Приложение A. Формат JSON 51
Приложение B. Протокол Modbus 53
Режим RTU 53
Регистры ModBus 54
Расчет контрольной суммы (CRC-16) 55
Передача пакета 55
Прием пакета 56
Приложение C. Таблицы кодирования текстовой информации 57
Основная таблица ASCII-символов 57
Расширенная таблица ASCII-символов. Кодировка Windows-1251 58

1. Технические характеристики

- Питание: постоянное напряжение 5В 1Вт. Максимальные значения от 3.7В до 5.6В,
- RS-485: Защита от перенапряжения (от -5В до +10В) и короткого замыкания,
- RS-232: Защита от перенапряжения (до +/-20В) и короткого замыкания,
- USB: High-speed 2.0. Тип разъема: Mini-USB,
- Температура хранения от -20’С до 50’С,
- Температура эксплуатации от -10’С до 45’С и влажности не более 90% при 20’С без образования конденсата.

Важно! Устройство боится электростатических разрядов. Нельзя класть устройство без корпуса на пакеты.

Эксплуатация или хранение устройства в условиях отличных от описанных выше ведет к потере гарантии.

Гарантийные обязательства распространяются только на устройства с заводским браком, при отсутствии следов неквалифицированного ремонта.

Установка неофициальных прошивок недопустима и влечет за собой лишение гарантии.

1.1. Сводная таблица характеристик

	Версия контроллера				
	Mono	**Lite**	**OEM**	**Custom**	**PRO**
Макс. кол-во LED-панелей*	1 шт	до 80 шт			
Разрешение экрана**	3200px	8192px при 5 бит на цвет, 40960px при 1 бит на цвет			
Кол-во цветов	1Bit-8, 2Bit-64, 3Bit-512, 4Bit-4096, 5Bit-32768				
Макс. частота обновления экрана***	400Гц				
Типы сканирования	Static/S1, S2, S4, S8, S16, S32				
Драйвера LED-панелей******	74HC595, DP5020B, ICND2037, ICND2038S, ICND2045N, SM16207S, SM16237, SM16227, SM16259, SM16188, … И их аналоги				
Порты LED-панелей	HUB75E, HUB08 (по запросу), HUB12 (по запросу)				
Макс. кол-во переменных ModBus	33 для H/W:3A.E27, 12 для H/W:3A.C27				
Макс. кол-во областей в коллекции/строк в экране	32 для H/W:3A.E27, 11 для H/W:3A.C27				
Память для строк (символов)	2000				
Макс. кол-во аргументов в строке	5				
Форматы картинок	PNG, GIF с поддержкой прозрачности				
Порты связи	USB (Mass Storage, Virtual COM-port), RS-485				USB, RS-485, RS-232
Протоколы	ModBus Master/Slave				

* Количество панелей приведено для случая когда разрешение каждой панели 32*16px на стандартном драйвере типа 74HC595, 1 бит на цвет.

** Указано максимальное кол-во пикселей на экране (разрешение экрана). Оно может отличаться в зависимости от типа подключенной LED-панели. Соотношение сторон может быть практически любым.
*** С увеличением разрешения экрана частота обновления падает.
**** Для корректной работы рекомендуются применять LED-панели с соответствующим файлом драйвера. Более полный список см. в разделе Поддерживаемые драйвера LED-панелей.

1.2. Поддерживаемые драйвера LED-панелей

Список содержит микросхемы управления светодиодами (драйвера), которые поддерживает контроллер LedMX. Список не полный, так как существует множество аналогов уже описанных микросхем, но с отличающейся маркировкой.

- 74HC595, DP5020B, ICND2037, SM16206S, SM16207S,
- ICND2038S,
- ICND2045N/ICND2038/SM16227,
- ICND2046AP/SM16237DS,
- ICND2055CP, SM16259, MBI5152, SM16188.

Поддерживаемые драйверы строк:

- 74HC138, 2013EP,
- SM5266,
- и их аналоги.

1.3. Параметры передачи данных:

- Порты RS-485/RS-232. Скорость передачи данных от 1200 до 115200 бод/с, 8 бит, четность: нет/чет./нечет., 1 или 2 стоповых бита. Максимальное количество регистров принимаемое/передаваемое за одну команду: 26. Протоколы: ModBus Master/Slave. Адрес устройства в сети ModBus от 1 до 247. По умолчанию используются параметры: 9600, 8/n/1, адрес 10.
- Порт USB (виртуальный COM-порт VCP). Максимальное количество регистров принимаемое/передаваемое за одну команду: 26. Протоколы: ModBus Slave. Адрес устройства в сети ModBus от 1 до 247. По умолчанию адрес 10. Скорость передачи, четность, количество бит и стоп-бит для виртуального порта не актуальны (передача работает при любых значениях).

2. Начало работы

Перед началом работы с устройством на него следует установить последнюю версию программного обеспечения (процесс обновления описан ниже).

Файлы с настройками, примеры использования и другая техническая информация всегда доступны на веб-сайте компании по адресу: cyberidea.io

Версия устройства (**Mono, Lite, PRO, Custom, OEM**), и версия программного обеспечения будут отображены на экране сразу после включения.

2.1. Настройка

Для конфигурирования устройства следует использовать кабель USB, операционная система определит новые устройства:

- CI Virtual COM Port - (не для всех версий устройства) виртуальный COM-порт, для полноценной работы нужен фирменный драйвер «VCP Driver»,
- Виртуальный диск с названием «Settings».

Все настройки хранятся на виртуальном диске в виде текстовых файлов. Настройки разделены на функциональные блоки. Если определенный функционал не требуется, то файл с настройками можно не копировать на устройство. Подробное описание файлов с настройками см. в главе 3.Конфигурационные файлы.

2.2. Обновление прошивки

Новые версии прошивок выпускаются при расширении функционала и внесении исправлений. Некоторые обновления обязательны, некоторые рекомендуются. Описание изменений по версиям прошивок всегда можно посмотреть в файле Errata.txt в архиве с прошивкой. Обновить прошивку возможно несколькими способами, описанными ниже.

Перед обновлением убедитесь, что версия прошивки соответствует версии вашей платы, указанной на шильдике. Например версия платы "H/W:3A.C27", ей соответствует прошивка, в названии которой указано “3A.C27". Тип лицензии (Mono/Lite/PRO) не важен.

- *ВНИМАНИЕ! Если на шильдике **отсутствует** маркировка версии платы ("H/W:##.###"), то следует выбирать прошивку из предыдущих версий, исходя из ревизии платы (указано на самой плате "Rev#.#") и типа лицензии (указан на шильдике).*
- *ВНИМАНИЕ! Если на шильдике указана версия прошивки ("F/W:#.#"), то **установка версии ниже указанной недопустима** и может привести к сбросу лицензии или нестабильной работе устройства. Если вы ошиблись с версией прошивки, то просто обновитесь на версию соответствующую вашему устройству.*
- *Пользователь может по своему усмотрению перейти на любую из доступных версий прошивки, но рекомендуется устанавливать самую свежую версию.*
- *Установка неофициальных прошивок недопустима и влечет за собой лишение гарантии.*
- *Если в процессе обновления прошивки произошел сбой питания или устройство не отвечает более 2-х минут, то следует заново обновить прошивку, как описано в главе "Аварийный режим обновления".*

- *После загрузки в режиме обновления важно чтобы на виртуальный диск не копировались никакие файлы кроме самой прошивки. Если на диск попал любой файл не относящийся к прошивке (системные файлы или файлы пользователя), то процесс обновления может быть некорректен. Следует выключить питание устройства и заново начать обновление одним из описанных ниже способов.*
- *Для корректного обновления прошивки рекомендуется операционная система Windows XP или Windows 7.*

2.2.1. Обновление через файл

- Установить связь между устройством и компьютером через порт USB,
- Скопировать на виртуальный диск устройства файл “FW_Update.txt”,
- Выключить питание и снова включить его,
- Устройство перейдет в режим обновления прошивки. Светодиод 2 начнет циклично мигать: 3 коротких вспышки - 3 длинных вспышки - 3 коротких вспышки,
- На компьютере появится новый диск с названием «FW Update»,
- Скопировать файл прошивки с расширением .upd на диск,
- Устройство прошивается во время копирования файла, не прерывайте процесс. Нельзя отключать кабель USB или питание,
- Светодиоды могут перестать мигать на время до 2-х минут, при этом диск для обновления прошивки пропадет из системы, это нормально,
- После окончании процесса обновления, устройство перезагрузится и начнет подмигивать как положенно для данной модели,
- Обновление прошивки завершено, можно продолжить обычное использование устройства. Если в процессе обновления произошел сбой, работа устройства в штатном режиме будет невозможна, тогда следует перейти к режиму аварийного обновления прошивки.

2.2.2. Обновление через конфигуратор (виртуальный COM-порт)

- Подать питание,
- Установить связь между устройством и компьютером через порт USB. Установить драйвера виртуального последовательного порта (VCP) при необходимости.
- Запустить программу Конфигуратор, выбрать порт, к которому подключено устройство, и нажать кнопку «Connect». В списке справа выбрать появившееся устройство. Отобразится окно настроек,
- Нажать на кнопку «Update Firmware», после чего устройство перейдет в режим обновления прошивки. Светодиод 2 начнет циклично мигать: 3 коротких вспышки - 3 длинных вспышки - 3 коротких вспышки,
- На компьютере появится новый диск с названием «FW Update»,
- Скопировать файл прошивки с расширением .upd на диск,
- Устройство прошивается во время копирования файла, не прерывайте процесс. Нельзя отключать кабель USB или питание,
- Светодиоды могут перестать мигать на время до 2-х минут, при этом диск для обновления прошивки пропадет из системы, это нормально,
- После окончании процесса перепрошивки, устройство перезагрузится и начнет подмигивать как положенно для данной модели,

- Обновление прошивки завершено, можно продолжить обычное использование устройства. Если в процессе обновления произошел сбой, работа устройства в штатном режиме будет невозможна, тогда следует перейти к режиму аварийного обновления прошивки.

2.2.3. Аварийный режим обновления

В следствии неправильной эксплуатации или сбоя в процессе установки прошивки штатный вход в процедуру обновления будет невозможен. Для аварийной прошивки устройства необходимо:

- Снять питание с устройства,
- Нажать кнопку Reset (при наличии), либо разобрать корпус и замкнуть контакты Reset на печатной плате,
- Подать питание на устройство, после чего оно перейдет в режим обновления прошивки. Светодиод 2 начнет циклично мигать: 3 коротких вспышки - 3 длинных вспышки - 3 коротких вспышки,
- Отпустить кнопку или снять перемычку с контактов,
- Установить связь между устройством и компьютером через USB-порт. На компьютере появится новый диск с названием «FW Update»,
- Скопировать файл прошивки с расширением .upd на диск,
- Устройство прошивается во время копирования файла, не прерывайте процесс. Нельзя отключать кабель USB или питание,
- Светодиоды могут перестать мигать на время до 2-х минут, при этом диск для обновления прошивки пропадет из системы, это нормально,
- После окончании процесса обновления, устройство перезагрузится и начнет подмигивать как положенно для данной модели,
- Обновление прошивки завершено, можно продолжить обычное использование устройства. Если в процессе обновления произошел сбой, работа устройства будет невозможна, тогда следует начать процедуру аварийного обновления прошивки сначала.

3. Конфигурационные файлы

Все настройки хранятся на виртуальном диске в виде текстовых файлов, и разделены на функциональные блоки. Если определенный функционал не требуется, то файл с настройками можно не копировать на устройство. Имена файлов состоят из обязательной части, части имени, которую может изменить пользователь и расширения файла. Далее будут приводиться только обязательные части названия, а пользовательская часть будет обозначена символом “*”. Например “IO*.txt” - означает, что обязательная часть это “IO”, символ “*” может быть заменен на символы пользователя, либо опущен, и расширение файла “txt”. Файл с названием “IO_LedMX_MbSlave_485.txt” имеет пользовательскую часть “LedMX_MbSlave_485” - это помогает лучше понимать назначение и содержимое этого файла. Общая длина названия файла ограничена 50 символами, может содержать символы русского и английского алфавитов, целые числа, символ пробела “ “, символ дефиса “-“ и символ нижнего подчеркивания “_”.

Настройки вступят в силу сразу после перезагрузки устройства.

Файлы доступные для конфигурирования:

- FW_Update.txt - файл для перехода в режим обновления прошивки. Для этого нужно скопировать файл на виртуальный диск и перезапустить устройство, файл удалится, а устройство перейдет в режим обновления. Для возврата к работе достаточно обновить прошивку или снова перезапустить устройство,
- IO*.txt - параметры связи с устройством, скорость передачи данных, протокол, адрес в сети ModBus и др.,
- Driver*.drv - файл драйвера для светодиодной панели. Более подробное описание в соответствующей главе.,
- User*.txt - файл с пользовательскими настройками. Более подробное описание в соответствующей главе.

Файлы имеют текстовый формат и открываются любым текстовым редактором, описание формата см. Приложение А. Формат JSON. После внесения изменений в файлы настроек, их следует сохранить на виртуальном диске и перезапустить устройство (сбросить питание). Для доступа к виртуальному диску следует подключить устройство к ПК через USB-кабель. Это не обычная «флешка», на нее нельзя копировать файлы, кроме тех, что предназначены для устройства. «Безопасное извлечение» после работы не обязательно.

- *Если во время записи файлов с настройками произошло отключение питания, то операционная система ПК может перестать корректно работать с виртуальным диском. Для устранения этой проблемы необходимо отформатировать диск: зайти в «Мой компьютер» и нажать правой кнопкой мыши на виртуальном диске, выбрать пункт меню «Форматировать». В окне форматирования диска* ***СНЯТЬ галочку «Быстрое форматирование»****, и начать процесс. Система может сообщить, что не удается завершить форматирование, следует проигнорировать сообщение и закрыть окно форматирования. Устройство снова готово к работе. Теперь необходимо заново скопировать файлы настроек и перезапустить устройство.*
- *В некоторых версиях прошивок после обновления также требуется произвести форматирование диска. Перед обновлением внимательно прочитайте файл Errata.txt в архиве с прошивкой.*

3.1. Файл IO*.txt

Файл содержит параметры связи, скорость передачи данных, протокол, адрес в сети ModBus и другие параметры для каждого порта передачи данных. Если в данном разделе не указано иное, то всегда используется настройка для порта “COM_1”:

```
"COM_1":        // Настройки для порта COM_1 (номер аппаратного порта устройства)
{
        "BaudRate":9600,                // Скорость передачи данных 1200, 2400, 4800, 9600, 19200,
                                                38400,  57600, 115200
        "BitsCount":8,                  // Количество бит данных 7, 8 или 9
        "Parity":"n",                   // Четность n-нет, e-чет., o-нечет.
        "StopBits":1,                   // Количество стоповых бит 1 или 2
        "ModBus_Address":10,            // Адрес устройства в сети ModBus (для ModBus Slave)

        "ModBus_Master":false,          // Режим работы ModBus: false-Slave, true-Master
        "ModBus_Timeout_10ms":10,       // Таймаут ожидания ответа от ведомого (для ModBus Master)
        "ModBus_Try_Count":3,           // Количество попыток обращения до объявления ошибки (для
                                                ModBus Master)
        "ModBus_Silent":false,          // Тихий режим (для ModBus Slave). true-устройство будет
                                                принимать команды от ведущего в сети, но не будет
                                                передавать ответ. Применяется для “дублирования”
                                                информации с основного экрана. Параметры связи и
                                                адрес сети должны совпадать с дублируемым экраном.
},
```

- *Название блока для настройки порта (например “COM_1”) относится именно к нумерации аппаратных портов самого устройства и не имеет отношения к нумерации или названию портов при подключении к ПК.*
- *Порт “VCP” (при наличии) определяется на ПК как виртуальный последовательный порт. Работает всегда как ModBus Slave.*

3.2. Файл Driver*.drv

Файл драйвера для светодиодной панели. Название файла должно соответствовать модели светодиодной панели, указанном на фирменном шильдике, на оборотной стороне панели.

- *Начиная с версии прошивки 3.7 формат драйверов и маркировка панелей изменились. Теперь применяются файлы драйвера с расширением “drv” (предыдущие версии имели расширение“txt”).*
- *Старый формат маркировки панелей xxxxxx-xxxxxxxxxxYYYY (22 символа с дефисом), новый xxxxxx-xxxxxxxxxxYYABB (23 символа с дефисом), причем символы BB могут отсутствовать.*
- *Для правильной работы на диске должно быть не более одного файла драйвера.*

3.3. Файл User*.txt

Файл с настройками режимов работы и другими пользовательскими данными. Некоторые блоки и параметры могут быть применены во всех режимах работы, другие только в определенных, например есть блоки и параметры полезные только в режиме ModBus Master. Набор блоков и параметров можно свободно менять, соблюдая синтаксис и разумную необходимость использования каждого параметра. Например нет необходимости описывать для каждой строки размер шрифта, выравнивание и скорость прокрутки, достаточно один раз задать их в блоке “Defaults”, данное правило действительно для всех параметров доступных для блока “Defaults”. Дополнительное описание в строках, нужно только если эти параметры будут отличаются от заданных по умолчанию.

> *Для правильной работы на диске должно быть не более одного пользовательского файла.*

3.3.1. Блок “ScreenSettings”

Блок применяется во всех режимах работы. Предназначен для описания основных параметров экрана:

- **“PanelsOn_X”:1,** - количество панелей физически расположенных по оси X,
- **“PanelsOn_Y”:1,** - количество панелей физически расположенных по оси Y,
- **“Gamma”:128,** - параметр влияющий на яркость экрана и градации цветов. Принимает значения от 0 (минимальная яркость) до 320 (максимальная яркость, но отсутствуют полутона цветов). По умолчанию 128 - это дает наилучший результат яркость/качество для картинок,
  > *При увеличении яркости также будет расти потребляемый ток, но не более указанной мощности на шильдике светодиодной панели.*

- **“BitsPerColor”:5,** - количество бит информации на каждый цвет. 1Bit-8 цветов, 2Bit-64, 3Bit-512, 4Bit-4096, 5Bit-32768 цветов.
  > *Чем больше битность цвета, тем лучше цветопередача экрана, но тем меньше максимальное разрешение экрана (доступное пользователю количество точек).*

Пример:

```
"ScreenSettings":
{
        "PanelsOn_X":1,
        "PanelsOn_Y":1,
        "Gamma":128,
        "BitsPerColor":5,
},
```

3.3.2. Блок “Defaults”

Блок применяется во всех режимах работы. Предназначен для описания параметров применяемых по умолчанию:

- **“OEM_Brand_1”:”txt”,** - *только для версии OEM*. Строка с названием вашего бренда, отображаемая при загрузке контроллера, строка 1. Прокрутка строки невозможна,

- **“OEM_Brand_2”:”txt”,** - *только для версии OEM*. Строка с названием вашего бренда, отображаемая при загрузке контроллера, строка 2. Прокрутка строки невозможна,
- **“PauseBeforeScrollingString”:500,** - (мс) пауза перед началом прокрутки длинных строк,
- **“SpeedScrollingString”:”1”,** - скорость прокрутки длинных строк. Рабочие значения от 10 до 1 и от -1 до масштаба заданного для оси X в FontSize, по 20мс. Это значит, что если задано значение 10, то строка будет смещаться на 1 пиксель каждые 10 * 20 = 200мс, и текст будет двигаться медленно. Если задано значение 1, то строка будет смещаться на 1 пиксель каждые 1 * 20 = 20мс, и соответственно текст будет двигаться быстро. Значения со знаком “-“ дают еще большую скорость и ограниченны только выбранным масштабом текста по оси X,
- **“SpeedScrollingPerforateFrame”:”1”,** - скорость прокрутки перфорированной рамки,
- **“FontSize”:”0x11”,** - размер шрифта по осям X/Y. Имеет вид 0xXY. Где X это масштаб по оси X, Y это масштаб по оси Y. Стандартный символ имеет размер 6 пикселей в ширину (ось X) и 8 пикселей в высоту (ось Y). Если задан масштаб 0x11, то символу будет обычного размера 6*8. Если масштаб: 0x21, то размер 12*8, если 0x12, то размер 6*16, если 0x33, то размер 18*24, и так далее. По каждой оси множитель может быть от 1 до F. Если масштаб задан одной цифрой например 0x08, то он будет принят как масштаб 0x88. Запись вида: 0xN0, где число N задано, а после стоит 0 считается недопустимой,
- **“Align”:”0x22”,** - выравнивание текста по осям X/Y. Имеет вид 0xXY. Где X может принимать значения: 1-выравнивание по левому краю, 2-по центру, 3-по правому краю. Y может принимать значения: 1-по верхнему краю, 2-по центру, 3-по нижнему краю. Если текст или картинки непомещаются по высоте в выделенную область, то выравнивание по оси Y всегда по центру для лучшей читаемости,
- **“Color”:”0xAARRGGBB”,** - цвет текста в формате RGB888 или ARGB8888. Компонент AA задает прозрачность цвета к фону. Может быть задан от 0xFF (цвет непрозрачный) до 0x09 (цвет практически полностью прозрачен). Значение 0x00 (или отсутствие компоненты в формате RGB888) означает, что цвет будет непрозрачен к фону,
- **“bgColor”:”0xAARRGGBB”,** - цвет фона в формате RGB888 или ARGB8888. Компонент AA для фонового цвета будет означать степень прозрачности к черному цвету. Соответственно для фона имеет смысл задавать цвет в формате RGB888,
- **“ColorRandomTime_100ms”:3,** - (100мс) период смены цвета, для строк в режиме переливающейся заливки (когда цвет строки задан = 0x01000000),
- **“Scene”:”0”,** - номер сцены, загружаемый сразу после запуска. При отсутствии параметра, всеравно будет загружена сцена с номером 0. В режиме ModBus Master в качестве параметра можно задать номер как напрямую, так и через ModBus переменную. Например “Scene”:”&Vars.MB_PLC_D0”,
- **“LoadAreasAndScreensImmediate”:false,** - режим загрузки областей и экранов. Если параметр отсутствует или задан как false, то область/экран будут загружены только если задан номер отличный от текущего. Если true, то загрузка будет происходить безусловно при каждой команде (но при плохой связи с контроллером может наблюдаться “мерцание” экрана из-за многократного выполнения команды),

- **“ResetStringAfterSwitch”:false** - При значении true система будет сбрасывать прокрутку строки при каждом изменении номера строки в StringNumber. При отсутствии параметра или при значении false прокрутка строки не будет сбрасываться,

Пример:

```
"Defaults":
{
        "OEM_Brand_1":"Стр.1",
        “OEM_Brand_2":"Стр.2",

        "PauseBeforeScrollingString":500,
        "SpeedScrollingString":"1",
        "SpeedScrollingPerforateFrame":"2",
        "FontSize":"0x11",
        "Align":"0x22",
        “Color":"0xFF0000",
        “bgColor":"0x000A00",
        "ColorRandomTime_100ms":3,
        "Scene":0,
},
```

3.3.3. Блок “AreasCollection”

Блок применяется во всех режимах работы. Предназначен для описания коллекций областей.

Коллекция - это набор областей объединенных под одним именем “Collection_n”, где n - это номер коллекции от 0 до 32000. В последствии зная номер коллекции и номер экрана, можно создать из них сцену (пара коллекция-экран), или если область подходит для нескольких экранов, то можно не создавать несколько одинаковых коллекций, а загружать разные экраны в одну и туже коллекцию.

- *Коллекции могут иметь произвольные номера от 0 до 32000.*

Область - это место на экране, куда сможет выводиться строка, рамка и картинки. На экране область выглядит как прямоугольник, описываемый координатами. Например описание {"X1":0, "Y1":0, "X2":31, "Y2":7, }, означает, что область будет занимать 32 пикселя по оси X и 8 пикселей по оси Y, начиная с координаты (0;0).

- *Области внутри коллекции описываются координатами на экране. Номер области внутри коллекции начинается с 0 и далее по порядку. Всего для каждой коллекции доступно от 1 до N областей (Максимальное количество областей в коллекции зависит от типа контроллера).*
- *Нулевая точка всегда находится в левом верхнем углу экрана.*

В описание каждой области могут входить не только координаты, но и параметры вывода строки (“SpeedScrollingString”, “FontSize”, “Align”). Эти параметры будут применены при отсутствии соответствующих параметров в описаниях строк (подробнее см. блок “Screens”).

Пример:

```
"AreasCollection":
{
        "Collection_0":
        [
                {"X1":0, "Y1":0, "X2":31, "Y2":7, },        // Описание параметров области 0
                {"X1":0, "Y1":8, "X2":31, "Y2":15, },       // Описание параметров области 1 и т.д.
                ……..

                                                            // Максимальное кол-во областей в
                                                                    коллекции зависит от типа
                                                                    контроллера. Счет по  порядку.
        ],

        ……..

        "Collection_n":             // Коллекция с номером n.
        [
                {"X1":0, "Y1":0, "X2":31, "Y2":15, “SpeedScrollingString":"1", "FontSize":"0x11",
“Align":"0x22", },                                          // Описание параметров области 0
        ],
},
```

3.3.4. Блок “Screens”

Блок применяется во всех режимах работы. Предназначен для описания экранов.

Экран - это набор строк объединенных под одним именем “Screen_n”, где n - это номер коллекции от 0 до 32000. В последствии зная номер коллекции и номер экрана, можно создать из них сцену (пара коллекция - экран), или выбрать в какую коллекцию загрузить данный экран.

- *Экраны могут иметь произвольные номера от 0 до 32000.*

Строка - это блок с описанием параметров вывода и сама информация в виде текста и ссылок на картинки. В строку могут быть добавлены динамически изменяющиеся данные - аргументы строки, причем строка необязательно должна содержать текст, а может содержать только аргумент с числом и/или с картинкой. Более подробно в примерах в конце этого раздела.

- *Номер строки внутри экрана начинается с 0 и далее по порядку. Всего для каждого экрана доступно от 1 до N строк (Максимальное количество строк в экране зависит от типа контроллера).*
- *Строки соответствуют областям, и ограничения на количество соответствующие.*
- *Флаги строк по умолчанию имеют значение **false**.*
- *Для вывода символа процент “%”, в строке нужно сделать запись вида “%%”.*
- *Строки поддерживают только указанные ниже аргументы, при этом их формат похож на формат для функции printf из языка Си. **Попытка применить аргументы строки отличные от описанных ниже может привести к некорректной работе контроллера**. Все записи вида ”%буква латинского алфавита” будут рассматриваться как аргументы строки.*

Параметры, которые могут входить в каждый блок описания строки:

- **“String”:”текст выводимый в строку”,** - вывод обычной текстовой строки,
- **“StringNumber”:”0”,** - вывод строки с индексом 0 из блока “Strings”, В режиме ModBus Master в качестве аргумента можно задать номер как напрямую, так и через ModBus переменную. Например “StringNumber”:”&Vars.MB_PLC_D0”,
  - *При отсутствии строки с указанным индексом, в область будет выведено сообщение “Нет строки с индексом n”.*

- **“String”:”текст с аргументом %d”,** - вывод строки с аргументом. %d - это аргумент вместо которого будет подставлено число (Int32). Каждая строка может содержать до 5 аргументов с числами, причем считаются они начиная с 0 по порядку появления в строке. Аргументы могут следовать друг за другом, например “String”:”%d%d”,. Значения для аргументов описываются в этом-же блоке параметров и имеют вид “StringArg_0”:”123”, “StringArg_1”:”123”, и т.д. В режиме ModBus Master значение аргумента можно задать напрямую: “StringArg_0”:”123”, или через ModBus переменную: “StringArg_0”:”&Vars.MB_PLC_D0”,
- **“String”:”текст и картинка %%i0”,** - вывод строки с картинкой. %%i0 - это аргумент вместо которого будет подставлена картинка из блока “Images”. Каждая строка может содержать несколько картинок, максимально доступное количество зависит от модификации контроллера и от размеров картинок. Чем больше точек занимает картинка, тем больше места она займет в памяти контроллера и тем меньше картинок одновременно можно вывести на экран. Несколько картинок могут быть выведены друг за другом, разделять аргументы пробелами необязательно, например “String”:“%%i0%%i1”, но следует учитывать, что после аргумента с номером картинки не может сразу идти аргумент с цифрой, их нужно разделить пробелом или текстом. Если одна и та-же картинка вызывается в строке несколько раз, то память будет занимать только один экземпляр. В режиме ModBus Master также можно динамически менять картинки, для этого следует задать номер строки с ссылкой на ModBus переменную (например “StringNumber”:”&Vars.MB_PLC_D0”), а в блоке “Strings” создать строки с аргументами для нужных картинок (например “Strings”: [“%%i0”, “%%i2”,]). Теперь меняя значение переменной Vars.MB_PLC_D0 можно переключать выводимые картинки,
  - *Аргументы с картинками не относятся к аргументам для чисел и не уменьшают их количество.*

- **“SpeedScrollingString”:”1”,** - скорость прокрутки длинных строк. Рабочие значения от 10 до 1 и от -1 до масштаба заданного для оси X в FontSize, по 20мс. Это значит, что если задано значение 10, то строка будет смещаться на 1 пиксель каждые 10 * 20 = 200мс, и текст будет двигаться медленно. Если задано значение 1, то строка будет смещаться на 1 пиксель каждые 1 * 20 = 20мс, и соответственно текст будет двигаться быстро. Значения со знаком “-“ дают еще большую скорость и ограниченны только выбранным масштабом текста по оси X. Если параметр не задан или равен 0, то будет взято значение из соответствующего параметра из области, в которую будет загружаться строка, либо при отсутствии значение по умолчанию из блока “Defaults”,
- **“FontSize”:”0x11”,** - размер шрифта по осям X/Y. Имеет вид 0xXY. Где X это масштаб по оси X, Y это масштаб по оси Y. Стандартный символ имеет размер 6

пикселей в ширину (ось X) и 8 пикселей в высоту (ось Y). Если задан масштаб 0x11, то символу будет обычного размера 6*8. Если масштаб: 0x21, то размер 12*8, если 0x12, то размер 6*16, если 0x33, то размер 18*24, и так далее. По каждой оси множитель может быть от 1 до F. Если масштаб задан одной цифрой например 0x08, то он будет принят как масштаб 0x88. Запись вида: 0xN0, где число N задано, а после стоит 0 считается недопустимой. Если параметр не задан или равен 0, то будет взято значение из соответствующего параметра из области, в которую будет загружаться строка, либо при отсутствии значение по умолчанию из блока “Defaults”,

- **“Align”:”0x22”,** - выравнивание текста по осям X/Y. Имеет вид 0xXY. Где X может принимать значения: 1-выравнивание по левому краю, 2-по центру, 3-по правому краю. Y может принимать значения: 1-по верхнему краю, 2-по центру, 3-по нижнему краю. Если параметр не задан или равен 0, то будет взято значение из соответствующего параметра из области, в которую будет загружаться строка, либо при отсутствии значение по умолчанию из блока “Defaults”. Для картинок и текста не помещающихся по высоте в заданную область всегда применяется выравнивание по оси Y по середине.,
- **“Color”:”0xAARRGGBB”,** - цвет текста в формате RGB888 или ARGB8888. Компонент AA задает прозрачность цвета к фону. Может быть задан от 0xFF (цвет непрозрачный) до 0x09 (цвет практически полностью прозрачен). Значение 0x00 (или отсутствие компоненты в формате RGB888) означает, что цвет будет непрозрачен к фону. Если параметр не задан, то будет взято значение по умолчанию из блока “Defaults”,
- **“bgColor”:”0xAARRGGBB”,** - цвет фона в формате RGB888 или ARGB8888. Компонент AA для фонового цвета будет означать степень прозрачности к черному цвету. Соответственно для фона имеет смысл задавать цвет в формате RGB888. Если параметр не задан или равен 0x00000000, то будет взято значение по умолчанию из блока “Defaults”,
- **“Blink”:true,** - флаг мигания строки. true - мигать, false - показывать не мигая,
- **“BlinkInverseColor”:true,** - флаг при котором строка будет циклично менять свой цвет с прямого на обратный. true - мигать, false - показывать заданным цветом,
- **“ShowOnlyTimes”:true,** - флаг, показать строку только 1 раз. Например если строка короткая, то она будет показана в течение времени заданном в блоке “Defaults”.”PauseBeforeScrollingString” и будет стерта, либо если строка длинная, то она будет выводиться, пока не скроется последний символ,
- **“Frame”:true,** - флаг, вывести рамку в текущую область. Рамка может быть выведена только в одну из областей, другие варианты не предусмотрены. Ширина рамки 1 пиксель, все флаги касающиеся вывода рамки будут работать только если рамка отображается. Если в загружаемом экране не в одной из строк рамка не используется, то рамка выведенная ранее будет удалена,
- **“FramePerforateLength”:11,** - длина штриха для рамки, если значение отлично от 0, то рамка будет “перфорированная” и по умолчанию начнет вращаться по часовой стрелке. Если задано значение 0, или параметр отсутствует, то рамка будет сплошной,
- **“FrameConterClockWise”:true,** - флаг направления вращения перфорированной рамки. true - против часовой стрелки, false - по часовой стрелке,
- **“FrameStop”:true,** - флаг, остановить вращение перфорированной рамки,

- **“FrameColor”:”0xAARRGGBB”,** - цвет рамки в формате RGB888 или ARGB8888. Режимы и ограничения аналогичны цвету для текста. Рамка может переливаться или быть одного цвета,
- **“SpeedScrollingPerforateFrame”:”1”,** - скорость прокрутки перфорированной рамки,

Пример:

```
"Screens":
{
        "Screen_0":
        [
                {“String”:"Строка 1", "Color":"0x01000000", “bgColor”:”0x000A00”, },
                {“String”:"Строка 2", "Color":"0xFF0000", },
                {"StringNumber":"0", "Color":"0xFFFFFF", },
                {“String”:"%d", "Color":"0xFFFFFF", “StringArg_0”:"123", },
                {“String”:"%%i0", },
                ……..
        ],

        ……..

        "Screen_n":             // Экран с номером n
        [
                {"StringNumber":"&Vars.MB_PLC_D0", "Color":"0x01000000", },
                {“String”:"%d %%i0", "Color":"0x0080FF", “StringArg_0”:"&Vars.MB_PLC_D0", },

                {“String":"Строка 3", "Color":"0xFFFFFF", "FontSize":"0x22", "Frame":true,
“FrameColor":"0x0020FF", },
                {“String":"Строка 4", "Color":"0xFF0000", “bgColor”:”0x0A0000”,
”FontSize”:"0x31", “Align":"0x22", “BlinkInverseColor":true, },
        ],
},
```

3.3.5. Блок “Scenes”

Блок применяется во всех режимах работы. Предназначен для описания сцен.

Сцена - это пара “коллекция - экран” объединенных под одним именем “Scene_n”, где n - это номер коллекции от 0 до 32000. Предназначены для более удобной работы, а также для демонстрационных режимов или для рекламы. Есть возможность автоматического переключения между сценами, без внешних команд или дополнительных устройств.

Сцены могут иметь произвольные номера от 0 до 32000.

Параметры для каждой сцены:

- **“AreaCollection”:0,** - номер коллекции,
- **“Screen”:0,** - номер экрана,
- **“TimeToJump_100ms”:50,** - (необязательный параметр) (100мс) время после загрузки текущей сцены для перехода к следующей. От 1 до 32000, если параметр задан = 0 или отсутствует, то будет задано время по умолчанию = 1,
- **“JumpToScene”:1,** - (необязательный параметр) номер сцены, на которую будет совершен переход по истечении времени в “TimeToJump_100ms”. Если параметр = 0 или отсутствует, то смены сцены не будет, и текущая сцена будет активной до тех пор, пока не будет получена команда на загрузку другой сцены.

Пример:

```
"Scenes":
{
        "Scene_0": {"AreaCollection":0, "Screen":0, "TimeToJump_100ms":50,
"JumpToScene":1, },
        "Scene_1": {"AreaCollection":0, "Screen":1, "TimeToJump_100ms":50,
"JumpToScene":0, },
},
```

3.3.6. Блок “Strings”

Необязательный блок, применяется по желанию пользователя. Представляет собой массив строк, которые можно выводить динамически в режиме ModBus Master или за ранее указать номера строк для вывода при описании экранов. (подробнее см. блок “Screens”). Количество строк в массиве может быть до 32000. Счет строк идет от 0 по порядку:

Пример:

```
"Strings":
[
        "АВТО+",
        "Мы рады приветствовать вас!",
        "Режим",
        "ВОДА ШАМПУНЬ ВОСК ОСМОС",
        "ПАУЗА",
        "ОСМОС",

        "%d",
        “%%i0”,
],
```

3.3.7. Блок “Images”

Необязательный блок, применяется по желанию пользователя и только если есть необходимость в использовании картинок. Каждая картинка описывается в блоке вида “Image_n”, где n - это номер блока для картинки от 0 до 32000. **Поддерживаются картинки в форматах *.png и *.gif**. Размер картинки лучше всего за ранее подогнать под необходимое разрешение в любом фото-редакторе. Например если область имеет размеры 32*16 px, то следует подготовить картинку размером соответствующим области. Если такой возможности нет, то можно взять картинку размером 32*32 px или 48*48 px и задать коэффициенты масштаба в параметрах в блоке для картинки. Первоначальная подготовка картинок перед применением позволяет получить наилучшее качество, а также дает возможность одновременно вывести бОльшее количество картинок.

Каждая картинка перед выводом на экран масштабируется (если заданы параметры масштабирования) и вычисляется размер фрейма (область экрана) под картинку. Далее картинка смещается внутри фрейма (если заданы параметры смещения) и только после этого выводится в нужное место в строке. Если размер фрейма картинки больше области, в которую она выводится, то картинка будет обрезана (масштабирование и смещение неприменимо к картинкам в формате *.gif).

- *Блоки для картинок могут иметь произвольные номера от 0 до 32000.*
- *Поддерживаются картинки в форматах *.png и *.gif*
- *Параметры смещения и масштабирования **не будут** применены, если картинка имеет формат gif.*
- *Файлы могут находиться как в корне диска, так и в папках.*

Картинки могут находиться в папках, для этого в параметре “File” указывается полный путь до каждой картинки. Если все картинки находятся в одной папке, то путь до нее можно указать в параметре “ImgFolder”:”img/“, а в описаниях самих картинок в параметре “File” оставить только имя. Параметр “ImgFolder”:”img/“ для удобства размещается перед блоком “Images”.

Параметры для каждой картинки:

- **“File”:”car_black.png”,** - название файла с картинкой. Длина не более 50 символов.
- **“Offset_X”:”0”,** - (необязательный параметр) смещение картинки внутри фрейма по оси X. Значение может быть как положительным, так и отрицательным,
- **“Offset_Y”:”0”,** - (необязательный параметр) смещение картинки внутри фрейма по оси Y. Значение может быть как положительным, так и отрицательным,
- **“ScalePercent_X”:”100”,** - (необязательный параметр) масштаб картинки внутри фрейма по оси X. Значение может быть от 100 до 0,
- **“ScalePercent_Y”:”100”,** - (необязательный параметр) масштаб картинки внутри фрейма по оси Y. Значение может быть от 100 до 0.
- **“TransparentBackground”:true,** - (необязательный параметр) сделать фон картинки прозрачным к фону области. Этот параметр актуален для формата *.gif, где существует понятие “Фон”, но нет прозрачности (как в фоормате *.png)

Пример:

```
"ImgFolder":"img/",
"Images":
{
        "Image_0": {"File":"car_black.png", "Offset_X":"0", "Offset_Y":"0", "ScalePercent_X":"50",
"ScalePercent_Y":"50", },
        "Image_1": {“File":"car_white.png", },
        "Image_2": {“File":"screw_orange.png", },
},
```

3.3.8. Блок “Vars”

Необязательный блок, применяется только для режима ModBus Master. Предназначен для описания всех переменных, которые будут опрашиваться у подчиненных устройств по протоколу ModBus. Каждая переменная описывается в блоке вида “MB_x”, где x - это текст, который может содержать только буквы латинского алфавита, цифры и знак нижнего подчеркивания “_”.

Опрос переменной происходит, только если в этом есть текущая необходимость. Например загружен экран с двумя строками, в первой строке в качества номера строки используется переменная ModBus, а во второй строке есть 2 аргумента, значения которых заданы разными переменными ModBus, тогда опрашиваться будут только 3 переменных.

- *В названии переменной можно использовать только буквы латинского алфавита, цифры и символ нижнего подчеркивания “_”.*
- ***Регистр букв важен** и должен быть одинаков как при объявлении переменной, так и при ее вызове.*
- *Максимальное количество одновременно опрашиваемых переменных зависит от типа контроллера (см. технические характеристики).*
- *Доступна работа с адресами в форматах “4x 40001 и т.д.” и “3x 30001 и т.д.”. В этом случае параметр **“MbFunc”** не учитывается и его не обязательно задавать. При такой записи регистров система сама выберет верный вариант опроса.*

Параметры для каждой переменной:

- **“MbAddr”:”1”,** - адрес подчиненного устройства,
- **“MbFunc”:”0x03”,** - функция ModBus для чтения данных из устройства, поддерживаются команды 0x03 и 0x04,
- **“RegAddr”:”0x1000”,** - адрес регистра,
- **“RegCount”:”1”,** - количество регистров. Обычно 1 или 2 регистра,
- **“PollTime_10ms”:”50”,** - (10 мс) период опроса переменной,
- **“ValueBitMask”:”0x0802”,** - (необязательный параметр) маска, указывающая какие биты будут участвовать в результате чтения переменной. Имеет длину до 32 бит (0xFFFFFFFF).
  - *Еесли параметр отсутствует, то к результату применяется маска 0xFFFFFFFF.*
  - *Параметр работает только если запрошено не более 2-х регистров,*

- **“ValueTypeBool”:true,** - (необязательный параметр) флаг, если true, то значение переменной будет рассматриваться только как булево. То есть, если переменная имеет любое значение кроме 0, то полученное значение будет округлено до 1, если 0, то 0,
- **“ValueOffset”:”0”,** - (необязательный параметр) Смещение для прочитанной переменной. Тип: “целое беззнаковое 32 бита" (int32). По умолчанию и при отсутствии параметра = 0,
- **“ValueMult”:”1.2”,** - (необязательный параметр) Множитель для прочитанной переменной. Может быть задано как целым числом ("1"), так и числом с плавающей точкой ("1.2") (float). По умолчанию и при отсутствии параметра = 1.

Пример:

```
"Vars":
{
        "MB_PLC_D0":
        {
                "MbAddr":"1",
                "MbFunc":"0x03",
                "RegAddr":"0x1000",
                "RegCount":"1",
                "PollTime_10ms":"50",
        },

        "MB_PLC_D1":
        {
```

```
                "MbAddr":"1",
                "MbFunc":"0x03",
                "RegAddr":"0x1002",
                "RegCount":"2",
                "PollTime_10ms":"50",
        },
},
```

3.3.9. Блок “Bridge_n”

Необязательный блок, применяется по желанию пользователя и только для режима ModBus Master. Предназначен для описания мостов для переброски групп переменных от одного подчиненного устройства к другому.

Мост - это задание на копирование группы регистров из одного подчиненного устройства в другое, объединенных под одним именем “Bridge_n”, где n - это номер моста от 0 до 32000. Мосты должны быть расположены друг за другом, номера мостов должны начинаться с 0.

- *Работа мостов начинается после запуска контроллера и продолжается все время.*
- *Период опроса каждого моста равен 100 мс.*
- *Каждый мост может перебрасывать от 1 до 26 регистров за раз.*
- *Каждый мост занимает 2 переменных ModBus, соответственно, если сейчас нет опроса переменных и для контроллера доступен опрос 33-х переменных, то можно создать до 16 мостов. Если задано допустим 5 мостов, то контроллер сможет дополнительно опрашивать не более 33 - 5 * 2 = 33 - 10 = 23 переменных ModBus.*
- *Доступна работа с адресами в форматах “4x 40001 и т.д.” и “3x 30001 и т.д.”. В этом случае система сама выберет верный вариант опроса при такой записи регистров.*
- *Чтение регистров происходит командой “0x03”, но если адрес регистра имеет вид “3x 30001 и т.д.”, то команда чтения будет заменена на “0x04”.*
- *Запись одного регистра происходит командой “0x06”, запись группы регистров командой “0x10”.*

Параметры для каждой переменной:

- **“From_MbAddr”:”10”,** - адрес подчиненного устройства, откуда будем читать данные,
- **“From_RegAddr”:”0x00”,** - адрес первого регистра, с которого начнется чтение,
- **“To_MbAddr”:”1”,** - адрес подчиненного устройства, куда будем писать данные,
- **“To_RegAddr”:”0x1000”,** - адрес первого регистра, с которого будет идти запись,
- **“RegCount”:”10”,** - количество регистров

Пример:

```
"Bridge_0":
{
        "From_MbAddr":"10", "From_RegAddr":"0x00",
        "To_MbAddr":"1", "To_RegAddr":"0x1000",
        "RegCount":"10",
},
```

3.3.10.Пример файла для демонстрации или рекламы

```
{
	"ScreenSettings":
	{
		"PanelsOn_X":1,
		"PanelsOn_Y":1,
		"Gamma":128,
		"BitsPerColor":5,
	},

	"Defaults":
	{
		"OEM_Brand_1":"Стр.1",
		“OEM_Brand_2":"Стр.2",

		"PauseBeforeScrollingString":500,
		"SpeedScrollingString":"1",
		"SpeedScrollingPerforateFrame":"2",
		"FontSize":"0x11",
		"Align":"0x22",
		“Color":"0xFF0000",
		"bgColor":"0x000A00",
		"ColorRandomTime_100ms":3,

		"Scene":"0",
	},

	"AreasCollection":
	{
		"Collection_0":
		[
			{"X1":0, "Y1":0, "X2":31, "Y2":15, "FontSize":"0x12", "Align":"0x22", },
		],

		"Collection_1":
		[
			{"X1":0, "Y1":0, "X2":31, "Y2":7, },
			{"X1":0, "Y1":8, "X2":31, "Y2":15, },
		],
	},

	"Screens":
	{
		"Screen_0":
		[
			{"String":"Добро", "Color":"0xFF8000", "BlinkInverseColor":true, },
			{"String":"пожаловать", "Color":"0x0080FF", "BlinkInverseColor":true, },
		],

		"Screen_1":
		[
			{"StringNumber":0, "Color":"0x99CCFF", },
			{"StringNumber":1, "Color":"0x01FFAF00", },
		],
	},
```

```
        "Scenes":
        {
                "Scene_0": {"AreaCollection":1, "Screen":0, "TimeToJump_100ms":60,
"JumpToScene":1, },
                "Scene_1": {"AreaCollection":1, "Screen":1, "TimeToJump_100ms":50,
"JumpToScene":0, },
        },

        "Strings":
        [
                "МЕГА",
                "МОЙКА",
        ],
},
```

4. Регистры ModBus

Устройство поддерживает команды ModBus: 0x02, 0x03, 0x04, 0x06 и 0x10. Максимальное количество регистров принимаемое/передаваемое за одну команду: 26.

Значения некоторых регистров загружаются из файлов настроек при включении устройства. Имя файла и название блока настроек будет указано в описании каждого регистра. Если для регистра разрешена запись, то его значение может быть изменено через ModBus, но изменения будут сброшены к настройкам из файлов при следующем перезапуске.

Обозначения регистров:

- R - только для чтения,
- W - для записи,
- R/W - для чтения и записи,
- E - регистр хранится в энергонезависимой памяти,
- N - регистры зарезервированы, запись в них запрещена.

Стандарт ModBus предусматривает передачу данных в регистрах. Каждый регистр имеет длину 16 бит (2 байта). Порядок передачи байт от старшего к младшему (Big-endian). То есть, если регистр содержит значение 0xA1B2, то сначала будет передан байт 0xA1, а затем байт 0xB2.

Следует помнить, что некоторые значения в памяти устройства могут иметь длину 32 бита (4 байта), что соответствует 2-ум регистрам. В стандарте не определено в каком именно порядке должны передаваться регистры (старший-младший или младший-старший), это зависит от конкретного устройства. **Все устройства компании CYBERiDEA хранят данные в порядке от младшего к старшему (Little-endian)**. Это значит, что если будет обращение к переменной со значением 0xA1B2C3D4, то сначала будет передан младший регистр (младшие 16 бит числа) со значением 0xC3D4, а затем старший регистр со значением 0xA1B2. По байтам порядок будет такой: 0xC3, 0xD4, 0xA1, 0xB2. Число 1234 (dec) = 0x0000 04D2 (hex) можно передать командой 0A 10 00 00 00 02 04 **04 D2 00 00**.

4.1. Формат данных

Если первым символом является буква U, значит переменная может иметь только положительные значения (без знака):

- Uint8 - беззнаковый 8 бит, значения от 0 до 255 (0xLL),
- Int8 - знаковый 8 бит, значения от -128 до 127 (0xLL),
- Uint16 - беззнаковый 16 бит, значения от 0 до 65535 (0xHHLL),
- Int16 - знаковый 16 бит, значения от -32768 до 32767 (0xHHLL),
- Uint32 - беззнаковый 32 бит, значения от 0 до 4294967295 (0xHHHL LHLL),
- Int32 - знаковый 32 бит, значения от --2147483648 до -2147483647 (0xHHHL LHLL),
- Uint64 - беззнаковый 64 бит, значения от 0 до (2^64 - 1) (0x HHHH HHLL LLHH LLLL),
- Int64 - знаковый 64 бит, значения от -(2^63) до (2^63 - 1) (0x HHHH HHLL LLHH LLLL).

4.2. Командный регистр

Это набор регистров в памяти устройства доступные для чтения и для записи, диапазон от 0x0100 до 0x0119, всего 26 регистров. Номер команды записывается по адресу 0x0100, статус выполнения команды в регистр 0x0101, остальные регистры должны быть записаны в соответствии с выбранной командой. Такое применение адресного пространства позволяет из одного места получать доступ ко всем функциям контроллера экрана. Например можно задать размер шрифта в определенной области экрана, при этом через одну команду можно обратиться к любой доступной области меняя всего несколько параметров. Все регистры из массива (0x0100 - 0x0119) доступны как для записи, так и для чтения, с некоторыми ограничениями. После того как команда запущена на выполнение запись в любой из регистров никак не может повлиять на ее выполнение (изменить номер команды или статус невозможно) до тех пор, пока она не завершит свою работу. Чтобы понять на какой стадии выполнения находится команда (0x0101) и узнать номер команды (0x0100), можно свободно читать соответствующие регистры. Как только команда завершит работу, регистр статуса (0x0101) изменит свое состояние на один из вариантов: “команда выполнена успешно" либо “ошибка”. В случае успешного выполнения регистры 0x0102 - 0x0119 буду содержать результат (если команда предполагает возврат результата своей работы).

- *Программист может представить командный регистр как функцию, в которую в качестве аргумента передается буфер с параметрами (регистры 0x0102 - 0x0119). После запуска, функция выполняется и возвращает результат обратно в этот же буфер.*
- *Чтение результата выполнения командны не обязательно, но следует помнить, что на выполнение команды нужно время (в зависимости от команды от 20 до 500 мс) и если передавать их слишком часто, то некоторые будут пропущены пока контроллер занят выполнением предыдущей команды.*

Описание регистров выглядит следующим образом:

0x0100	**0xNN**	**Название функции**	**R W**	**Тип переменной**
				Кол-во регистров

где:

0xNN - номер команды,

R/W - тип доступа на чтение и запись,

Тип переменной - тип значения переменной,

Кол-во регистров - количество регистров задействованных в команде. Если указан диапазон, значит есть минимально необходимое количество регистров для команды, а остальные регистры могут быть использованы по необходимости.

Назначение регистров:

0x0100 - номер выполняемой команды.

0x0101 - статус выполнения команды:

-1 Ошибка выполнения команды,

0 Командный регистр еще не запускался,

1 Запустить выполнение команды,

2 Команда выполняется,

3 Команда выполнена успешно. Результат готов для чтения.

0x0102-0x0119 - регистры с параметрами для команды. Назначение регистров и их количество зависит от того какая команда будет выполняться. Обращение к регистрам может быть произведено в произвольном порядке. Например нужно вывести текст в область на экране, для этого запускаем команду 0.

Можно сразу произвести запись всех регистров одним пакетом:

0x0100 (номер команды) = 0x00 (вывести текст в область),
0x0101 (статус выполнения) = 1 (запустить команду на выполнение),
0x0102 (индекс области) = 1 (индекс 1 означает обращение к области 2)
0x0103 - 0x0104 (цвет в формате RGB888) = 0xFF0000 (красный цвет)
0x0105 (размер шрифта по осям X/Y) = 0x21 (по оси X = 2, по оси Y = 1)
0x0106 (выравнивание текста по осям X/Y) = 0x21 (по оси X = по центру, по оси Y = сверху)
0x0107 (скорость прокрутки текста) = 0 (взять из файла “User*.txt”)
0x0108 (флаги отображения текста) = 0 (не менять)
0x0109 (первый символ текста или признак индекса) = 0x1B (в следующем регистре находится индекс строки в файле)
0x010A (индекс строки с текстом) = 1 (вывести строку с индексом 1 раздел «**Strings**» файл «Uset_*.txt»

Можно произвести запись в несколько произвольных этапов, например:

1) записать регистры с параметрами команды:

0x0102 (индекс области) = 1 (индекс 1 означает обращение к области 2)
0x0103 - 0x0104 (цвет в формате RGB888) = 0xFF0000 (красный цвет)
0x0105 (размер шрифта по осям X/Y) = 0x21 (по оси X = 2, по оси Y = 1)
0x0106 (выравнивание текста по осям X/Y) = 0x21 (по оси X = по центру, по оси Y = сверху)
0x0107 (скорость прокрутки текста) = 0 (взять из файла “User*.txt”)
0x0108 (флаги отображения текста) = 0 (не менять)
0x0109 (первый символ текста или признак индекса) = 0x1B (в следующем регистре находится индекс строки в файле)
0x010A (индекс строки с текстом) = 1 (вывести строку с индексом 1 раздел «**Strings**» файл «Uset_*.txt»

2) записать номер команды:

0x0100 (номер команды) = 0x00 (вывести текст в область),

3) запустить команду на выполнение:

0x0101 (статус выполнения) = 1 (запустить команду на выполнение),

Основное условие, чтобы:

-либо все регистры были записаны с адреса 0x0100 одной операцией,

-либо подготовить все регистры, а затем запустить команду на выполнение, записав «1» в регистр 0x0101.

Структура командного регистра:

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Регистр с параметрами 0,
Регистр 0x0103: Регистр с параметрами 1,
……
Регистр 0x0119: Регистр с параметрами 23.

4.3. Описание регистров

0x2C	Номер текущей коллекции областей	R	Uint16
			1

Регистр содержит номер текущей загруженной коллекции областей.

Регистр 0: Номер текущей коллекции областей.

0x2D	Номер текущего экрана	R	Uint16
			1

Регистр содержит номер текущего загруженного экрана.

Регистр 0: Номер текущего экрана.

0x0100	0x00	Вывести строку в область экрана	R W	Struct. 10 - 26

Вывести текст в заданную область экрана. При этом сразу задаются все доступные параметры отображения строки. В каждом регистре находится 1 буква. Символом завершения текстовой строки всегда служит символ с кодом 0x00.

- *Чтобы использовать строку из файла (файл User_*.txt, раздел **«Strings»**), необходимо в **регистр 0x0109** записать значение 0x1B, а в **регистр 0x010A** записать индекс строки (начиная с 0). Если указан индекс строки которой нет, будет выведен текст «Нет строки с индексом'n'».*
- *Аргументы строки могут быть переданы начиная с регистра 0x010B. Имеют тип Int32 и занимают по 2 регистра.*

В данной команде аргументы доступны только для строк вызываемых через индекс строки. Например необходимо в текст добавить меняющуюся числовую информацию. В строке «Температура воды %d'C» можно менять значение температуры, при этом символы "%d" будут заменены на значение аргумента. Значения для аргументов можно передавать в регистрах сразу после регистра с индексом строки. Максимальное количество аргументов для каждой строки зависит от модели контроллера и описано в сводной таблице характеристик.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103 - 0x0104: Цвет в формате ARGB8888 или RGB888. Компонента A задает степень прозрачности перед цветом фона. Может быть задана от 0xFF до 0x00, причем если значение A меньше 0x09, то компонента будет принята равной 0xFF,
Регистр 0x0105: Размер шрифта по осям X/Y. Если задан 0, то настройка будет взята из файла "User*.txt". Имеет вид 0xXY. Где X это масштаб по оси X, Y это масштаб по оси Y. Стандартный символ имеет размер 6 пикселей в ширину (ось X) и 8 пикселей в высоту (ось Y). Если задан масштаб 0x11, то символу будет обычного размера 6*8. Если масштаб: 0x21, то размер 12*8, если 0x12, то размер 6*16, если 0x33, то размер 18*24, и так далее. По каждой оси множитель может быть от 1 до F. Если масштаб задан одной цифрой например 0x08, то он будет принят как масштаб 0x88. Запись вида: 0xN0, где число N задано, а после стоит 0 считается недопустимой. Если передано значение 0, то будет применен размер заданный в файле "User*.txt",
Регистр 0x0106: Выравнивание текста по осям X/Y. Если задан 0, то настройка будет взята из файла "User*.txt". Имеет вид 0xXY. Где X может принимать значения: 1-выравнивание по левому краю, 2-по центру, 3-по правому краю. Y может принимать значения: 1-по верхнему краю, 2-по центру, 3-по нижнему краю,
Регистр 0x0107: Скорость прокрутки текста. Если задан 0, то настройка будет взята из файла "User*.txt". Рабочие значения от 10 до 1 и от -1 до масштаба заданного для оси X в FontSize (0x0105), по 20мс. Это значит, что если задано значение 10, то строка будет смещаться на 1 пиксель каждые 10 * 20 = 200мс, и текст будет двигаться медленно. Если задано значение 1, то строка будет смещаться

на 1 пиксель каждые 1 * 20 = 20мс, и соответственно текст будет двигаться быстро. Значения со знаком “-“ дают еще большую скорость и ограниченны только выбранным масштабом текста по оси X,

Регистр 0x0108: Флаги отображения текста. Если задан 0, то настройка останется без изменений. Имеет вид: 0b 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00, где каждый бит это флаг (0 или 1):

- 00 - (U)показать строку ,
- 01 - (U)мигать (приоритет выше чем у мигания инвертируя цвет),
- 02 - (U)мигать инвертируя цвет строки,
- 03 - резерв,

  …
- 09 - строка только что принята (еще не выводилась на экран),
- 10 - (U)вывести строку только 1 раз (после полной прокрутки она исчезнет),
- 11 - строка закончила свой вывод (после полной прокрутки флаг поднимется автоматически).



Флаги обозначенные символом (U) доступны для чтения/записи пользователем, остальные только для чтения. Если нужно установить флаги, то в регистре обязательно устанавливается флаг 00 и любые флаги, доступные для записи.

Регистр 0x0109: ASCII символ первой буквы (либо 0x1B для вызова строки по индексу),

Регистр 0x010A: ASCII символ второй буквы (либо индекс строки из блока “Strings”),

Регистр 0x010B: ASCII символ третьей буквы (или может содержать младшую часть аргумента 0),

Регистр 0x010C: ASCII символ четвертой буквы (или может содержать старшую часть аргумента 0),

Регистр 0x010D: ASCII символ пятой буквы (или может содержать младшую часть аргумента 1),

Регистр 0x010E: ASCII символ шестой буквы (или может содержать старшую часть аргумента 1),

……

Регистр 0x0119: см. описание регистра 0x0109.

0x0100	0x01	**Короткий вывод строки в область экрана**	**R W**	**Struct.** **4 - 26**

Короткий аналог команды “0x00”. Вывести текст в заданную область экрана. Задается только область для вывода, остальные параметры отображения строки остаются неизменными. В каждом регистре находится 1 буква. Символом завершения текстовой строки всегда служит символ с кодом 0x00.

- *Чтобы использовать строку из файла (файл User_*.txt, раздел **«Strings»**), необходимо в **регистр 0x0103** записать значение 0x1B, а в **регистр 0x0104** записать индекс строки (начиная с 0). Если указан индекс строки которой нет, будет выведен текст «Нет строки с индексом'n'».*
- *Аргументы строки могут быть переданы начиная с регистра 0x0105. Имеют тип Int32 и занимают по 2 регистра.*

В данной команде аргументы доступны только для строк вызываемых через индекс строки. Например необходимо в текст добавить меняющуюся числовую информацию. В строке «Температура воды %d'C» можно менять значение температуры, при этом символы "%d" будут заменены на значение переменной. Значения для аргументов можно передавать в регистрах сразу после регистра с индексом строки. Максимальное количество аргументов для каждой строки зависит от модели контроллера и описано в сводной таблице характеристик.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103: ASCII символ первой буквы (либо 0x1B для вызова строки по индексу),
Регистр 0x0104: ASCII символ второй буквы (либо индекс строки из блока "Strings"),
Регистр 0x0105: ASCII символ третьей буквы (или может содержать младшую часть аргумента 0),
Регистр 0x0106: ASCII символ четвертой буквы (или может содержать старшую часть аргумента 0),
Регистр 0x0107: ASCII символ пятой буквы (или может содержать младшую часть аргумента 1),
Регистр 0x0108: ASCII символ шестой буквы (или может содержать старшую часть аргумента 1),
……
Регистр 0x0119: см. описание регистра 0x0103.

0x0100	0x02	Вывести число с разделителем в область экрана	R W	Struct. 12

Вывести число с разделителем в заданную область экрана. При этом сразу задаются все доступные параметры отображения строки. Разделителем является точка ".". Позиция разделителя указывается в символах справа налево. Правый символ числа считается нулевым. Например в число "123456789" добавить разделитель в позицию 1, мы получим строку "12345678.9". Если позиция разделителя (**0x010B**) выходит за рамки имеющегося числа, то слева добавляются нули. Например если в число "1234" добавить разделитель в позицию 5, мы получим строку "0.01234". Позиция разделителя может быть задана в диапазоне от 0 до 9. Максимальная длина строки вместе с разделителем (при наличии) не более 11 символов.

Регистр 0x0100: Номер команды,

Регистр 0x0101: Статус выполнения,

Регистр 0x0102: Индекс области на экране,

Регистр 0x0103 - 0x0104: Цвет в формате ARGB8888 или RGB888. Компонента A задает степень прозрачности перед цветом фона. Может быть задана от 0xFF до 0x00, причем если значение A меньше 0x09, то компонента будет принята равной 0xFF,

Регистр 0x0105: Размер шрифта по осям X/Y. Если задан 0, то настройка будет взята из файла “User*.txt”. Имеет вид 0xXY. Где X это масштаб по оси X, Y это масштаб по оси Y. Стандартный символ имеет размер 6 пикселей в ширину (ось X) и 8 пикселей в высоту (ось Y). Если задан масштаб 0x11, то символу будет обычного размера 6*8. Если масштаб: 0x21, то размер 12*8, если 0x12, то размер 6*16, если 0x33, то размер 18*24, и так далее. По каждой оси множитель может быть от 1 до F. Если масштаб задан одной цифрой например 0x08, то он будет принят как масштаб 0x88. Запись вида: 0xN0, где число N задано, а после стоит 0 считается недопустимой,

Регистр 0x0106: Выравнивание текста по осям X/Y. Если задан 0, то настройка будет взята из файла “User*.txt”. Имеет вид 0xXY. Где X может принимать значения: 1-выравнивание по левому краю, 2-по центру, 3-по правому краю. Y может принимать значения: 1-по верхнему краю, 2-по центру, 3-по нижнему краю,

Регистр 0x0107: Скорость прокрутки текста. Если задан 0, то настройка будет взята из файла “User*.txt”. Рабочие значения от 10 до 1 и от -1 до масштаба заданного для оси X в FontSize (0x0105), по 20мс. Это значит, что если задано значение 10, то строка будет смещаться на 1 пиксель каждые 10 * 20 = 200мс, и текст будет двигаться медленно. Если задано значение 1, то строка будет смещаться на 1 пиксель каждые 1 * 20 = 20мс, и соответственно текст будет двигаться быстро. Значения со знаком “-“ дают еще большую скорость и ограниченны только выбранным масштабом текста по оси X,

Регистр 0x0108: Флаги отображения текста. Если задан 0, то настройка останется без изменений. Имеет вид: 0b 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00, где каждый бит это флаг (0 или 1):

- 00 - (U)показать строку ,
- 01 - (U)мигать (приоритет выше чем у мигания инвертируя цвет),
- 02 - (U)мигать инвертируя цвет строки,
- 03 - резерв,

  …
- 09 - строка только что принята (еще не выводилась на экран),
- 10 - (U)вывести строку только 1 раз (после полной прокрутки она исчезнет),
- 11 - строка закончила свой вывод (после полной прокрутки флаг поднимется автоматически).



Флаги обозначенные символом (U) доступны для чтения/записи пользователем, остальные только для чтения. Если нужно установить флаги, то в регистре обязательно устанавливается флаг 00 и любые флаги, доступные для записи.

Регистр 0x0109 - 0x010A: (Int32) Число для вывода на экран,
Регистр 0x010B: Позиция символа-разделятеля справа налево (от 0 до 9),
……
Регистр 0x0119: см. описание регистра 0x0109.

0x0100	0x03	**Короткий вывод числа с разделителем в область экрана**	**R W**	**Struct.** **6**

Короткий аналог команды “0x02”. Вывести число с разделителем в заданную область экрана. Задается только область для вывода, остальные параметры отображения строки остаются неизменными. Разделителем является точка “.”. Позиция разделителя указывается в символах справа налево. Правый символ числа считается нулевым. Например в число “123456789” добавить разделитель в позицию 1, мы получим строку “12345678.9”. Если позиция разделителя (**0x0105**) выходит за рамки имеющегося числа, то слева добавляются нули. Например в число “1234” добавить разделитель в позицию 5, мы получим строку “0.01234”. Позиция разделителя может быть задана в диапазоне от 0 до 9. Максимальная длина строки вместе с разделителем (при наличии) не более 11 символов.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103 - 0x0104: (Int32) Число для вывода на экран,
Регистр 0x0105: Позиция символа-разделятеля справа налево (от 0 до 9).

0x0100	0x04	**Вывести рамку в область экрана**	**R W**	**Struct.** **9**

Вывести рамку в заданную область экрана, при этом содержимое области и параметры вывода не меняются, но текст будет автоматически смещен с учетом рамки. Рамка может быть цельной или перфорированной (пунктир), может не двигаться или вращаться по/против часовой стрелки. Есть возможность задать размер штриха, скорость прокрутки, цвет и режимы мигания.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103 - 0x0104: Цвет в формате ARGB8888 или RGB888. Компонента A задает степень прозрачности перед цветом фона. Может быть задана от 0xFF до 0x00, причем если значение A меньше 0x09, то компонента будет принята равной 0xFF,
Регистр 0x0105: Резерв,
Регистр 0x0106: Размер штриха в пикселах, 0 - цельная рамка без перфорации,

Регистр 0x0107: Скорость прокрутки перфорированной рамки. Если задан 0, то будет взято значение по умолчанию из файла “User*.txt”. Рабочие значения от 10 до 1 по 20мс. Это значит, что если задано значение 10, то рамка будет смещаться на 1 пиксель каждые 10 * 20 = 200мс - это медленная прокрутка. Если задано значение 1, то рамка будет смещаться на 1 пиксель каждые 1 * 20 = 20мс - это быстрая прокрутка,

Регистр 0x0108: Флаги отображения рамки. Если задан 0, то настройка останется без изменений. Имеет вид: 0b 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00, где каждый бит это флаг (0 или 1):

- 00 - (U) 0-скрыть рамку/1-показать рамку ,
- 01 - (U) 0-прокрутка рамки по часовой стрелке/1-прокрутка рамки против часовой стрелки,
- 02 - (U) 0-прокрутка есть/1-прокрутка остановлена.

Флаги обозначенные символом (U) доступны для чтения/записи пользователем, остальные только для чтения.

0x0100	0x05	**Очистить область от рамки**	**R W**	**Struct.** **3**

Очистить область от рамки, параметры вывода рамки будут утеряны. Для подтверждения команды в регистр 0x0102 должно быть записано число 0xC0.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Для подтверждения команды в регистр должно быть записано число 0xC0.

0x0100	0x06	**Загрузить коллекцию областей**	**R W**	**Struct.** **3**

Загрузить коллекцию областей, при этом предыдущий набор коллекций и все строки будут стерты. Все коллекции описываются в блоке “AreasCollection" файла “User*.txt”. Каждая коллекция - это массив. Элементы массива - это блоки параметров для каждой области (координаты начала и конца). Коллекция имеет вид “Collection_n”, где n - это номер коллекции. Номера коллекций могут следовать не по порядку, но не должны повторяться. Номер коллекции может быть задан числом от 0 до 255, например коллекция “Collection_128" имеет номер 128.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,

Регистр 0x0102: Номер коллекции.

0x0100	0x07	**Вывести экран в текущую коллекцию**	**R W**	**Struct.** **3**

Вывести экран в текущую коллекцию, при этом все строки предыдущего экрана будут стерты. Все экраны описываются в блоке “Screens" файла “User*.txt”. Каждый экран - это массив. Элементы массива - это блоки строк для вывода в каждую область (Текст строки или ссылка на строку, цвет, шрифт и выравнивание текста внутри области). Экран имеет вид “Screen_n”, где n - это номер экрана. Номера экранов могут следовать не по порядку, но не должны повторяться. они никак не связаны с номерами коллекций. Номер экрана может быть задан числом от 0 до 255, например экран “Screen_128" имеет номер 128.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Номер экрана.

0x0100	0x08	**Загрузить коллекцию областей и вывести в нее экран**	**R W**	**Struct.** **4**

Загрузить коллекцию областей и вывести в нее экран. Выполнение команды равносильно последовательному выполнению команд: загрузить коллекцию областей = “0x06” а затем вывести в нее экран = “0x07”.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Номер коллекции областей,
Регистр 0x0103: Номер экрана.

0x0100	0x09	**Изменить цвет текста в заданной области**	**R W**	**Struct.** **5**

Изменить цвет текста в заданной области. Цвет задается в формате RGB888, например красный цвет = 0xFF0000, зеленый = 0x00FF00, синий = 0x0000FF. Значение яркости каждой компоненты цвета может быть установлено от 0x00 до 0xFF.

Также поддерживается специальный формат цвета ARGB8888, где A-компонента - это степень прозрачности цвета к цвету фону. Когда A-компонента отсутствует, равна 0 или равна 0xFF, то цвет полностью непрозрачен. Если A-компонента задана от 0xFE до 0x09, то

прозрачность цвета будет увеличиваться и при значении 0x09 цвет будет практически полностью прозрачен к фону. Значения A-компоненты от 0 до 5 зарезервированны под специальные флаги цвета:

- 0 - A-компонента будет работать как будто задана = 0xFF,
- 1 - Случайное цикличное переключение цвета. При этом остальные компоненты RGB должны быть установлены в 0. Например: 0x01000000. Частота переключения цветов задается в файле “User*.txt” в блоке “Defaults”.”ColorRandomTime_100ms”.

Если в области был задан черный цвет = 0x000000 и поступила команда для изменения цвета со значением 0x000000 (или 0x00000000 в ARGB-формате), то будет применен цвет по умолчанию из файла “Uset*.txt” блок “Defaults”.”Color”.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103 - 0x0104: Цвет в формате ARGB888 или RGB888.

0x0100	0x0A	**Изменить размер шрифта в заданной области**	**R W**	**Struct.** **4**

Изменить размер шрифта в заданной области. Имеет вид 0xXY. Где X это масштаб по оси X, Y это масштаб по оси Y. Стандартный символ имеет размер 6 пикселей в ширину (ось X) и 8 пикселей в высоту (ось Y). Если задан масштаб 0x11, то символу будет обычного размера 6*8. Если масштаб: 0x21, то размер 12*8, если 0x12, то размер 6*16, если 0x33, то размер 18*24, и так далее. По каждой оси множитель может быть от 1 до F. Если масштаб задан одной цифрой например 0x08, то он будет принят как масштаб 0x88. Запись вида: 0xN0, где число N задано, а после стоит 0 считается недопустимой. Если передано значение 0, то будет применен размер заданный по умолчанию в файле “User*.txt” блок “Defaults”.”FontSize”.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103: Размер шрифта по осям X/Y.

0x0100	0x0B	**Изменить выравнивание текста в заданной области**	**R W**	**Struct.** **4**

Изменить выравнивание текста в заданной области. Имеет вид 0xXY. Где X может принимать значения: 1-выравнивание по левому краю, 2-по центру, 3-по правому краю. Y может принимать значения: 1-по верхнему краю, 2-по центру, 3-по нижнему краю. Если

передано значение 0, то будет применено выравнивание заданное по умолчанию в файле “User*.txt” блок “Defaults”.”Align”.

Для картинок и текста не помещающихся по высоте в заданную область всегда применяется выравнивание по оси Y по середине.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103: Выравнивание текста по осям X/Y.

0x0100	0x0C	**Изменить скорость прокрутки текста в заданной области**	**R W**	**Struct.** **4**

Изменить скорость прокрутки текста в заданной области (имеет смысл только для строк, не помещающихся в области по оси X). Рабочие значения от 10 до 1 и от -1 до масштаба заданного для оси X в FontSize (0x0105), по 20мс. Это значит, что если задано значение 10, то строка будет смещаться на 1 пиксель каждые 10 * 20 = 200мс, и текст будет двигаться медленно. Если задано значение 1, то строка будет смещаться на 1 пиксель каждые 1 * 20 = 20мс, и соответственно текст будет двигаться быстро. Значения со знаком “-“ дают еще большую скорость и ограниченны только выбранным масштабом текста по оси X. Если передано значение 0, то будет применена скорость заданная по умолчанию в файле “User*.txt” блок “Defaults”.”SpeedScrollingString”.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103: Скорость прокрутки текста.

0x0100	0x0D	**Изменить флаги вывода строки в заданной области**	**R W**	**Struct.** **4**

Изменить флаги вывода строки в заданной области. Если задан 0, то настройка останется без изменений. Имеет вид: 0b 15 14 13 12 11 10 09 08 07 06 05 04 03 02 01 00, где каждый бит это флаг (0 или 1):

- 00 - (U)показать строку ,
- 01 - (U)мигать (приоритет выше чем у мигания инвертируя цвет),
- 02 - (U)мигать инвертируя цвет строки,
- 03 - резерв,

  …
- 09 - строка только что принята (еще не выводилась на экран),

- 10 - (U)вывести строку только 1 раз (после полной прокрутки она исчезнет),
- 11 - строка закончила свой вывод (после полной прокрутки флаг поднимется автоматически).

Флаги обозначенные символом (U) доступны для чтения/записи пользователем, остальные только для чтения. Если нужно установить флаги, то в регистре обязательно устанавливается флаг 00 и любые флаги, доступные для записи.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области на экране,
Регистр 0x0103: Флаги отображения текста.

0x0100	0x0E	**Загрузить сцену**	**R W**	**Struct.**
				3

Загрузить сцену. Сцена - это набор состоящий из коллекции областей и привязанного к ней экрана. Сцены описаны в файле “User*.txt” в блоке “Scenes”.”Scene_n”, где n - это номер сцены. Номера сцен могут следовать не по порядку, но не должны повторяться. Номер сцены может быть задан числом от 0 до 255, например сцена “Scene_128" имеет номер 128.

Отличие от команды 0x08 (Загрузить коллекцию областей и вывести в нее экран), в том, что для сцен можно дополнительно задать время через которое экран автоматически переключится на другую сцену. Время для смены сцены задается в блоке “Scenes”.”Scene_n”.”TimeToJump_100ms”, а номер следующей сцены в блоке “Scenes”.”Scene_n”.”JumpToScene”. Таким образом можно реализовать автоматическую работу без участия контроллера (например для рекламы). Сцена загружаемая по умолчанию после включения задается в блоке “Defaults”.”Scene”.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Номер сцены.

0x0100	0x0F	**Очистить область (или весь экран)**	**R W**	**Struct.**
				4

Очистить область от мусора, при этом все что выводилось в этой области: строка, рамка и параметры их вывода будут утеряны. Для подтверждения команды в регистр 0x0102 должно быть записано число 0xC0. В регистре 0x0103 передается индекс области, индексы всегда следуют по порядку и начинаются с 0. Если индекс области задан как 0xFFFF, то будут

очищены все области экрана (после этого необходимо заново загрузить сцену, или коллекцию областей и экран).

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Для подтверждения команды в регистр должно быть записано число 0xC0,
Регистр 0x0103: Индекс области для очистки.

0x0100	0x10	**Сбросить контроллер (загрузить значения по умолчанию)**	**R W**	**Struct.**
				3

Команда не полностью сбрасывает контроллер, а очищает экран и загружает все настройки и сцену заданные по умолчанию. Для подтверждения команды в регистр 0x0102 должно быть записано число 0xC0.

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Для подтверждения команды в регистр должно быть записано число 0xC0.

0x0100	0x1E	**Получить размер свободной памяти в куче для строк**	**R W**	**Struct.**
				3

Команда на получение размера свободной памяти в куче для строк. *“Куча" - это область памяти, в которой может храниться какая-либо информация*, в нашем случае “Куча” это область для хранения строк. **Всего в куче для строк доступно 2000 байт**. Каждый символ занимает 1 байт, кроме того каждая структура описания строки также занимает память, около 32 - 40 байт для каждой строки в текущем экране.

Зная количество свободной памяти, разработчик может понять сколько еще символов он сможет записать в строки текущего экрана.

> *Команда предполагает возвращение результата, таким образом сначала нужно запустить команду на выполнение, а затем дождаться окончания и прочитать результат.*

Запуск на выполнение:
Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,

Чтение результата:

Регистр 0x0100: Номер последней команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Количество свободной памяти в куче для строк в байтах.

0x0100	0x1F	**Получить размер буфера для строки**	**R W**	**Struct.** **4**

Команда на получение размера буфера выделенного под строку в указанной области. Каждая область предназначена для вывода строк текста. Текст хранится в памяти контроллера, размер памяти отводимый под строку задается за ранее - это и есть буфер. Допустим мы хотим вывести в область строку “Съешь еще этих мягких булок, да выпей чаю”, эта строка имеет длину в 41 символ, таким образом нам нужен буфер с минимальной длиной в 41 символ, или больше. При выводе строк способами, отличными от команд “0x23” и “0x24”, буфер необходимого размера выделяется автоматически.

- *Если в данный момент выведена строка из статической памяти (например сообщение об ошибке), то размер буфера будет указан = 0, так как строка находится в памяти, которую нельзя перезаписать, для этого нужно сначала выделить память под буфер строки командой “0x20”.*
- *Команда предполагает возвращение результата, таким образом сначала нужно запустить команду на выполнение, а затем дождаться окончания и прочитать результат.*

Запуск на выполнение:
Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области, для которой нужно получить длину буфера.

Чтение результата:
Регистр 0x0100: Номер последней команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области,
Регистр 0x0103: Размер буфера под строку в байтах.

0x0100	0x20	**Задать размер буфера для строки**	**R W**	**Struct.** **4**

Команда на изменение размера буфера выделенного под строку в указанной области. Каждая область предназначена для вывода строк текста. Текст хранится в памяти контроллера, размер памяти отводимый под строку задается за ранее - это и есть буфер. Допустим мы хотим вывести в область строку “Съешь еще этих мягких булок, да выпей

чаю”, эта строка имеет длину в 41 символ, таким образом нам нужен буфер с минимальной длиной в 41 символ, или больше. При выводе строк способами, отличными от команд “0x23” и “0x24”, буфер необходимого размера выделяется автоматически.

Следует иметь в виду, что старый буфер со строкой будет удален, а буфер с новым размером будет пуст.

- *При выделении памяти под буфер, нет необходимости “закладывать” дополнительный байт под символ окончания строки “\0”, это происходит автоматически.*
- *Команда предполагает возвращение результата, таким образом сначала нужно запустить команду на выполнение, а затем дождаться окончания и прочитать результат.*

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области, для которой нужно задать длину буфера,
Регистр 0x0103: Размер буфера под строку в байтах.

0x0100	0x21	**Читать содержимое буфера строки (по 1 символу в регистр)**	**R W**	**Struct.**
				5 - 26

Команда чтения содержимого буфера выделенного под строку в указанной области. Каждый символ будет скопирован в младшую часть регистра, старшая часть всегда равна 0. Если запрошено количество символов больше чем доступно регистров (не более 26), то будет прочитано только доступное количество регистров.

- *Следует иметь в виду, что строки выводятся только до символа завершения строки “\0”, и если прочитан текст в котором встречается символ завершения строки, то текст после него на экран не выводился.*
- *Команда предполагает возвращение результата, таким образом сначала нужно запустить команду на выполнение, а затем дождаться окончания и прочитать результат.*

Запуск на выполнение:
Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области, из которой читать буфер,
Регистр 0x0103: Позиция символа с которого начать чтение (смещение). Первый символ строки имеет позицию 0,
Регистр 0x0104: Количество байт (кол-во символов).

Чтение результата:
Регистр 0x0100: Номер последней команды,
Регистр 0x0101: Статус выполнения,

Регистр 0x0102: Индекс области,
Регистр 0x0103: Позиция первого символа (смещение),
Регистр 0x0104: Количество байт (кол-во символов), прочитанное в действительности,
Регистр 0x0105: ASCII символ первой буквы,
Регистр 0x0106: ASCII символ второй буквы,
……
Регистр 0x0119: см. описание регистра 0x0105.

0x0100	0x22	**Читать содержимое буфера строки (по 2 символа в регистр)**	**R W**	**Struct.** **5 - 26**

Команда чтения содержимого буфера выделенного под строку в указанной области. Каждый регистр может хранить по 2 символа, в младшей части первый символ, в старшей второй. Если запрошено количество символов больше чем доступно регистров (не более 26), то будет прочитано только доступное количество регистров.

- *Следует иметь в виду, что строки выводятся только до символа завершения строки "\0", и если прочитан текст в котором встречается символ завершения строки, то текст после него на экран не выводился.*
- *Команда предполагает возвращение результата, таким образом сначала нужно запустить команду на выполнение, а затем дождаться окончания и прочитать результат.*

Запуск на выполнение:
Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области, из которой читать буфер,
Регистр 0x0103: Позиция символа с которого начать чтение (смещение). Первый символ строки имеет позицию 0,
Регистр 0x0104: Количество байт (кол-во символов).

Чтение результата:
Регистр 0x0100: Номер последней команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области,
Регистр 0x0103: Позиция первого символа (смещение),
Регистр 0x0104: Количество байт (кол-во символов) прочитанное в действительности,
Регистр 0x0105: ASCII символ второй буквы, ASCII символ первой буквы,
Регистр 0x0106: ASCII символ четвертой буквы, ASCII символ третьей буквы,
……
Регистр 0x0119: см. описание регистра 0x0105.

0x0100	0x23	Писать символы в буфер строки (по 1 символу в регистре)	R W	Struct. 5 - 26

Команда на запись символов в буфер строки в указанной области. В буфер будет скопирован каждый символ из младшей части регистра, старшая часть не имеет значения. Если происходит попытка записать количество символов больше чем доступно в буфере строки, то будет записано только доступное количество символов. Узнать размер буфера или задать его размер можно командами “0x1F” и “0x20” соответственно.

- *Если запись начата не с позиции 0, то в начале буфера могут быть символы с кодом окончания строки “\0”, следует иметь в виду, что строки выводятся только до символа завершения строки.*
- *Если нужно сначала полностью сформировать длинную строку, а только потом отобразить ее в области, то можно передать строку, у которой в позиции 0 находится символ завершения строки. А когда строка будет сформирована, то просто поменять нулевой символ на отличный от символа завершения строки. Текст сразу будет отображен в своей области.*
- *Команда предполагает возвращение результата, таким образом сначала нужно запустить команду на выполнение, а затем дождаться окончания и прочитать результат.*

Запуск на выполнение:

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области, в буфер которой писать,
Регистр 0x0103: Позиция символа с которого начать запись (смещение). Первый символ строки имеет позицию 0,
Регистр 0x0104: Количество байт (кол-во символов),
Регистр 0x0105: ASCII символ первой буквы,
Регистр 0x0106: ASCII символ второй буквы,
……
Регистр 0x0119: см. описание регистра 0x0105.

Чтение результата:

Регистр 0x0100: Номер последней команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области,
Регистр 0x0103: Позиция первого символа (смещение),
Регистр 0x0104: Количество байт (кол-во символов), записанное в действительности.

0x0100	0x24	Писать символы в буфер строки (по 2 символа в регистре)	R W	Struct. 5 - 26

Команда на запись символов в буфер строки в указанной области. В буфер будет скопирован каждый символ из регистра. В младшей части регистра находится первый символ, в старшей части второй. Если происходит попытка записать количество символов больше чем доступно в буфере строки, то будет записано только доступное количество символов. Узнать размер буфера или задать его размер можно командами “0x1F” и “0x20” соответственно.

- *Если запись начата не с позиции 0, то в начале буфера могут быть символы с кодом окончания строки “\0”, следует иметь в виду, что строки выводятся только до символа завершения строки.*
- *Если нужно сначала полностью сформировать длинную строку, а только потом отобразить ее в области, то можно передать строку, у которой в позиции 0 находится символ завершения строки. А когда строка будет сформирована, то просто поменять нулевой символ на отличный от символа завершения строки. Текст сразу будет отображен в своей области.*
- *Команда предполагает возвращение результата, таким образом сначала нужно запустить команду на выполнение, а затем дождаться окончания и прочитать результат.*

Запуск на выполнение:

Регистр 0x0100: Номер команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области, в буфер которой писать,
Регистр 0x0103: Позиция символа с которого начать запись (смещение). Первый символ строки имеет позицию 0,
Регистр 0x0104: Количество байт (кол-во символов),
Регистр 0x0105: ASCII символ второй буквы, ASCII символ первой буквы,
Регистр 0x0106: ASCII символ четвертой буквы, ASCII символ третьей буквы,
……
Регистр 0x0119: см. описание регистра 0x0105.

Чтение результата:

Регистр 0x0100: Номер последней команды,
Регистр 0x0101: Статус выполнения,
Регистр 0x0102: Индекс области,
Регистр 0x0103: Позиция первого символа (смещение),
Регистр 0x0104: Количество байт (кол-во символов), записанное в действительности.

4.4. Таблица регистров

Адрес	CMD	R W	Кол-во регистров	Описание	Значение	Заводское значение
0x2C		R	1	Номер текущей коллекции областей	Uint16	-
0x2D		R	1	Номер текущего экрана	Uint16	-
0x0100	0x00	R W	10-26	Вывести строку в область экрана	Структура	-
	0x01		4-26	Короткий вывод строки в область экрана	Структура	-
	0x02		12	Вывести число с разделителем в область экрана	Структура	-
	0x03		6	Короткий вывод числа с разделителем в область экрана	Структура	-
	0x04		9	Вывести рамку в область экрана	Структура	-
	0x05		3	Очистить область от рамки	0xC0	0
	0x06		3	Загрузить коллекцию областей	Номер коллекции	-
	0x07		3	Вывести экран в текущую коллекцию	Номер экрана	-
	0x08		4	Загрузить коллекцию областей и вывести в нее экран	Структура	-
	0x09		5	Изменить цвет текста в заданной области	Структура	-
	0x0A		4	Изменить размер шрифта в заданной области	Структура	-
	0x0B		4	Изменить выравнивание текста в заданной области	Структура	-
	0x0C		4	Изменить скорость прокрутки текста в заданной области	Структура	-
	0x0D		4	Изменить флаги вывода строки в заданной области	Структура	-
	0x0E		3	Загрузить сцену	Номер сцены	-
	0x0F		4	Очистить область (или весь экран)	Структура	-
	0x10		3	Сбросить контроллер (загрузить значения по умолчанию)	0xC0	0
	0x1E		3	Получить размер свободной памяти в куче для строк	Структура	-
	0x1F		4	Получить размер буфера для строки	Структура	-
	0x20		4	Задать размер буфера для строки	Структура	-
	0x21		5-26	Читать содержимое буфера строки (по 1 символу в регистр)	Структура	-

Адрес	CMD	R W	Кол-во регистров	Описание	Значение	Заводское значение
0x0100	0x22	R W	5-26	Читать содержимое буфера строки (по 2 символа в регистр)	Структура	-
	0x23		5-26	Писать символы в буфер строки (по 1 символу в регистре)	Структура	-
	0x24		5-26	Писать символы в буфер строки (по 2 символа в регистре)	Структура	-

5. Маркировка устройств / Информация для заказа

5.1. Контроллер LedMX

Маркировка	C	L	X	3	A	M	-	1	T	R	U
Номер поля	1	2		3		4		5	6	7	

1. Производитель: C - CYBERiDEA
2. Тип устройства: LX - контроллер для LED-экранов LedMX
3. Аппаратная платформа: 3A
4. Версия устройства/тип лицензии:
    - M - Mono,
    - L - Lite,
    - O - OEM,
    - C - Custom,
    - P - PRO.
5. Тип напряжения питания: 1 - DC (постоянный ток)
6. Напряжение питания:
    - T - 5V,
    - W - 5-24V.
7. Порты для связи:
    - S - USART (RS-232),
    - R - RS-485,
    - P - USART/RS-485 Интерфейсы занимают одни и те-же выводы,
    - U - USB,
    - i - Digital Input,
    - O - Digital Output,
    - N - Порты для связи отсутствуют.

5.2. LED-панель

LED-панели маркируются 2 способами: короткий для заказа и длинный для точной идентификации файла драйвера.

5.2.1. Короткая маркировка (для заказа)

Короткий вариант не включает особенности драйвера и обозначает лишь характеристики значимые для пользователя.

Маркировка	C	L	P	-	S	5	0	5	R	G	B
Номер поля	1	2			3	4	5	6	7		

1. Поставщик: C - CYBERiDEA
2. Тип устройства: LP - LED-панель
3. Тип светодиодов:
    - S - SMD,
    - D - DIP.
4. Размер панели:
    - 5 - 320*160 мм,
    - 4 - 192*192 мм,
    - 3 - 256*128 мм,

- 2 - 192*96 мм.

5. Зарезервировано
6. Разрешение панели:
   - B - 128x64px,
   - A - 64*64px,
   - 9 - 80*40px,
   - 8 - 64*32px,
   - 7 - 32*32px,
   - 6 - 40*20px,
   - 5 - 32*16px.
7. Цвет:
   - R - Red (красный),
   - G - Green (зеленый),
   - B - Blue (синий),
   - W - White (белый),
   - Y - Yellow (желтый),
   - RG - Red/Green (красный/зеленый),
   - RGB - RGB (полноцветный).

5.2.2. Длинная маркировка (для подбора драйвера)

Длинная маркировка предназначена для точной идентификации файла драйвера.

Маркировка	**C**	**L**	**P**	**5**	**0**	**8**	**-**	**U**	**7**	**2**	**0**	**0**	**2**	**0**	**D**	**5**	**X**	**F**	**0**	**0**	**4**	**0**	**2**
Номер поля	1	2		3	4			5	6	7						8	9	10	11		12	13		

1. Поставщик: C - CYBERiDEA
2. Тип устройства: LP - LED-панель
3. Шаг пикселя:
   - A - P10,
   - 8 - P8,
   - 6 - P6,
   - 5 - P5,
   - 4 - P4,
   - 3 - P3,
   - 2 - P2,
   - 9 - P2,5.
4. Тип сканирования:
   - 01 - Static/S1,
   - 02 - S2,
   - 04 - S4,
   - 08 - S8,
   - 16 - S16,
   - 32 - S32.
5. Исполнение:
   - U - Outdoor (для улицы),
   - N - Indoor (для помещений).
6. Цвет:
   - 1 - Red (красный),

- 2 - Green (зеленый),
- 3 - Blue (синий),
- 4 - White (белый),
- 5 - Yellow (желтый),
- 6 - Red/Green (красный/зеленый),
- 7 - RGB (полноцветный).

7. Служебные данные драйвера 1
8. Разрешение панели:
    - 2 - 32*16px,
    - 3 - 32*32px,
    - 4 - 40*20px,
    - 5 - 64*32px,
    - 6 - 64*64px,
    - 7 - 80*40px,
    - 8 - 128*64px.
9. Размер панели:
    - X - 320*160 мм,
    - M - 256*128 мм,
    - L - 192*192 мм,
    - S - 192*96 мм.
10. Порт:
    - E - HUB75E,
    - F - HUB75,
    - C - HUB12,
    - 8 - HUB08.
11. Служебные данные драйвера 2
12. Служебные данные драйвера 3
13. Служебные данные драйвера 4

5.3. Панель оператора

Панели оператора всегда содержат 1 контроллер версии LedMX Mono и 1 полноцветную LED-панель выбранного размера. Так-же в комплекте могут быть опции. Для удобства заказа маркировка практически полностью совпадает с короткой маркировкой LED-панели.

Маркировка	**C**	**O**	**P**	**-**	**S**	**5**	**0**	**5**	**F**	**2**
Номер поля	1	2			3	4	5	6	7	8

1. Поставщик: C - CYBERiDEA
2. Тип устройства: OP - Панель оператора
3. Тип светодиодов:
    - S - SMD,
    - D - DIP.
4. Размер панели:
    - 5 - 320*160 мм,
    - 4 - 192*192 мм,
    - 3 - 256*128 мм,
    - 2 - 192*96 мм.

5. Зарезервировано
6. Разрешение панели:
    - B - 128*64px,
    - A - 64*64px,
    - 9 - 80*40px,
    - 8 - 64*32px,
    - 7 - 32*32px,
    - 6 - 40*20px,
    - 5 - 32*16px.
7. Цвет: F - RGB (полноцветный)
8. Зарезервировано

Приложение А. Формат JSON

Формат JSON используется для представления объектов в виде текстовой строки. Это один из наиболее удобных форматов хорошо воспринимаемых человеком.
Формат предполагает записи вида:

- { … } - объект. Все объекты начинаются с фигурной скобки и завершаются фигурной скобкой с запятой,
- […] - массив, элементами могут быть другие объекты или значения, размерность массивов не задается,
- Значения одного из типов:
  - строки в двойных кавычках,
  - целое число может быть записано в двойных кавычках или без них. Число с плавающей запятой пишется только в двойных кавычках, разделителем дробной части может быть только точка “.”,
  - логическое значение true/false, двойные кавычки не обязательны.

Имя параметра всегда пишется в двойных кавычках, значение параметра может быть записано в двойных кавычках или без них в зависимости от его типа (строки всегда пишутся в двойных кавычках). Имя параметра отделяется от его значения символом двоеточия ”:”, пробелы запрещены. Параметры, объекты, массивы и элементы массивов разделяются запятой “,”. В данной реализации комментарии запрещены, весь текст является чувствительным к регистру, например записи “Count”:10 и “count”:10 не равнозначны. Каждое значение объекта может быть получено через обращение к нему по имени. Если “объект_1” вложен в “объект_0”, то значение объекта 1 будет получено через путь: “объект_0.объект_1”. Элементы массивов начинаются с нуля и неразрывно получают индекс +1 к каждому следующему элементу. Если “объект_1” находится в массиве “массив_0” и имеет индекс 1, то значение объекта 1 будет получено через путь: “массив_0[1].объект_1”.

Пример:

```
{
        “Параметр_0”:”Значение параметра 0",
        “Параметр_1”:"Значение параметра 1",

        “Блок_0”:
        {
                “Блок_0”:{ “Параметр_0”:0, “Параметр_1”:1, },
                “Блок_1”:{ “Параметр_0”:0, “Параметр_1”:1, },
        },

        “Массив_0”:
        [
                “Элемент 0”,
                “Элемент 1”,
                …
                “Элемент n",
        ],

        “Блок_1”:
        {
                “Массив_0”:
                [
                        { “Параметр_0”:0, “Параметр_1”:"1", “Параметр_2”:”777.50”,  },
                        { “Параметр_0”:1, },
```

```
                ],

                “Массив_1”:
                [
                        { “Параметр_0”:”Строка с текстом”, },
                        { “Параметр_0”:”Строка с текстом”,  “Параметр_1”:true, },
                ],
        },
},
```

Приложение В. Протокол Modbus

Протокол Modbus RTU является частью стандартного протокола Modicon-Modbus, с официальным стандартом можно ознакомиться по адресу:

http://modbus.org/docs/PI_MBUS_300.pdf.

Далее будут описаны основные принципы работы протокола, достаточные для полного использования возможностей устройства.

Устройства в сети Modbus работают по принципу Мастер -> Подчиненный (один ко многим). Это означает, что только Мастер сети может начать обмен (сделать Запрос), а Подчиненные всегда слушают сеть и выполняют команды или отвечают на запросы от Мастера.

Запрос мастера содержит адрес подчиненного, код функции, указывающий на необходимое действие, данные для отправки и контрольную сумму для проверки сообщения. Ответное сообщение от подчиненного содержит поля подтверждения выполнения команды, данные для возврата и контрольную сумму для проверки сообщения. Если найдена ошибка в принятом сообщении или подчиненный не может выполнить запрошенное действие, то подчиненный возвращает сообщение об ошибке.

Режим RTU

Система кодирования: - каждый символ состоит из 8-бит, с шестнадцатеричным значением 0x00 - 0xFF,
- каждый регистр имеет длину 16 бит (2 байта). Порядок передачи байт от старшего к младшему (Big-endian). (кроме контрольной суммы CRC),

Каждый байт содержит: - 1 стартовый бит,
- 8 информационных бит, младший значащий бит передается первым (LSB first),
- 1 бит проверки четности чет./нечет.; без бита, если проверка не требуется,
- 1 стоповый бит, если есть проверка четности; 2 бита, если нет проверки четности,

Контрольная сумма пакета: - CRC16, передается сначала младший байт, затем старший (Lo Hi),
- стартовое значение 0xFFFF,
- полином 0xA001.

В этом режиме каждое сообщение должно начинаться с тишины интервалом не менее 3,5 символов (T1-T2-T3-T4, как показано ниже), время передачи одного символа T равно скорости передачи в бодах/с. При реализации алгоритма удобное всего использовать таймаут на отсутствие принимаемых символов (равный времени передачи 4х символов), запускаемый при приеме первого символа сообщения и сбрасываемый при каждом новом полученном символе. Когда таймаут сработал, считается, что пакет принят полностью и готов к обработке. Первым полем сообщения является адрес устройства.

Все устройства Modbus постоянно слушают сеть, и каждое декодирует сообщение, чтобы понять адресовано оно или нет. *Сообщение должно оканчиваться интервалом*

тишины и если он меньше 3,5 символов, то следующее сообщение будет принято как продолжение текущего, тогда последнее поле с контрольной суммой CRC не совпадет.

НАЧАЛО	АДРЕС	ФУНКЦИЯ	ДАННЫЕ	CRC	КОНЕЦ
T1-T2-T3-T4	8 БИТ	8 БИТ	n x 8 БИТ	16 БИТ	T1-T2-T3-T4

Пример кадра с сообщением ModBus RTU.

Поле «Адрес» может содержать значения от 1 до 247 (от 248 до 255 зарезервированы) (*от 1 до 32 в прошивках до версии 1.5.0 включительно),* это адрес подчиненного устройства, для которого предназначено сообщение. Адрес 0 используется для широковещательных команд не требующих ответа, сразу для всех устройств (в функционале устройства не поддерживается)

Поле «Функция» содержит номер функции, которую должен выполнить подчиненный. Например чтение или запись значения регистра по заданному адресу. В зависимости от функции, подчиненный должен дать ответ о успешно выполненной операции или об ошибке.

Например мастер дает команду на чтение регистра:

0000 0011 (0x03)

Если команда выполнена успешно, то ответом будет сообщение с содержимым регистра. Если произошла шибка, то ответное сообщение модифицируется, установкой старшего бита в байте команды:

1000 0011 (0x83)

К коду команды для сообщения с ошибкой добавляется код ошибки, указывающий на тип возникшей проблемы. При этом мастер сети ModBus должен указать оператору на ошибки связи с устройством и/или попытаться отправить сообщение снова.

Поле «Данные» содержит полезные данные и в зависимости от номера функции может иметь разный формат.

Поле «Сумма CRC» содержит контрольную сумму для проверки целостности пакета.

Регистры ModBus

Стандарт ModBus предусматривает передачу данных в регистрах. Каждый регистр имеет длину 16 бит (2 байта). Порядок передачи байт от старшего к младшему (Big-endian). То есть, если регистр содержит значение 0xA1B2, то сначала будет передан байт 0xA1, а затем байт 0xB2.

Следует помнить, что некоторые значения в памяти устройства могут иметь длину 64 бита (4 байта), что соответствует 2-ум регистрам. В стандарте не определено в каком именно порядке должны передаваться регистры (старший-младший или младший-старший), это зависит от конкретного устройства. **Все устройства компании CYBERiDEA хранят данные в порядке от младшего к старшему (Little-endian)**. Это значит, что если будет

обращение к переменной со значением 0xA1B2C3D4, то сначала будет передан младший регистр (младшие 16 бит числа) со значением 0xC3D4, а затем старший регистр со значением 0xA1B2. По байтам порядок будет такой: 0xC3, 0xD4, 0xA1, 0xB2. Если происходит обращение к структуре данных длиной больше 2-ух байт, то все регистры также располагаются от младшего к старшему. Например структура из 10 байт “0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x09” будет передана так: 0x0809, 0x0607, 0x0405, 0x0203, 0x0001.

Расчет контрольной суммы (CRC-16)

Контрольная сумма пакета данных занимает 2 байта и располагается в конце пакета. Порядок байт: младший байт, затем старший (Lo Hi). Ниже представлена функция для расчета CRC, в параметрах передается указатель на буфер с пакетом и его длина.

```
uint16_t GetCRC16(char *buf, int len)
{
	uint16_t crc = 0xFFFF;

	for (int pos = 0; pos < len; pos++) {
		crc ^= (uint16_t)buf[pos];

		for (int i = 8; i != 0; i--) {
			if ((crc & 0x0001) != 0) {
				crc >>= 1;
				crc ^= 0xA001;
			}
			else
				crc >>= 1;
		}
	}

	return crc;
}
```

Передача пакета

При формировании пакета на передачу во временный буфер записывается сам пакет + 2 байта резервируются для контрольной суммы. CRC должно быть вычислено без учета зарезервированных байт. Результат записывается в зарезервированные байты и пакет передается в сеть.

Прием пакета

После приема, адрес указанный в пакете сравнивается с адресом устройства, если они совпадают, то пакет предназначен для этого устройства. Далее пакет проверяется на целостность, для этого вызывается функция подсчета CRC но уже для всего пакета, включая байты CRC. Если пакет не поврежден, то функция вернет 0 и устройство начинает выполнение команды.

- *Если адрес или контрольная сумма не совпали, то устройство ничего не отвечает,*

После успешного выполнения команды, устройство формирует ответ соответствующий номеру функции. Мастер принимает ответ и команда считается выполненной.

Приложение С. Таблицы кодирования текстовой информации

Для кодирования текстовой информации используются специальные таблицы символов. Основные символы цифр, знаков препинания, и английских букв кодируются стандартной таблицей символов ASCII. Для кодирования русских букв и некоторых дополнительных символов применяется кодировка Windows-1251.

Некоторые специальные символы могут не отображаться.

Основная таблица ASCII-символов

DEC	DEC	CHAR	DEC	HEX	CHAR	DEC	HEX	CHAR	DEC	HEX	CHAR
0	0x00	[NULL]	32	0x20	[SPACE]	64	0x40	@	96	0x60	`
1	0x01	[START OF HEADING]	33	0x21	!	65	0x41	A	97	0x61	a
2	0x02	[START OF TEXT]	34	0x22	"	66	0x42	B	98	0x62	b
3	0x03	[END OF TEXT]	35	0x23	#	67	0x43	C	99	0x63	c
4	0x04	[END OF TRANS.]	36	0x24	$	68	0x44	D	100	0x64	d
5	0x05	[ENQUIRY]	37	0x25	%	69	0x45	E	101	0x65	e
6	0x06	[ACKNOWLEDGE]	38	0x26	&	70	0x46	F	102	0x66	f
7	0x07	[BELL]	39	0x27	'	71	0x47	G	103	0x67	g
8	0x08	[BACKSPACE]	40	0x28	(	72	0x48	H	104	0x68	h
9	0x09	[HORIZONTAL TAB]	41	0x29	)	73	0x49	I	105	0x69	i
10	0x0A	[LINE FEED]	42	0x2A	*	74	0x4A	J	106	0x6A	j
11	0x0B	[VARTICAL TAB]	43	0x2B	+	75	0x4B	K	107	0x6B	k
12	0x0C	[FROM FEED]	44	0x2C	,	76	0x4C	L	108	0x6C	l
13	0x0D	[CARRIAGE RETURN]	45	0x2D	-	77	0x4D	M	109	0x6D	m
14	0x0E	[SHIFT OUT]	46	0x2E	.	78	0x4E	N	110	0x6E	n
15	0x0F	[SHIFT IN]	47	0x2F	/	79	0x4F	O	111	0x6F	o
16	0x10	[DATA LINK ESCAPE]	48	0x30	0	80	0x50	P	112	0x70	p
17	0x11	[DEVICE CONTROL 1]	49	0x31	1	81	0x51	Q	113	0x71	q
18	0x12	[DEVICE CONTROL 2]	50	0x32	2	82	0x52	R	114	0x72	r
19	0x13	[DEVICE CONTROL 3]	51	0x33	3	83	0x53	S	115	0x73	s
20	0x14	[DEVICE CONTROL 4]	52	0x34	4	84	0x54	T	116	0x74	t
21	0x15	[NEGATIVE ACK.]	53	0x35	5	85	0x55	U	117	0x75	u
22	0x16	[SYNCHRONOUS IDLE]	54	0x36	6	86	0x56	V	118	0x76	v
23	0x17	[ENG OF TRANS. BLOCK]	55	0x37	7	87	0x57	W	119	0x77	w
24	0x18	[CANCEL]	56	0x38	8	88	0x58	X	120	0x78	x
25	0x19	[END OF MEDIUM]	57	0x39	9	89	0x59	Y	121	0x79	y
26	0x1A	[SUBSTITUTE]	58	0x3A	:	90	0x5A	Z	122	0x7A	z
27	0x1B	[ESCAPE]	59	0x3B	;	91	0x5B	[	123	0x7B	{
28	0x1C	[FILE SEPARATOR]	60	0x3C	<	92	0x5C	\	124	0x7C	|
29	0x1D	[GROUP SEPARATOR]	61	0x3D	=	93	0x5D	]	125	0x7D	}
30	0x1E	[RECORD SEPARATOR]	62	0x3E	>	94	0x5E	^	126	0x7E	~
31	0x1F	[UNIT SEPARATOR]	63	0x3F	?	95	0x5F	_	127	0x7F	[DEL]

Расширенная таблица ASCII-символов. Кодировка Windows-1251

Буквы “Ё” и “ё” не выводятся и будут заменены на “Е” и “е” соответственно.

DEC	HEX	CHAR	DEC	HEX	CHAR	DEC	HEX	CHAR	DEC	HEX	CHAR
128	0x80	Ђ	160	0xA0	[пробел]	192	0xC0	А	224	0xE0	а
129	0x81	Ѓ	161	0xA1	Ў	193	0xC1	Б	225	0xE1	б
130	0x82	‚	162	0xA2	ў	194	0xC2	В	226	0xE2	в
131	0x83	ѓ	163	0xA3	J	195	0xC3	Г	227	0xE3	г
132	0x84	„	164	0xA4	¤	196	0xC4	Д	228	0xE4	д
133	0x85	…	165	0xA5	Ґ	197	0xC5	Е	229	0xE5	е
134	0x86	†	166	0xA6	¦	198	0xC6	Ж	230	0xE6	ж
135	0x87	‡	167	0xA7	§	199	0xC7	З	231	0xE7	з
136	0x88	€	168	0xA8	Ё	200	0xC8	И	232	0xE8	и
137	0x89	‰	169	0xA9	©	201	0xC9	Й	233	0xE9	й
138	0x8A	Љ	170	0xAA	Є	202	0xCA	К	234	0xEA	к
139	0x8B	‹	171	0xAB	«	203	0xCB	Л	235	0xEB	л
140	0x8C	Њ	172	0xAC	¬	204	0xCC	М	236	0xEC	м
141	0x8D	Ќ	173	0xAD	-	205	0xCD	Н	237	0xED	н
142	0x8E	Ћ	174	0xAE	®	206	0xCE	О	238	0xEE	о
143	0x8F	Џ	175	0xAF	Ї	207	0xCF	П	239	0xEF	п
144	0x90	ђ	176	0xB0	°	208	0xD0	Р	240	0xF0	р
145	0x91	‘	177	0xB1	±	209	0xD1	С	241	0xF1	с
146	0x92	’	178	0xB2	I	210	0xD2	Т	242	0xF2	т
147	0x93	“	179	0xB3	i	211	0xD3	У	243	0xF3	у
148	0x94	”	180	0xB4	ґ	212	0xD4	Ф	244	0xF4	ф
149	0x95	•	181	0xB5	µ	213	0xD5	Х	245	0xF5	х
150	0x96	–	182	0xB6	¶	214	0xD6	Ц	246	0xF6	ц
151	0x97	—	183	0xB7	·	215	0xD7	Ч	247	0xF7	ч
152	0x98	[NULL]	184	0xB8	ё	216	0xD8	Ш	248	0xF8	ш
153	0x99	™	185	0xB9	№	217	0xD9	Щ	249	0xF9	щ
154	0x9A	љ	186	0xBA	є	218	0xDA	Ъ	250	0xFA	ъ
155	0x9B	›	187	0xBB	»	219	0xDB	Ы	251	0xFB	ы
156	0x9C	њ	188	0xBC	j	220	0xDC	Ь	252	0xFC	ь
157	0x9D	ќ	189	0xBD	S	221	0xDD	Э	253	0xFD	э
158	0x9E	ћ	190	0xBE	s	222	0xDE	Ю	254	0xFE	ю
159	0x9F	џ	191	0xBF	ї	223	0xDF	Я	255	0xFF	я