

# **Избранные статьи из руководства по программированию и применению контроллеров CANNY®**

**Данный документ автоматически сформирован на  
основании содержания online-энциклопедии CANNY.**

**Актуальная версия документации доступна на сайте  
<http://wiki.canny.ru/>**

<b>1 CANNY 7.....</b>	<b>1</b>
1.1 Общие сведения.....	1
1.2 Устройство и принцип работы.....	2
1.3 Режимы работы.....	5
1.4 Среда исполнения функциональных диаграмм.....	6
1.5 Ресурсы контроллера.....	8
1.6 Технические требования.....	14
<b>2 Общие сведения о ПЛК.....</b>	<b>16</b>
2.1 Основные определения и используемые сокращения.....	16
2.2 Что такое контроллер?.....	16
2.3 Программное обеспечение ПЛК.....	16
2.4 Как работает ПЛК.....	17
2.5 Программирование без программиста.....	18
<b>3 CANNY 7, Системные ресурсы и режимы работы.....</b>	<b>19</b>
3.1 Общее описание.....	19
3.2 Сброс контроллера.....	19
3.3 Встроенный светодиод контроллера.....	20
3.4 Режим пониженного энергопотребления.....	20
3.5 Сторожевой таймер.....	22
3.6 Фактическое время выполнения функциональной диаграммы.....	23
3.7 Идентификатор устройства.....	24
3.8 Контроль активности интерфейсов контроллера.....	25
3.9 Идентификатор вендора устройства.....	25
<b>4 CANNY 7, Драйвер каналов ввода-вывода.....</b>	<b>27</b>
4.1 Общее описание.....	27
4.2 Регистры драйвера.....	27
4.3 Нейтральное состояние канала.....	30
4.4 Режим дискретного выхода.....	30
4.5 Режим широтно-импульсного выхода.....	31
4.6 Режим дискретного входа.....	33
4.7 Режим счетчика.....	34
4.8 Эквивалентные принципиальные электрические схемы.....	35
4.9 Электрическая защита.....	37

<b>5 CANNY 7, Драйвер высокочастотного широтно-импульсного модулятора (ВЧ ШИМ).....</b>	<b>39</b>
5.1 Общее описание.....	39
5.2 Регистры драйвера.....	39
<b>6 CANNY 7, Драйвер UART - RS232 - Modbus.....</b>	<b>41</b>
6.1 Общее описание.....	41
6.2 Регистры драйвера.....	42
6.3 Работа контроллера в режиме UART.....	44
6.4 Работа контроллера в режиме RS-232.....	45
6.5 Реализация Modbus RTU.....	46
<b>7 CANNY 7, Драйвер CAN.....</b>	<b>49</b>
7.1 Общее описание.....	49
7.2 Регистры драйвера.....	49
7.3 Примеры.....	53
<b>8 CANNY 7, Драйвер LIN.....</b>	<b>55</b>
8.1 Общее описание.....	55
8.2 Регистры драйвера.....	55
8.3 Работа контроллера в режиме MASTER.....	59
8.4 Работа контроллера в режиме SLAVE.....	60
8.5 Работа контроллера в режиме MULTISLAVE.....	61
8.6 Режим экономии энергии (режим пониженного энергопотребления).....	62
<b>9 CANNY 7, Драйвер I2C.....</b>	<b>64</b>
9.1 Общее описание.....	64
9.2 Регистры драйвера I2C.....	64
9.3 Особенности работы драйвера I2C.....	66
<b>10 CANNY 7, Драйвер Dallas 1-Wire.....</b>	<b>68</b>
10.1 Общее описание.....	68
10.2 Регистры драйвера 1-Wire.....	68
<b>11 CANNY 7, Параметры пользовательской конфигурации.....</b>	<b>73</b>
11.1 Общее описание.....	73
11.2 Регистры параметров пользовательской конфигурации.....	73
11.3 Пример использования параметров пользовательской конфигурации.....	74

<b>12 CANNY 7, Энергонезависимая память (ЭНП).....</b>	<b>76</b>
12.1 Общее описание.....	76
12.2 Регистры энергонезависимой памяти.....	76
<b>13 CANNY 7, Драйвер пульта ИК ДУ.....</b>	<b>78</b>
13.1 Общее описание.....	78
13.2 Регистры драйвера пульта ИК ДУ.....	78

# 1 CANNY 7

**CANNY 7** — компактный программируемый логический контроллер ориентированный на автомобильное, бытовое и промышленное применение.

## 1.1 Общие сведения

Программируемый логический контроллер CANNY7, ввиду небольшого количества внешних каналов, может быть отнесен к классу интеллектуальных реле или NanoPLC. Тем не менее, возможностей CANNY7 достаточно для решения многих задач автоматизации, контроля и управления.

Уникальность CANNY7 заключается в совокупности особенностей, позволяющих назвать его первым программируемым логическим контроллером ориентированным на автомобильное применение. Такими особенностями являются:

- номинальное напряжение питания и каналов ввода-вывода 0 / 12В (18В max);
- максимальный выходной ток каждого из 11 каналов ввода-вывода контроллера: +/-120мА, достаточен для управления типовыми автомобильными реле;
- интерфейс CAN 2.0В совместимый с ISO-11898, SAE J2411 широко применяемым в автомобилях;
- встроенные средства управления собственным энергопотреблением контроллера в диапазоне от 5 до 60мА, позволяющие экономно расходовать заряд аккумулятора во время простоя автомобиля;
- энергонезависимая память программ и шестьдесят четыре 16-и битные ячейки энергонезависимой памяти данных доступные пользователю приложению, способные сохранить критически важные данные при сбоях питания;
- широкий диапазон рабочих температур от -40 до + 85 оС;
- встроенная защита от высоковольтных выбросов и короткого замыкания;
- компактный корпус соответствующий классу защиты IP50 подходит для монтажа и эксплуатации в составе оборудования кабины автомобиля;
- пакеты специализированного системного и прикладного программного обеспечения для работы с автомобильными сетями CAN и LIN.

Для написания пользовательских программ CANNY 7 создан графический язык программирования CFD, позволяющий быстро создавать эффективные пользовательские приложения — функциональные диаграммы. Бесплатная интегрированная среда разработки CannyLab, содержит средства редактирования, отладки и записи программного обеспечения в контроллер.

Для записи программного обеспечения в контроллер не требуется специального оборудования, необходим только ПК с портом USB версии 1.1 или выше, с установленной на нем средой CannyLab и стандартный кабель miniUSB.

Доступный пользователю объем памяти контроллера способен вместить программы, состоящие из нескольких сотен функциональных блоков, что позволяет реализовать достаточно сложные алгоритмы.

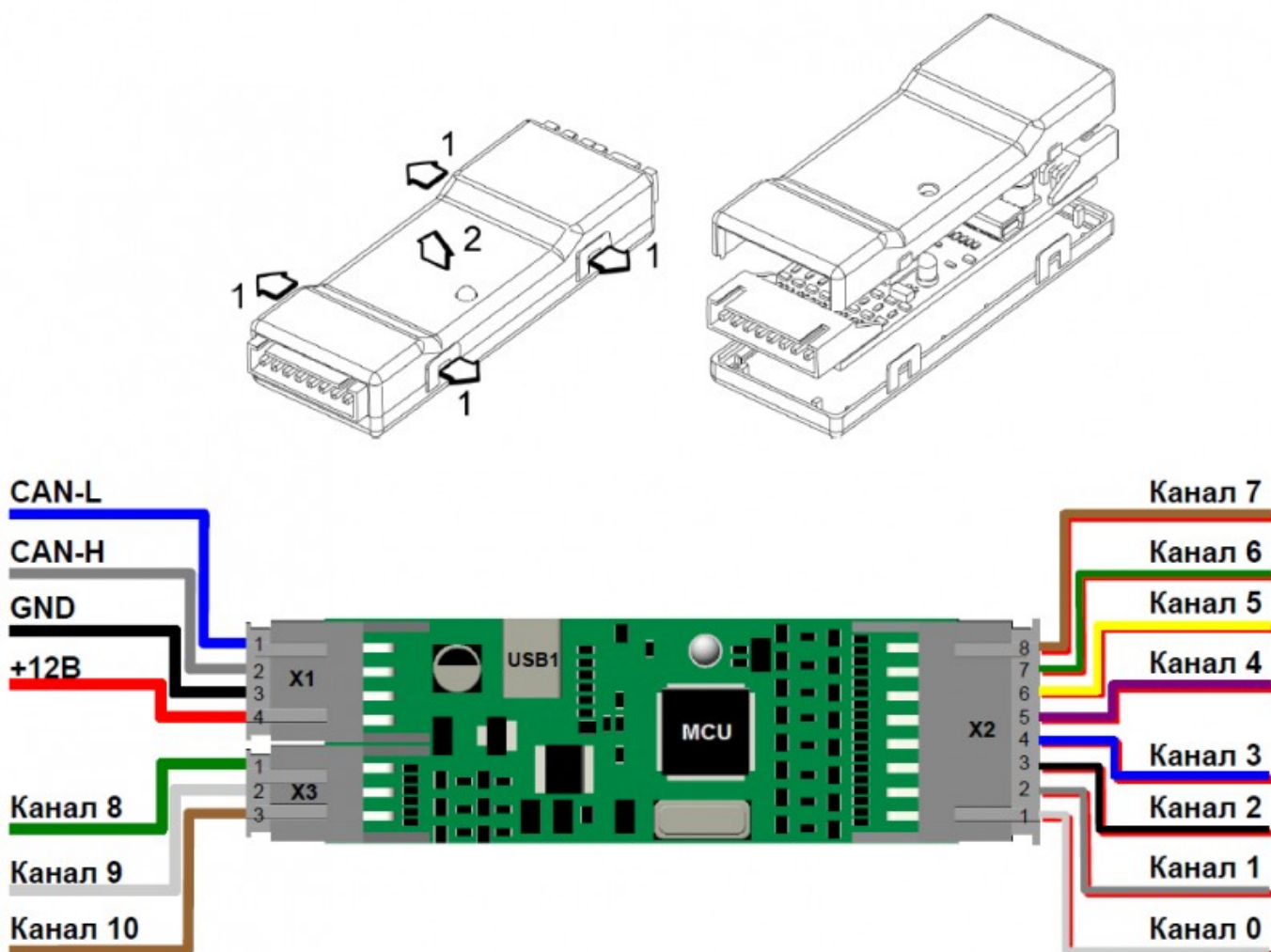
Каждый из 11 каналов ввода-вывода может работать в любом из 98 режимов, определяющих напряжение, ток и временные параметры входного и выходного сигналов. Кроме того, отдельные каналы могут быть сконфигурированы для работы в цифровом режиме, для приема/передачи данных таких протоколов как 1-Wire, I2C, RS-232, LIN. Конфигурация любого канала может быть установлена и изменена из пользовательского приложения.

Двухцветный светодиодный индикатор, управляемый из пользовательского приложения удобен для индикации режимов работы контроллера и диагностики.

## **1.2 Устройство и принцип работы**

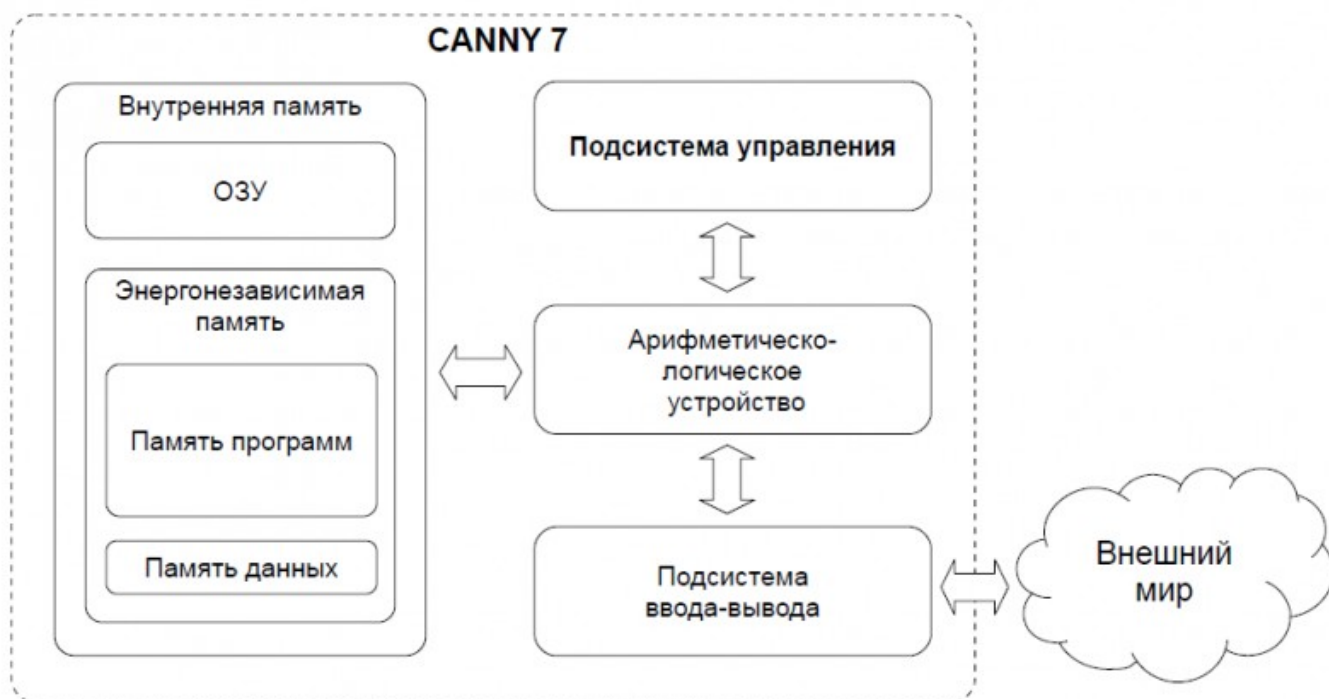
### **1.2.1 Внешний вид и расположение элементов**

Основными конструктивными элементами CANNY 7 являются: микроконтроллер (MCU) со вспомогательными цепями, система электропитания всех элементов контроллера, схема согласования электрических уровней каналов ввода-вывода, система электрической защиты, разъемы и индикаторный светодиод, размещенные на единой печатной плате 65 x 23 мм установленной внутри быстроразборного пластикового корпуса. Контроллер имеет три наружных разъема и один внутренний. Для подключения контроллера к питанию и внешним устройствам, в комплект его поставки включен набор соединительных жгутов. Наружный разъем X1 содержит четыре контакта: вход питания +12В, вход питания GND, CAN-H и CAN-L. Наружный разъем X2 содержит восемь контактов, соответствующих первым восьми каналам контроллера, начиная с канала №0 и заканчивая каналом №7. Наружный разъем X3 содержит три контакта, соответствующих каналам №8, №9 и №10 контроллера. Внутренний разъем USB1 контроллера служит для подключения интерфейсного кабеля miniUSB связывающего контроллер с ПК.



## 1.2.2 Программная архитектура

CANNY 7 является цифровым программируемым вычислительным управляющим устройством. В целом, для CANNY 7 справедливы общие сведения о программируемых логических контроллерах изложенные во введении к настоящему руководству. Основными элементами CANNY 7 являются: арифметическо-логическое устройство (АЛУ), внутренняя память, подсистема управления ходом исполнения команд и система ввода-вывода.



Арифметическо-логическое устройство — вычислительное ядро CANNY 7. АЛУ обеспечивает исполнение системного программного обеспечения и пользовательских функциональных диаграмм, помещенных во внутреннюю память контроллера. Внутренняя память контроллера разделяется на энергонезависимую память программ, энергонезависимую память данных и оперативную память данных. Подсистема управления ходом обработки команд, отвечает за переключение и настройку режимов работы контроллера. Система ввода-вывода обеспечивает связь контроллера с внешним миром, с использованием как дискретных каналов ввода-вывода, так и стандартных цифровых интерфейсов CAN / LIN / RS232 / USB.

### 1.2.3 Структура программного обеспечения



Программное обеспечение CANNY 7 состоит из: программного загрузчика, системного ПО (операционной системы и драйверов) и пользовательской функциональной диаграммы.



Программный загрузчик обеспечивает работу контроллера в режиме загрузки ПО, организуя передачу данных между CANNY 7 и персональным компьютером по протоколу USB, осуществляет проверку целостности и запись переданного от ПК программного обеспечения во внутреннюю память контроллера. Программный загрузчик помещается во внутреннюю память контроллера в процессе его производства и не может быть удален или изменен пользователем. Системное программное обеспечение CANNY 7 распространяется производителем в виде файлов формата CCX и содержит операционную систему и набор драйверов, обеспечивающих исполнение пользовательской функциональной диаграммы и её взаимодействие с ресурсами контроллера. Модификация пользователем содержимого данных файлов не допускается. Содержимое различных файлов CCX может быть многократно записано пользователем в контроллер. Пользовательская функциональная диаграмма создается и модифицируется пользователем в интегрированной среде разработки CannyLab и, после записи в контроллер, задает алгоритм его работы в автономном режиме. Пользовательские диаграммы могут быть многократно записаны в контроллер и сохранены из среды CannyLab в файлы формата CFD.

## **1.3 Режимы работы**

Предусмотрено несколько режимов работы контроллера, предназначенных для выполнения основных операций с ним.

### **1.3.1 Режим загрузки ПО**

В данном режиме, контроллер функционирует под управлением встроенного программного загрузчика, выполняющего запись системного программного обеспечения и функциональной диаграммы в контроллер по командам CannyLab. Вход в режим осуществляется автоматически, при установлении соединения контроллера с ПК по интерфейсу USB. При переходе в данный режим выполняется общий сброс контроллера: исполнение контроллером функциональной диаграммы прекращается, каналы ввода-вывода контроллера переводятся в нейтральное состояние, включается встроенный зеленый светодиод контроллера. При установлении связи с контроллером со стороны программного обеспечения ПК, зеленый светодиод контроллера переходит в мерцающий режим. Выход из данного режима происходит автоматически, при разрыве соединения контроллера с ПК. Если в момент выхода из режима загрузки ПО, энергонезависимая память программ контроллера содержала корректно записанное системное программное обеспечение, то контроллер переходит в автономный режим работы, в противном случае, происходит возврат в режим загрузки ПО.

### **1.3.2 Автономный режим**

Автономный режим является основным режимом работы контроллера. В данном режиме контроллер под управлением загруженного в него системного программного обеспечения последовательно, в бесконечном цикле, исполняет функциональную диаграмму, работая по алгоритму заданному пользователем. Переход в данный режим происходит автоматически, при подключении контроллера к внешнему питанию 12В в отсутствие USB соединения. При работе в данном режиме, функциональной диаграмме пользователя доступны все ресурсы контроллера, драйверы которых включены в загруженное системное программное обеспечение.

### **1.3.3 Автономный режим пониженного энергопотребления**

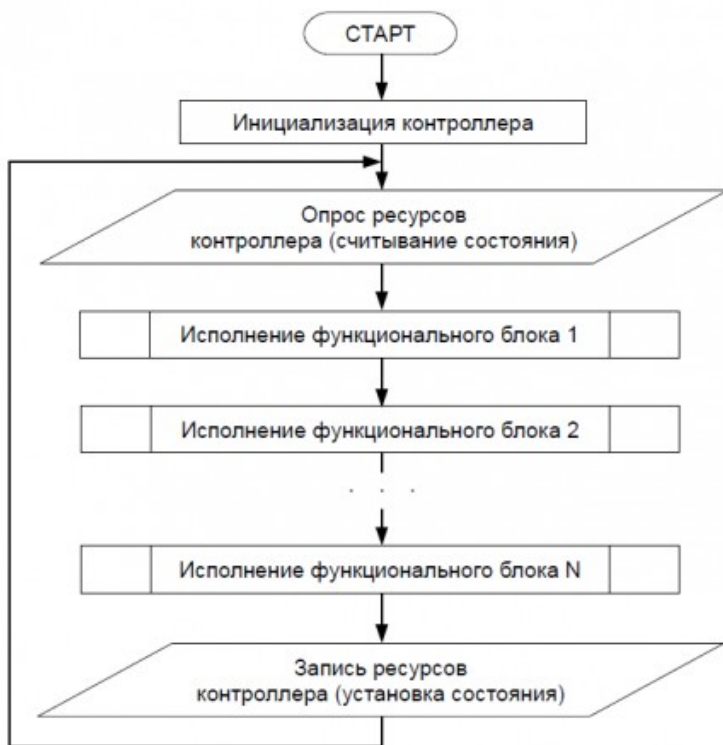
Данный режим является вариантом обычного автономного режима, в котором после каждого цикла исполнения функциональной диаграммы, контроллер делает паузу в работе, снижая своё энергопотребление до минимального. Таким образом, контроллер работает в пульсирующем режиме, периодически «засыпая» и «просыпаясь». Включением, отключением и настройкой параметров данного режима управляет функциональная диаграмма. Использование данного режима актуально при разработке систем, ориентированных на батарейное питание, таких как бортовое автомобильное оборудование.

## **1.4 Среда исполнения функциональных диаграмм**

### **1.4.1 Представление функциональной диаграммы**

Созданная в среде CannyLab графическая функциональная диаграмма, непосредственно перед записью в контроллер автоматически обрабатывается транслятором, который выполняет проверку диаграммы на непротиворечивость, определяет порядок выполнения функциональных блоков и преобразует диаграмму в исполняемый код — последовательность машинных команд АЛУ контроллера CANNY 7.

### **1.4.2 Порядок исполнения**



Исполняемый код диаграммы, при записи в контроллер уже содержащий системное программное обеспечение, включается в последовательность машинных команд системного ПО. Таким образом, общая последовательность команд контроллера с загруженным системным ПО и функциональной диаграммой, будет состоять из: процедуры инициализации, исполняемой однократно после каждого сброса контроллера и исполняемого кода функциональной диаграммы, обрамленного процедурами управления ресурсами контроллера, и помещенного в бесконечно исполняемый цикл – *цикл выполнения диаграммы*.

Некоторые драйверы, включенные в состав системного ПО контроллера, например драйвер CAN, требуют безотлагательной реакции контроллера на возникающие в процессе приема и передачи данных программные события. Программный код таких драйверов обрабатывается контроллером асинхронно, параллельно с основным потоком исполнения. На время обработки асинхронных вызовов драйверов, исполнение основного цикла выполнения диаграммы временно приостанавливается.

### 1.4.3 Доступ к ресурсам контроллера

Все доступные пользователю из функциональной диаграммы ресурсы: системные ресурсы контроллера, подсистема ввода-вывода и дополнительные драйверы включенные в состав системного ПО, отображаются на защищенное адресное пространство внутренней памяти контроллера. Данное адресное пространство разделено на регистры чтения (контроля) и регистры записи.

Пользователь имеет возможность указать регистр чтения в качестве источника входных

данных практически любого функционального блока на диаграмме и, тем самым, извлечь и использовать при реализации собственных алгоритмов сведения, полученные контроллером из внешнего мира. Например информацию об электрическом потенциале на каком-либо контакте контроллера, или содержимое пакета данных принятого контроллером из CAN.

Регистр записи может быть использован в качестве получателя выходных данных любого функционального блока на диаграмме. Таким образом, пользователь осуществляет управление ресурсами контроллера из функциональной диаграммы, получая возможность воздействовать на объекты внешнего мира. Например, переключить внешнее реле, изменив электрический потенциал на одном из контактов контроллера, к которому подключена его обмотка; включить контрольный светодиод; задать режим работы CAN; отправить пакет данных.

Порядок использования большинства ресурсов контроллера включает в себя задание пользователем необходимых параметров их работы, например полярности выходных каналов, полярности и чувствительности входных каналов, скорости обмена данными по CAN и т. д.

Задание таких параметров производится в форме записи специальных констант в один или в несколько определенных регистров контроллера, в зависимости от того, конфигурацию какого из ресурсов требуется задать. Например, установкой константы со значением 121 в регистр, расположенный по адресу 2432 задается режим работы канала №0 в качестве выхода положительной полярности.

Стандартный положительный выход — Рег.конфиг.канал 0

В среде CannyLab, для удобства пользователя, все доступные регистры контроллера поименованы, как и все специальные константы, использующиеся при взаимодействии с ресурсами контроллера. Поэтому для пользователя CannyLab данная операция будет выглядеть как установка константы с именем «Стандартный положительный выход» в регистр с именем «Регистр конфигурации канала №0».

Рег.вх.знач.канал 0 — значение на входе

Установив таким образом режим работы канала №0, мы можем по появлению значения «1» в регистре расположенном по адресу 2465 («Регистр входного значения канала №0»), узнать о приложении положительного электрического потенциала к контакту №1 разъема X2 контроллера.

## 1.5 Ресурсы контроллера

## 1.5.1 Системные ресурсы и режимы работы

*Основная статья:* **CANNY 7, Системные ресурсы и режимы работы**

Системные ресурсы контроллера отображаются на группу регистров чтения и группу регистров записи. Обращаясь к данным регистрам из функциональной диаграммы, можно получить востребованные в практическом применении сведения о текущем состоянии контроллера и управлять режимами его работы. Список регистров системных ресурсов находится в разделе «Состояние контроллера» справочника регистров контроллера, который доступен пользователю CANNY Lab через контекстное меню входов и выходов типа «Адрес» функциональных блоков.

## 1.5.2 Драйвер каналов ввода-вывода

*Основная статья:* **CANNY 7, Драйвер каналов ввода-вывода**

Пользователям CANNY 7 доступны одиннадцать дискретных каналов ввода-вывода общего назначения. Каждый канал физически представлен соответствующим контактом разъема X2 (Каналы №№0..7) либо разъема X3 (Каналы №№8,9 и 10) контроллера. Записывая и считывая данные соответствующих регистров драйвера, функциональная диаграмма может как управлять электрическим потенциалом на каждом из этих контактов так и получать информацию о текущем значении потенциала каждого из них.

Физические характеристики каналов позволяют подключать к ним различные внешние исполнительные устройства — электромагнитные реле, небольшие электродвигатели, светодиоды, слаботочные цепи управления оборудованием. В качестве внешних источников дискретных сигналов способных управлять работой контроллера, возможно использовать механические, электромеханические и электронные кнопки и переключатели, генераторы импульсов, источники напряжения 0-12В, транзисторные выходы различной аппаратуры и т.п.

Режим и параметры работы любого из каналов задаются функциональной диаграммой. В каждый момент времени канал может работать только в одном из возможных режимов, однако допускается динамическое переопределение конфигурации канала из функциональной диаграммы в процессе ее выполнения.

## 1.5.3 Драйвер высокочастотного широтно-импульсного модулятора (ВЧ ШИМ)

*Основная статья:* **CANNY 7, Драйвер высокочастотного широтно-импульсного модулятора (ВЧ ШИМ)**

Два из одиннадцати каналов ввода-вывода (Канал №1 и Канал №2) CANNY 7 поддерживают работу в режиме высокочастотного широтно-импульсного модулятора. Каналы могут быть задействованы независимо друг от друга и иметь независимые настройки скважности сигнала и подтяжки линии, однако, период высокочастотного ШИМ является параметром, общим для обоих каналов. В режиме ВЧ ШИМ, временные параметры ШИМ – период и скважность задаются в диапазоне от 2 до 20000 микросекунд, с шагом 1 микросекунда.

В режиме ВЧ ШИМ канал имеет фиксированную полярность импульсов — GND 100мА. Генерация может вестись как в режиме открытого коллектора – подтяжка линии отсутствует или внешняя, так и в режиме с внутренней подтяжкой к +12В (задается установкой значения в соответствующем регистре). В данном режиме канал работает асинхронно функциональной диаграмме, что позволяет добиться максимальной стабильности временных параметров генерируемого сигнала.

*Примечание: В режиме высокочастотного широтно-импульсного модулятора электрическая защита канала от короткого замыкания находится в отключенном состоянии! Перегрузка или короткое замыкание каналов контроллера находящихся в режиме ВЧ ШИМ может привести к выходу канала контроллера из строя!*

#### 1.5.4 Драйвер UART / RS232 / Modbus

*Основная статья: CANNY 7, Драйвер UART - RS232 - Modbus*

Два из одиннадцати каналов ввода-вывода (Канал №9 и Канал №10) CANNY 7 поддерживают работу в режиме приема/передачи данных последовательных протоколов UART, RS-232 и могут быть использованы для связи контроллеров друг с другом или с внешним оборудованием поддерживающим данные протоколы связи. Каналы могут быть задействованы независимо друг от друга и иметь индивидуальные настройки скорости передачи данных, типа и конфигурации используемого протокола, подтяжки линии.

Реализация **UART** в контроллерах CANNY7 позволяет организовать последовательный прием и передачу данных по одному проводу в полудуплексном режиме. Таким образом CANNY7 может иметь 2 независимых подключения с использованием протокола UART. Контроль состояния канала передачи данных должен осуществляться пользователем из функциональной диаграммы. Если канал свободен, то устройство может начать передачу данных, в противном случае устройство должно дожидаться освобождения линии.

Реализация протокола **RS-232** в контроллерах CANNY7, при использовании обоих каналов UART данных, позволяет организовать обмен данными с другим RS-232 устройством в дуплексном режиме, т.е. по одному каналу выполнять отправку данных, а

по другому одновременно осуществлять прием данных.

Протокол **Modbus** в контроллерах CANNY7 реализуется как поверх UART, так и поверх RS-232. В качестве ADU (Application Data Unit) используется компактный двоичный вариант - Modbus RTU. Проверка целостности данных осуществляется с помощью автоматически рассчитываемой контрольной суммы (CRC). Размер пакета ограничен 16 байтами включая CRC.

*Примечание: Для корректной работы всех протоколов на базе UART/RS-232 необходимо, чтобы контакты GND устройств, совершающих обмен данными, были приведены к единому потенциалу ("общая земля").*

*Примечание: В реализации UART активным уровнем линии является потенциал GND 100mA, пассивным - положительный потенциал заданный внутренней или внешней подтяжкой канала контроллера. В реализации RS-232 — потенциалы обратные.*

Драйвер UART / RS232 / Modbus в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвер дискретного ввода-вывода. Таким образом, при активации драйвера UART / RS232 / Modbus, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера дискретного ввода-вывода будет проигнорировано контроллером.

### 1.5.5 Драйвер CAN

*Основная статья: **CANNY 7, Драйвер CAN***

Два специальных внешних вывода контроллера CANNY 7, расположенные на 4х контактом разъеме: CAN-H и CAN-L, предназначены для подключения к цифровой информационной **шине CAN**.

### 1.5.6 Драйвер LIN

*Основная статья: **CANNY 7, Драйвер LIN***

Два из одиннадцати каналов ввода-вывода CANNY 7, которые могут быть переданы под управление драйвера UART/RS-232 (Канал №9 и Канал №10), могут быть использованы для организации приема-передачи данных как два независимых канала драйвера **LIN**.

Каналы драйвера LIN могут подключаться как вместе так и по отдельности, иметь индивидуальные настройки скорости передачи данных, подтяжки линии и типа узла сети MASTER или SLAVE.

Драйвер LIN в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвер дискретного ввода-вывода. Таким образом, при активации драйвера LIN, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера дискретного ввода-вывода будет проигнорировано контроллером.

### 1.5.7 Драйвер I<sup>2</sup>C

*Основная статья:* **CANNY 7, Драйвер I<sup>2</sup>C**

В качестве линий связи (SDA и SCL) может быть назначена любая пара каналов контроллера CANNY 7. При этом, данные каналы должны быть подтянуты к напряжению 5В резисторами номиналом от 1 кОм до 10 кОм снаружи. Особенность реализации протокола I<sup>2</sup>C в контроллерах CANNY7 состоит в том, что CANNY7 может выступать только в качестве ведущего (Master) узла сети и обмен данными между устройствами, который может быть как одно- так и двунаправленным, происходит отдельными сеансами, с максимальной длиной сообщения I<sup>2</sup>C внутри одного сеанса равной 16 байтам, т. е. открытие одновременно несколько сеансов с разными устройствами не допускается. Скорость обмена фиксированная и составляет 100 кбит/с. Общее число ведомых устройств на линии может достигать нескольких десятков.

Драйвер I<sup>2</sup>C в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвер дискретного ввода-вывода. Таким образом, при активации драйвера I<sup>2</sup>C, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера дискретного ввода-вывода будет проигнорировано контроллером.

### 1.5.8 Драйвер Dallas 1-Wire

*Основная статья:* **Драйвер Dallas® 1-Wire®**

Контроллер CANNY7 может быть использован в качестве ведущего (MASTER) узла в однопроводной сети передачи данных **Dallas 1-Wire**.

Для подключения контроллера CANNY7 к шине 1-Wire может использоваться любой из его каналов ввода-вывода. При этом, данный канал должен быть снаружи подтянут к напряжению 5В резистором номиналом от 3 кОм до 7 кОм. В контроллерах CANNY7 предусмотрена возможность обращения к конкретному устройству на шине 1-Wire по его адресу, что позволяет организовать работу с контроллером нескольких ведомых устройств по одному каналу, в том числе, используя несколько каналов контроллера, возможно его последовательное подключение к нескольким шинам 1-Wire.



Драйвер Dallas 1-Wire в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвера ввода-вывода. Таким образом, при активации драйвера Dallas 1-Wire, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера ввода-вывода будет проигнорировано контроллером.

## 1.5.9 Параметры пользовательской конфигурации

*Основная статья: **CANNY 7, Параметры пользовательской конфигурации***

Параметры пользовательской конфигурации могут быть заданы конечным пользователем контроллера в момент загрузки в него программного обеспечения с использованием Исполняемого файла автономной загрузки ПО в контроллер. После загрузки ПО и запуска контроллера в автономном режиме, установленные пользователем таким образом данные, становятся доступны функциональной диаграмме в соответствующих регистрах контроллера.

Грамотное использование пользовательских параметров существенно повышает гибкость и универсальность решений на базе контроллера, позволяя конечному пользователю, не имеющему навыков работы с CannyLab, вносить безопасные изменения в работу алгоритма контроллера используя простой пользовательский интерфейс.

## 1.5.10 Энергонезависимая память (ЭНП)

*Основная статья: **CANNY 7, Энергонезависимая память (ЭНП)***

Для исключения потери критически важной информации (состояния контроллера, состояния внешних устройств и т. п.) при сбросе питания, в контроллере CANNY7 предусмотрено наличие энергонезависимой памяти. Сохраненные в ней значения будут доступны после восстановления питания контроллера в специальных регистрах.

Пользователю доступны 64 шестнадцатибитные ячейки энергонезависимой памяти, доступ к которым осуществляется с помощью соответствующих регистров чтения и записи.

*Примечание: Работа с энергонезависимой памятью не требует какой-либо специальной предварительной конфигурации.*

## 1.5.11 Драйвер пульта ИК ДУ

*Основная статья: **CANNY 7, Драйвер пульта ИК ДУ***

Контроллер CANNY7 позволяет принимать и передавать команды инфракрасных пультов дистанционного управления (ИК ДУ) в широко распространенных форматах NEC и extended NEC. Работа драйвера возможна в трех режимах: только прием, только передача или прием/передача. Для приема и передачи используются два любых канала контроллера.

При передаче команд ИК ДУ, используемый для этого канал контроллера CANNY7 генерирует только модулирующий сигнал. Для формирования пакетов импульсов контроллеру требуется наличие несущей частоты, источником которой может выступать как один из каналов ВЧ ШИМ CANNY7, так и внешний генератор ШИМ. Прием команд ИК ДУ требует наличия внешнего демодулятора, например TSOP1736 или аналогичного.

Драйвер ИК ДУ в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвер дискретного ввода-вывода. Таким образом, при активации драйвера ИК ДУ, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера дискретного ввода-вывода будет проигнорировано контроллером.

## 1.6 Технические требования

### 1.6.1 Электрические характеристики и требования к условиям эксплуатации

Напряжение питания	9...18 В
Потребляемый ток: в рабочем режиме (не более)	55 мА
в энергосберегающем режиме (не более)	5,5 мА
Максимальный ток каждого канала в режиме выхода	+120 мА / -120 мА
Сопротивление канала в режиме входа	4 кОм или 200 кОм
Диапазон рабочих температур	-40°C...+85°C
Степень защищенности от пыли и влаги	IP50
Защита электрических цепей:	

- от короткого замыкания канала — программная;
- от перегрузки канала — внутренними токоограничительными сгораемыми резисторами;
- от смены полярности источника питания — внутренним диодом;
- схема подавления высоковольтных выбросов при коммутации индукционной нагрузки каналов с №0 по №7: диод и варистор;
- схема подавления высоковольтных выбросов при коммутации индукционной нагрузки каналов с №8 по №10 — отсутствует.

## **1.6.2 Меры безопасности**

В цепях контроллера отсутствует опасное для жизни обслуживающего персонала напряжение. Открытые контакты контроллера при эксплуатации находятся под напряжением. Любые подключения к контроллеру и работы по его техническому обслуживанию производятся только при отключенном питании контроллера и подключенных к нему устройств.

## **1.6.3 Монтаж и подключение**

Монтаж и подключение контроллера должны производиться только квалифицированными специалистами, изучившими настоящую документацию. Монтаж контроллера должен производиться в местах соответствующих требованиям к условиям эксплуатации контроллера.

Не допускается попадание влаги на контакты выходных соединителей и внутренние элементы контроллера. Запрещается использование контроллера при наличии в атмосфере кислот, щелочей и иных агрессивных веществ.

Установка контроллера и прокладка кабелей подключаемых к нему должна производиться на расстоянии не менее 0.3 метра от высоковольтных силовых линий и источников сильных электромагнитных излучений — мощных реле, контакторов, газоразрядных ламп. Не допускается попадания влаги на корпус контроллера в месте его установки.

## **1.6.4 Транспортирование и хранение**

Контроллеры транспортируются всеми видами закрытого транспорта. Размещение и крепление транспортной тары с упакованными устройствами в транспортных средствах должны обеспечивать их устойчивое положение и не допускать перемещений во время транспортирования.

Транспортирование и хранение должны осуществляться при температуре окружающего воздуха от минус -40 до +85 °С и относительной влажности воздуха – до 80%, с соблюдением мер защиты от ударов и вибраций. В воздухе не должны присутствовать кислоты, щелочи и иные агрессивные вещества. Контроллеры следует хранить на стеллажах.

## 2 Общие сведения о ПЛК

### 2.1 Основные определения и используемые сокращения

ПО программное обеспечение.

ПК персональный компьютер.

ПЛК программируемый логический контроллер

CFD Canny Functional Diagram, графический язык программирования использующийся в интегрированной среде разработки CannyLab.

### 2.2 Что такое контроллер?

Контроллер - управляющее устройство, применяемое в промышленности, на транспорте, в других отраслях и в быту, для автоматического управления оборудованием по заданному алгоритму. Контроллеры широко используются для управления технологическими процессами, поддержания физических параметров объекта управления на заданном уровне и схожих по содержанию задач.

Одним из самых перспективных типов контроллеров, на сегодняшний день, является электронный программируемый логический контроллер (ПЛК), алгоритмы работы которого описываются программно, хранятся во внутренней памяти контроллера и выполняются встроенным в ПЛК микропроцессором. Взаимодействие электронного контроллера с объектом управления происходит посредством входящих и исходящих электрических сигналов.

Программируемые логические контроллеры, широко применяемые во встраиваемых системах, системах контроля и управления, имеют относительно простую для понимания архитектуру. Конструктивно это, как правило, довольно компактное устройство, состоящее из одного или нескольких соединенных между собой электронных модулей, содержащих разъемы для подачи на контроллер питания и подключения внешних входных и выходных электрических линий, позволяющих контроллеру взаимодействовать с внешним миром.

### 2.3 Программное обеспечение ПЛК

Современный ПЛК имеет развитый комплекс программных средств, состоящих из операционной системы ПЛК, предоставляемой производителем контроллера и сторонних

или собственных программных средств, предназначенных для разработки, отладки и записи в контроллер пользовательских программ.

Операционная система, отвечающая за выполнение контроллером пользовательского приложения, обслуживает низкоуровневую систему ввода-вывода контроллера, интерфейсы передачи данных, управляет распределением памяти, режимами энергопотребления, таймерами, осуществляет обработку ошибок, позволяя пользователю, разрабатывающему приложение, полностью сосредоточиться на алгоритмической части решения прикладной задачи.

Средства разработки и отладки пользовательских программ, позволяют создавать и корректировать программы, реализующие алгоритмы работы контроллера, моделировать на ПК процесс выполнения программы контроллером, наблюдать за промежуточными результатами вычислений, а так же записывать программное обеспечение в контроллер.

## 2.4 Как работает ПЛК

Перед применением ПЛК, в него необходимо загрузить операционную систему и пользовательскую программу, разработанную для решения конкретной прикладной задачи. Все программное обеспечение ПЛК обычно располагается в энергонезависимой памяти и защищено от повреждения в случае сброса питания. Изменение кода прикладной программы в памяти ПЛК может быть выполнено пользователем многократно.

Типовая схема работы ПЛК может быть описана следующим образом.

К внешним каналам ввода контроллера подключаются датчики, к каналам вывода - исполнительные механизмы. На контроллер подается питание и его операционная система немедленно начинает циклически исполнять пользовательское приложение.

Цикл выполнения приложения состоит из следующих, последовательных этапов:

- операционная система считывает состояние каждого входного канала контроллера и записывает его во внутреннюю память контроллера;
- операционная система, последовательно, команда за командой, выполняет всё пользовательское приложение: каждая исполняемая команда приложения считывает из внутренней памяти необходимые ей данные, производит с ними вычисления и записывает результаты своих расчетов во внутреннюю память контроллера;

- операционная система получает из внутренней памяти те значения, которые необходимо отобразить на выходе контроллера, и переводит выходные каналы в соответствующее состояние, после чего, весь цикл выполнения повторяется с начала.

Рассмотренный подход к архитектуре среды исполнения пользовательской программы, дает возможность реализовать логически параллельное исполнение контроллером нескольких задач в рамках одной пользовательской программы, позволяя автоматизировать управление несколькими одновременно происходящими процессами используя единственный ПЛК.

## **2.5 Программирование без программиста**

Одной из базовых идей, лежащих в основе использования ПЛК, является упрощение системы программирования и повышение наглядности языковых средств до уровня, доступного для понимания техническому специалисту хорошо знающему и непосредственно эксплуатирующему оборудование, но не обладающему специальными знаниями в области разработки программного обеспечения.

Такой специалист, получив простой и понятный инструмент выражения своих знаний об алгоритмах управления процессами, находящимися в его ведении, во многих случаях будет способен самостоятельно реализовать и отладить программу ПЛК, а при необходимости перенастроить параметры работы оборудования и своевременно изменить программу управления.

Зачастую качество программы ПЛК созданной таким специалистом оказывается выше, чем программы, написанной по его заданию профессиональным программистом не знакомым со всеми особенностями работы автоматизируемого процесса.

# 3 CANNY 7, Системные ресурсы и режимы работы

## 3.1 Общее описание

Системные ресурсы контроллера отображаются на группу регистров чтения и группу регистров записи. Обращаясь к данным регистрам из функциональной диаграммы, можно получить востребованные в практическом применении сведения о текущем состоянии контроллера и управлять режимами его работы. Список регистров системных ресурсов находится в разделе «Состояние контроллера» справочника регистров контроллера, который доступен пользователю CANNY Lab через контекстное меню входов и выходов типа «Адрес» функциональных блоков.

## 3.2 Сброс контроллера

Сброс контроллера происходит в результате любого из трех событий: при включении питания контроллера, при программном сбросе из функциональной диаграммы или по команде сторожевого таймера. При сбросе выполняется инициализация контроллера: все содержимое оперативной памяти очищается, каналы ввода-вывода переводятся в нейтральное состояние, драйверы системного программного обеспечения переводятся в исходное состояние, устанавливается режим нормального энергопотребления и выполнение функциональной диаграммы начинается с начала. Содержимое энергонезависимой памяти контроллера при сбросе не изменяется.

Информация о том, что произошел сброс доступна в регистре «Регистр контроля восстановления питания».

Регистр	Возвращаемые значения	
Регистр контроля восстановления питания	1	= текущий цикл выполнения диаграммы является первым с момента программного сброса или восстановления питания контроллера
	0	= текущий цикл выполнения диаграммы не является первым с момента сброса или восстановления питания

Принудительный сброс контроллера производится записью ненулевого значения в «Регистр сброса». В этом случае сброс контроллера происходит немедленно после окончания цикла выполнения функциональной диаграммы, в ходе которого произошла такая запись.

Регистр	Ожидаемые значения	
Регистр сброса	$\geq 1$	= запустить процедуру принудительного сброса контроллера
	0	= значение игнорируется

### 3.3 Встроенный светодиод контроллера

Контроллер имеет встроенный двухцветный (зеленый / красный) светодиод, управление включением которого каждым из цветов осуществляется из функциональной диаграммы путем записи определенных значений в соответствующие регистры.

Регистр	Ожидаемые значения	
Регистр включения зеленого светодиода	$\geq 1$	= включить встроенный зеленый светодиод контроллера
	0	= выключить встроенный зеленый светодиод контроллера
Регистр включения красного светодиода	$\geq 1$	= включить встроенный красный светодиод контроллера
	0	= выключить встроенный красный светодиод контроллера

*Примечание: Включение светодиода одновременно в обоих цветах не предусмотрено конструкцией контроллера, поэтому при задании режима одновременного включения красного и зеленого светодиода из функциональной диаграммы, светодиод включится зеленым цветом (приоритет зеленого).*

Фрагмент функциональной диаграммы, включающий встроенный красный светодиод контроллера на одну секунду после каждого сброса контроллера.



### 3.4 Режим пониженного энергопотребления

После сброса контроллер начинает работу в режиме нормального энергопотребления, функциональная диаграмма выполняется непрерывно. Переход в режим пониженного энергопотребления осуществляется по команде функциональной диаграммы, записью ненулевого значения в «Регистр установки режима пониженного энергопотребления». Переход в режим пониженного энергопотребления и происходит немедленно после окончания цикла выполнения функциональной диаграммы, в ходе которого была произведена такая запись, в отсутствие условий, препятствующих этому переходу.



Продолжительность фазы «сна» может быть задана из функциональной диаграммы путем записи значения в «Регистр масштаба времени режима пониженного энергопотребления».

По умолчанию, если на функциональной диаграмме отсутствует запись в соответствующий регистр, фаза «сна» режима пониженного энергопотребления длится 1000мс. Это означает, что находясь в режиме пониженного энергопотребления, в отсутствие условий перехода в режим нормального энергопотребления, контроллер делает паузу продолжительностью 1 секунду после каждого цикла выполнения функциональной диаграммы.

Регистр	Ожидаемые значения
Регистр режима пониженного энергопотребления	$\geq 1$ = перейти в режим пониженного энергопотребления 0 = вернуться в режим нормального энергопотребления
Регистр масштаба времени режима пониженного энергопотребления	0...65535 = продолжительность фазы «сна» в миллисекундах после каждого цикла выполнения функциональной диаграммы

Возврат контроллера в режим нормального энергопотребления происходит либо принудительно: немедленно после окончания цикла выполнения функциональной диаграммы, в ходе которого было записано значение «0» в «Регистр установки режима пониженного энергопотребления», либо автоматически в результате любого из следующих событий:

- при изменении электрического потенциала на любом контакте контроллера соответствующем каналу, сконфигурированному как активный вход или счетчик импульсов;
- при включенном из функциональной диаграммы любом из драйверов CAN, LIN, UART/RS232, ИК и другие при изменении электрического потенциала на соответствующем драйверу контакте контроллера.

*Примечание: Информация об изменениях электрического потенциала на контактах контроллера, соответствующих каналам, сконфигурированным как активный вход или счетчик импульсов, и изменениях электрического потенциала на контактах контроллера, соответствующих включенному драйверу CAN, LIN, UART/RS232, ИК и другие, доступна пользователю через специальный регистр состояния контроллера - "Регистр контроля активности интерфейсов контроллера".*

Информация о текущем режиме энергопотребления доступна в регистре «Регистр контроля режима энергопотребления».

Регистр	Возвращаемые значения	
Регистр контроля режима пониженного энергопотребления	1	= контроллер находится в режиме пониженного энергопотребления
	0	= контроллер находится в режиме нормального энергопотребления

*Примечание: При создании функциональных диаграмм, использующих режим пониженного энергопотребления, следует учитывать побочный эффект, привносимый изменением масштаба времени. Эффект выражается в том, что приращение счетчиков времени функциональных блоков: задержек включения, выключения и генераторов ШИМ в режиме пониженного энергопотребления происходит скачкообразно, в соответствии с временем фактически проведенным в фазе «сна» (по умолчанию с шагом 1000 мс).*

Фрагмент функциональной диаграммы, реализующий типовое управление режимом пониженного энергопотребления: переход в режим пониженного энергопотребления в отсутствие в течение 10 секунд условий препятствующих этому и автоматический возврат в нормальный режим при активности периферии контроллера или по установке запрета «засыпания» из диаграммы:



*Примечание: Обратите внимание на инверсию по выходу функционального блока №3.*

## 3.5 Сторожевой таймер

Для исключения вхождения контроллера в бесконечный цикл вне функциональной диаграммы, что может произойти в случае его неправильного подключения или неизвестных ошибок в системном программном обеспечении, предусмотрен сторожевой таймер - Watch Dog Timer (WDT).

В интегрированной среде разработки CannyLab, до версии 1.4, WDT был по умолчанию выключен, и его можно было включать и отключать из диаграммы.

Начиная с CannyLab версии 1.4, сторожевой таймер включен постоянно, можно лишь изменять его период, который по умолчанию равен 1 сек.

Если пауза в работе функциональной диаграммы превысит установленный период WDT, то произойдет автоматический сброс контроллера.

### 3.6 Фактическое время выполнения функциональной диаграммы

Время, требующееся контроллеру для выполнения функциональной диаграммы в реальных условиях эксплуатации зависит от числа и типов функциональных блоков присутствующих на диаграмме, числа задействованных драйверов входящих в состав системного программного обеспечения и их активности. На практике, цикл выполнения диаграммы CANNY 7 содержащей около 400 функциональных блоков и активно взаимодействующей с драйвером CAN продолжается приблизительно 9 мс.

*Примечание: При создании функциональной диаграммы, следует учитывать эффект привносимый продолжительностью её цикла. Эффект выражается в том, что приращение счетчиков времени функциональных блоков: задержек включения, выключения и генераторов ШИМ происходит скачкообразно. Так, при фактической длительности цикла в равной 6 мс, фактический период всех генераторов ШИМ на диаграмме будет кратен 6 мс.*

Информация о продолжительности предыдущего цикла выполнения функциональной диаграммы контроллера доступна в регистре «Регистр контроля длительности программного цикла».

Регистр	Возвращаемые значения
Регистр контроля длительности программного цикла, мс	0...65535 = продолжительность предыдущего полного цикла выполнения функциональной диаграммы в целых долях миллисекунд.

*Примечание: Наиболее точным способом измерения общего времени работы контроллера, например при реализации часов, является суммирование с накоплением значений получаемых из регистра «Регистр контроля длительности программного цикла» в ходе каждого цикла выполнения функциональной диаграммы.*

Фрагмент функциональной диаграммы, реализующий высокоточный счетчик секунд, пригодный для использования в часах реального времени:



## 3.7 Идентификатор устройства

С выходом обновленного системного загрузчика контроллеров CANNY7 версии 001004, при изготовлении устройств, каждому из них присваивается свой идентификационный номер, который можно использовать в дальнейшем при разработке пользовательских диаграмм для дополнительной их защиты от несанкционированного использования.

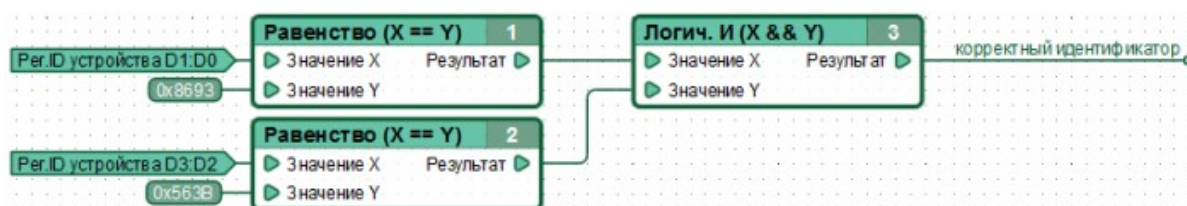
Доступ к работе с идентификатором устройства осуществляется через соответствующие специальные системные регистры контроллера.

Регистр	Возвращаемые значения
Регистр идентификатора устройства D1:D0	0...0xFFFF = значение двух младших байт (D1 и D0) индивидуального идентификационного номера контроллера;
Регистр идентификатора устройства D3:D2	0...0xFFFF = значение двух старших байт (D3 и D2) индивидуального идентификационного номера контроллера.

В процессе разработки пользовательской диаграммы, из CannyLab, идентификатор устройства можно узнать обратившись к информации об устройстве, доступной в пункте «Устройство» => «Информация» главного меню программы или по нажатию кнопки «Информация» панели инструментов, где он представлен в виде 4х байтового (32-битного) числа, с расположением старшего байта слева.

Например, идентификатор 0x563B8693 будет представлен так: регистр идентификатора устройства D1:D0 равен 0x8693, регистр идентификатора устройства D3:D2 равен 0x563B.

Пример функциональной диаграммы, иллюстрирующей работу с идентификатором устройства. В диаграмме значение, прочитанное из регистров идентификатора устройства, сравнивается с заданными и в случае их совпадения в именованную сеть «корректный идентификатор» сохраняется значение «1».



## 3.8 Контроль активности интерфейсов контроллера

"Регистр контроля активности интерфейсов контроллера" - синтетический регистр, отражающий текущую активность задействованных в пользовательской диаграмме внешних интерфейсов контроллера, либо включенных в режиме счетчика или в режиме активного входа каналов ввода-вывода. В те моменты времени, когда по задействованным пользователем интерфейсам контроллера CAN/LIN/UART/ИК и т.д. не осуществляется прием либо передача каких-либо сигналов и не происходит изменений электрического потенциала на соответствующих активным каналам-входам контактах контроллера, в "Регистре контроля активности интерфейсов контроллера" находится значение "0".

Использование данного регистра удобно в алгоритмах управления режимами энергопотребления контроллера.

Регистр	Возвращаемые значения
Регистр контроля активности интерфейсов контроллера	1...65535 = в течении предыдущего цикла выполнения диаграммы, на одном или нескольких задействованных в диаграмме интерфейсах или активных каналах ввода-вывода обнаружена активность
	0 = в течении предыдущего цикла выполнения диаграммы, ни на одном задействованном в диаграмме интерфейсе контроллера или активном канале ввода-вывода активности не обнаружено

## 3.9 Идентификатор вендора устройства

С выходом обновленного системного загрузчика контроллеров CANNY 7 версии 001005, при изготовлении устройств, каждому из них присваивается идентификационный номер их вендора (поставщика), который можно использовать в дальнейшем при разработке пользовательских диаграмм для дополнительной их защиты от несанкционированного использования.

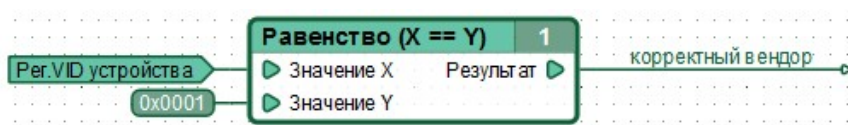
Идентификатор вендора устройства, назначаемый производителем контроллеров, является одинаковым для всех контроллеров, предназначенных для одного контрагента, или может быть установлен отдельно на конкретную партию контроллеров. Для его производства контроллеров с конкретным идентификатором вендора необходимо обратиться к производителю.

Доступ к работе с идентификатором устройства осуществляется через соответствующий специальный системный регистр контроллера.

Регистр	Возвращаемые значения
Регистр идентификатора вендора устройства	0...0xFFFF = значение индивидуального идентификационного номера вендора;

В процессе разработки пользовательской диаграммы, из CannyLab, идентификатор устройства можно узнать обратившись к информации об устройстве, доступной в пункте «Устройство» => «Информация» главного меню программы или по нажатию кнопки «Информация» панели инструментов, где он представлен в виде 2х байтового (16-битного) числа, с расположением старшего байта слева.

Пример функциональной диаграммы, иллюстрирующей работу с идентификатором вендора устройства. В диаграмме значение, прочитанное из регистра идентификатора вендора устройства, сравнивается с заданными и в случае их совпадения в именованную сеть «корректный вендор» сохраняется значение «1».



## 4 CANNY 7, Драйвер каналов ввода-вывода

### 4.1 Общее описание

Пользователям CANNY 7 доступны одиннадцать дискретных каналов ввода-вывода общего назначения. Каждый канал физически представлен соответствующим контактом разъема X2 либо разъема X3 контроллера. Записывая и считывая данные соответствующих регистров драйвера, функциональная диаграмма может как управлять электрическим потенциалом на каждом из этих контактов так и получать информацию о текущем значении потенциала каждого из них.

Физические характеристики каналов позволяют подключать к ним различные внешние исполнительные устройства — электромагнитные реле, небольшие электродвигатели, светодиоды. В качестве внешних источников дискретных сигналов способных управлять работой контроллера, возможно использовать механические, электромеханические и электронные кнопки и переключатели, генераторы импульсов, источники напряжения 0-12В и транзисторные выходы различной аппаратуры и т.п.

Режим и параметры работы любого из каналов задаются функциональной диаграммой. В каждый момент времени канал может работать только в одном из возможных режимов, однако допускается динамическое переопределение конфигурации канала из функциональной диаграммы в процессе ее выполнения.

Ряд драйверов контроллера CANNY 7, а именно CANNY 7, Драйвер высокочастотного широтно-импульсного модулятора (ВЧ ШИМ), CANNY 7, Драйвер UART - RS232 - Modbus, CANNY 7, Драйвер LIN, CANNY 7, Драйвер I2C, CANNY 7, Драйвер Dallas 1-Wire и CANNY 7, Драйвер пульта ИК ДУ, в своей работе используют ресурсы драйвера каналов ввода-вывода и, при этом, имеют более высокий приоритет. Таким образом, при использовании указанными драйверами тех или иных каналов контроллера, доступ драйвера ввода-вывода к этим каналам невозможен.

### 4.2 Регистры драйвера

Параметры, определяющие режим работы и текущее состояние каналов контроллера, задаются для каждого канала независимо друг от друга. Ниже приведено описание допустимых значений регистров управления работой каналов ввода-вывода во всех основных режимах.

Регистр	Ожидаемые значения
---------	--------------------

Регистр установки конфигурации канала №0 Регистр установки конфигурации канала №1 Регистр установки конфигурации канала №2 ... Регистр установки конфигурации канала №10	1...65535 = установить конфигурацию канала контроллера, определяющую текущий режим и параметры его работы (задается специальной константой из справочника констант);  0 = перевести в нейтральное положение соответствующий каналу контакт контроллера и исключить возможность изменения его состояния из функциональной диаграммы.
Регистр установки периода канала №0 Регистр установки периода канала №1 Регистр установки периода канала №2 ... Регистр установки периода канала №10	<p>В режиме входа-счетчика:</p> <p>1...65535 = период в миллисекундах подсчета числа импульсов на соответствующем контакте контроллера и обновления полученного значения регистре контроля выходного значения канала;</p> <p>0 = включить канал в режиме счетчика с накоплением.</p> <p>В режиме широтно-импульсного выхода:</p> <p>1...65535 = период в миллисекундах генерируемых импульсов на соответствующем контакте контроллера;</p> <p>0 = прекратить генерацию импульсов и установить потенциал на соответствующем контакте контроллера равным потенциалу состояния «ВЫКЛ» текущей конфигурации канала.</p> <p>В режиме дискретного входа:</p> <p>0...65535 = значение игнорируется;</p> <p>В режиме дискретного выхода:</p> <p>0...65535 = значение игнорируется;</p>
Регистр установки выходного значения канала №0 Регистр установки выходного значения канала №1 Регистр установки выходного значения канала №2 ...	<p>В режиме входа-счетчика:</p> <p>0...65535 = значение игнорируется.</p> <p>В режиме широтно-импульсного выхода:</p> <p>0...65535 = заполнение (скважность) в миллисекундах генерируемых импульсов на соответствующем контакте контроллера.</p> <p>В режиме дискретного входа:</p> <p>0...65535 = значение игнорируется;</p> <p>В режиме дискретного выхода:</p> <p>1...65535 =</p>



Регистр установки выходного значения канала №10	0	установить на соответствующем контакте контроллера электрический потенциал заданный конфигурацией данного канала для состояния «ВКЛ»;  = установить на соответствующем контакте контроллера электрический потенциал заданный конфигурацией данного канала для состояния «ВЫКЛ».
---	---	---

Регистры контроля драйвера каналов ввода-вывода разделяются на содержащие информацию о состоянии драйвера ввода-вывода в целом и на содержащие информацию о состоянии каждого канала индивидуально. Ниже приведено описание возвращаемых значений регистров контроля драйвера каналов ввода-вывода во всех основных режимах работы.

Регистр	Возвращаемые значения	
Регистр контроля активности ввода-вывода	1	= в ходе прошедшего цикла выполнения функциональной диаграммы зарегистрировано изменение электрического потенциала на каком-либо контакте контроллера соответствующем каналу, сконфигурированному как активный вход или счетчик импульсов; либо диаграммой было изменено значение регистров выходных каналов или входов-счетчиков;
	0	= за прошедший цикл выполнения функциональной диаграммы изменений на активных входах контроллера или изменений значений регистров драйвера не обнаружено.
Регистр контроля перегрузки ввода-вывода	1	= на одном или нескольких каналах контроллера включенных как силовой выход, обнаружено короткое замыкание и канал переведен в режим защиты;
	0	= ни на одном из каналов контроллера не обнаружено короткого замыкания.
Регистр контроля входного значения канала №0	1	В режиме дискретного входа: = на соответствующем контакте контроллера установился электрический потенциал соответствующий полярности данного входа;
Регистр контроля входного значения канала №1	0	= на соответствующем контакте контроллера установился электрический потенциал не соответствующий полярности данного входа.
Регистр контроля		

входного значения канала №2		В режиме входа-счетчика:
...		0...65535 = число импульсов зарегистрированных на соответствующем контакте контроллера за прошедший период времени заданной продолжительности.
Регистр контроля входного значения канала №10		В режиме дискретного или широтно-импульсного выхода:
	1	= на соответствующем контакте контроллера обнаружен электрический потенциал соответствующий полярности данного выхода в режиме «ВКЛ»;
	0	= на соответствующем контакте контроллера обнаружен электрический потенциал не соответствующий полярности данного выхода в режиме «ВКЛ».

## 4.3 Нейтральное состояние канала

Каналы автоматически переводятся в нейтральное состояние в следующих случаях:

- при работе контроллера в режиме загрузки программного обеспечения;
- при работе контроллера в автономном режиме, если конфигурация канала не задана, задана константой «Нейтральное состояние» или значением «0»;
- в момент программного сброса или сброса питания контроллера, до перехода в автономный режим и начала исполнения функциональной диаграммы;
- в случае аварии контроллера или отказа системного программного обеспечения.

Электрически, нейтральное положение канала («воздух») эквивалентно высокоомному входу, внутренне соединенному с контактом GND контроллера резистором номинала ~200 кОм.

## 4.4 Режим дискретного выхода

Канал, сконфигурированный для работы в режиме дискретного выхода, устанавливает на соответствующем контакте контроллера электрический потенциал соответствующий состоянию «ВКЛ» при записи ненулевого значения в регистр выходного значения канала, и устанавливает на соответствующем контакте контроллера электрический потенциал соответствующий состоянию «ВЫКЛ» при записи значения «0» в регистр выходного значения канала.

Конфигурация канала для работы в данном режиме, определяется константой, задающей комбинацию параметров, определяющих электрический потенциал и силу тока

на контакте соответствующего канала контроллера в положениях «ВКЛ» и «ВЫКЛ».

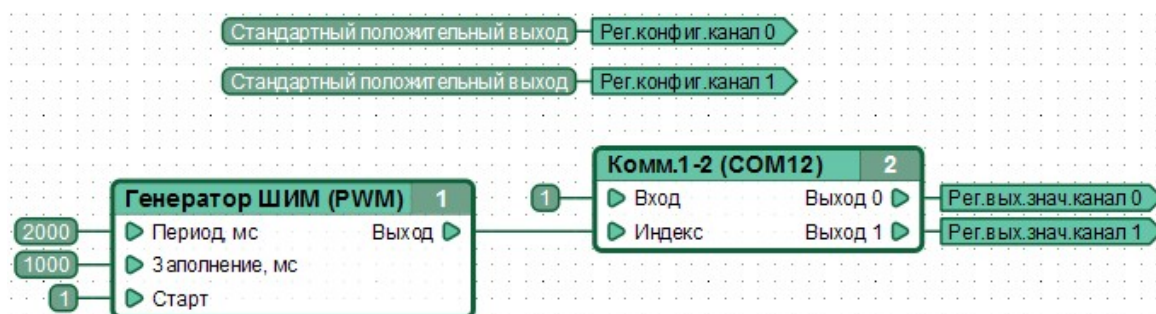
Параметр	Перечень допустимых значений
Тип канала	Дискретный выход.
Состояние «ВКЛ»	плюс (1мА); ПЛЮС (100мА); минус (1мА); МИНУС (100мА);воздух.
Состояние «ВЫКЛ»	плюс (1мА); ПЛЮС (100мА); минус (1мА); МИНУС (100мА);воздух.

Именованные константы, представляющие доступные пользователю комбинации параметров конфигурации каналов, содержатся в разделе «Конфигурация каналов ввода-вывода» справочника констант CannyLab, доступ к которому осуществляется через контекстное меню входа функционального блока, имеющего тип «Константа».

Для перевода канала контроллера в режим дискретного выхода, необходимо в соответствующий каналу «Регистр конфигурации канала №XX» поместить значение константы, соответствующей выбранному режиму работы.

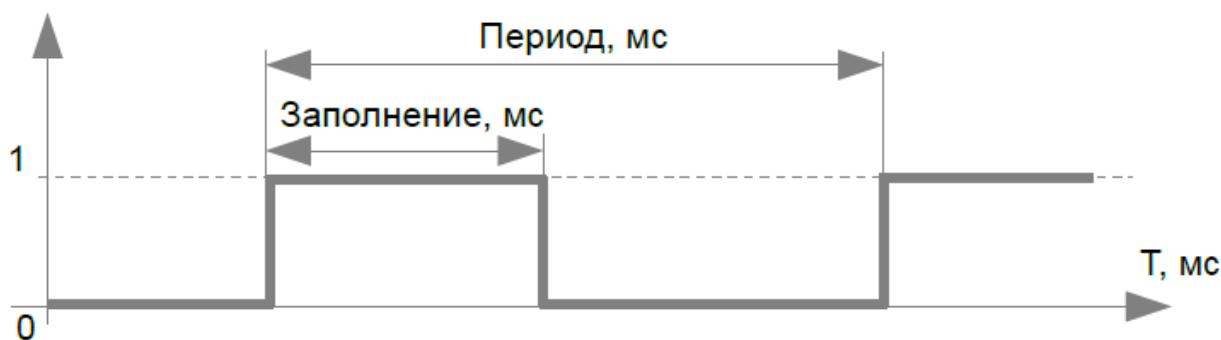
*Примечание: Для канала работающего в конфигурации дискретного выхода так же возможно получить значение, соответствующее фактическому текущему электрическому потенциалу на контакте данного канала, что позволяет использовать канал в режиме обратной связи.*

Пример функциональной диаграммы выполняющей ежесекундное поочередное переключение электрических потенциалов с +12В(100 мА) на GND(1 мА) на контактах контроллера, соответствующих каналам №0 и №1.



## 4.5 Режим широтно-импульсного выхода

Канал сконфигурированный для работы в режиме широтно-импульсного выхода, генерирует на соответствующем контакте контроллера широтно-импульсный сигнал заданной полярности, частоты и заполнения импульсов. В данном режиме канал работает асинхронно функциональной диаграмме, что позволяет добиться большей стабильности временных параметров генерируемого сигнала, чем при организации широтно-импульсного генератора средствами функциональной диаграммы.



Конфигурация канала для работы в данном режиме, определяется константой, представляющей комбинацию параметров, определяющих электрический потенциал и силу тока на контакте, в активной «ВКЛ» и пассивной «ВЫКЛ» фазе генерации соответствующего канала контроллера, и парой числовых значений, определяющих частоту и заполнение генерируемых импульсов в миллисекундах.

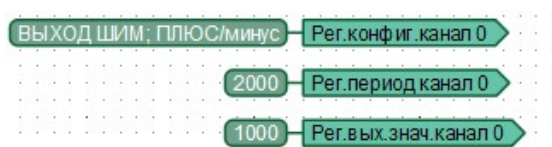
Параметр	Перечень допустимых значений
Тип канала	Широтно-импульсный выход.
Состояние «ВКЛ»	плюс (1мА); ПЛЮС (100мА); минус (1мА); МИНУС (100мА);воздух.
Состояние «ВЫКЛ»	плюс (1мА); ПЛЮС (100мА); минус (1мА); МИНУС (100мА);воздух.

Именованные константы, представляющие доступные пользователю комбинации параметров конфигурации каналов, содержатся в разделе «Конфигурация каналов ввода-вывода» справочника констант CannyLab, доступ к которому осуществляется через контекстное меню входа функционального блока, имеющего тип «Константа».

Для перевода канала контроллера в режим широтно-импульсного выхода, необходимо:

- в соответствующий каналу «Регистр конфигурации канала №XX» поместить значение константы, соответствующей выбранному режиму работы;
- в соответствующий каналу «Регистр периода канала №XX» поместить числовое значение от 0 до 65535, устанавливающее период генерируемых импульсов в мс;
- в соответствующий каналу «Регистр выходного значения канала №XX» поместить числовое значение от 0 до 65535, устанавливающее заполнение генерируемых импульсов в мс.

Функциональная диаграмма выполняющая ежесекундное переключение электрического потенциала с +12В(100 мА) на GND(1 мА) на контакте контроллера соответствующем каналу №0:



## 4.6 Режим дискретного входа

Канал, сконфигурированный для работы в режиме дискретного входа, возвращает значение «1» в регистре своего входного значения, если на соответствующем контакте контроллера установился электрический потенциал соответствующий состоянию «Полярность входа»; и возвращает значение «0» в регистре входного значения, когда на соответствующем контакте контроллера установился электрический потенциал не соответствующий состоянию «Полярность входа».

Конфигурация канала для работы в данном режиме, определяется константой, представляющей комбинацию параметров, определяющих электрический потенциал на контакте принимаемый за состояние «1» соответствующего канала контроллера, наличие и потенциал внутренней «подтяжки» контакта контроллера, чувствительность и активность канала в режиме пониженного энергопотребления контроллера.

Параметр	Перечень допустимых значений
Тип канала	Дискретный вход.
Полярность входа	ПЛЮС; МИНУС.
Подтяжка	плюс; минус; воздух.
Режим ожидания	активный; пассивный.
Чувствительность	максимум (0 мс); высокая (20 мс); норма (200 мс); низкая (700 мс).

Изменение потенциала на контакте канала, находящегося в режиме активного ожидания, приведет к немедленному автоматическому выходу контроллера из режима пониженного энергопотребления. Изменение потенциала на контакте канала, находящегося в режиме пассивного ожидания, не повлияет на режим энергопотребления контроллера.

Чувствительность канала определяет его «защиту от дребезга», т.е. задает временной интервал, в течение которого электрический потенциал на соответствующем каналу контакте контроллера должен оставаться неизменным, для того чтобы считаться установившимся и изменить состояние регистра входного значения канала.

Именованные константы, представляющие доступные пользователю комбинации параметров конфигурации каналов, содержатся в разделе «Конфигурация каналов ввода-вывода» справочника констант CannyLab, доступ к которому осуществляется через контекстное меню входа функционального блока, имеющего тип «Константа».

Для перевода канала контроллера в режим дискретного входа, необходимо в соответствующий каналу «Регистр конфигурации канала №XX» поместить значение константы, соответствующей выбранному режиму работы.

Пример функциональной диаграммы, включающей встроенный зеленый светодиод контроллера при поступлении и удержании в течение не менее 200мс на соответствующем каналу №0 контакте контроллера, потенциала GND:



## 4.7 Режим счетчика

Канал сконфигурированный для работы в режиме счетчика, возвращает в регистре своего выходного значения число, соответствующее количеству переключений электрического потенциала на соответствующем контакте контроллера из состояния противоположного параметру «Полярность входа» в состояние соответствующее параметру «Полярность входа» за прошедший период времени заданной продолжительности. Таким образом, канал в данном режиме выполняет функцию счетчика передних фронтов сигнала или частотомера.

В данном режиме канал работает асинхронно функциональной диаграмме, что позволяет добиться большей точности определения временных параметров исследуемого сигнала и измерять сигнал большей частоты, чем при организации счетчика импульсов средствами функциональной диаграммы.

Конфигурация канала для работы в данном режиме, определяется константой, представляющей комбинацию параметров, определяющих электрический потенциал, при появлении которого на контакте соответствующего канала контроллера регистрируется передний фронт сигнала, наличие и потенциал внутренней «подтяжки» контакта контроллера.

Конфигурация канала для работы в данном режиме задается комбинацией следующих параметров:

Параметр	Перечень допустимых значений
Тип канала	Вход-счетчик.
Полярность входа	ПЛЮС; МИНУС.
Подтяжка	плюс; минус; воздух.

В режиме счетчика, канал всегда находится в активном ожидании с максимальной чувствительностью. Подсчет драйвером числа изменений состояния такого канала

ведется асинхронно.

Именованные константы, представляющие доступные пользователю комбинации параметров конфигурации каналов, содержатся в разделе «Конфигурация каналов ввода-вывода» справочника констант CannyLab, доступ к которому осуществляется через контекстное меню входа функционального блока, имеющего тип «Константа».

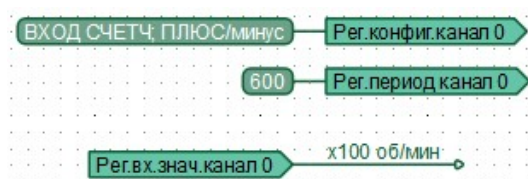
Для перевода канала контроллера в режим входа-счетчика, необходимо:

- в соответствующий каналу «Регистр конфигурации канала №XX» передать значение константы, соответствующей выбранному режиму работы;
- в соответствующий каналу «Регистр периода канала №XX» передать числовое значение от 0 до 65535, устанавливающее период подсчета импульсов в мс;

При ненулевом значении в регистре периода, значение в регистре входного значения канала обновляется один раз в период и содержит число импульсов зарегистрированное счетчиком за прошедший период.

Подсчет счетчиком импульсов ведется с переполнением. Это означает, что при достижении значения 65535 и последующем увеличении на единицу, значение счетчика устанавливается равным нулю, но каждый последующий импульс вновь увеличивает значение счетчика на единицу.

Пример функциональной диаграммы тахометра с разрешающей способностью 100 оборотов в минуту, подсчитывающем число импульсов на соответствующем канале №0 контакте контроллера:

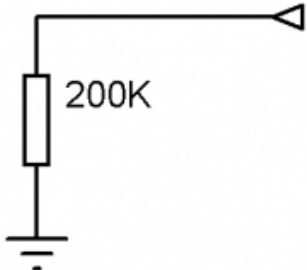
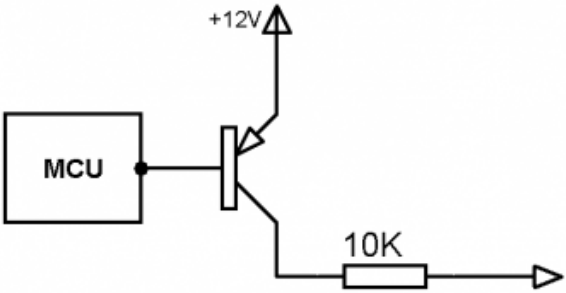
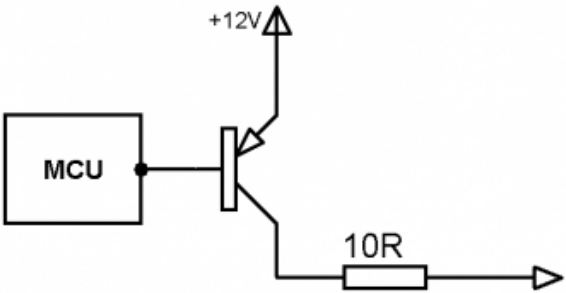
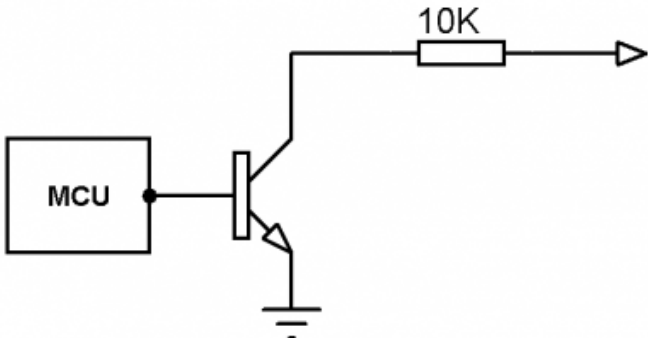
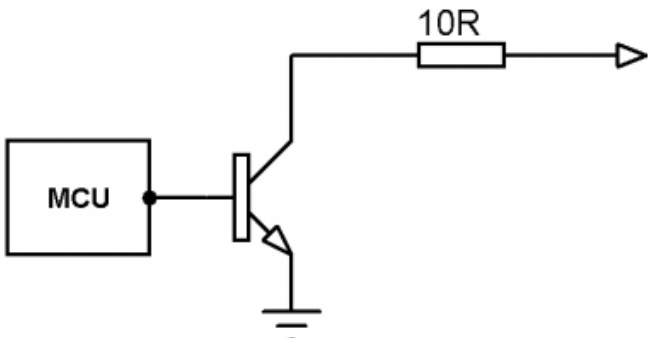


*Примечание: Если период установлен равным нулю, то счетчик непрерывно ведет подсчет импульсов с накоплением результата в регистре выходного значения канала. В данном режиме значение, этого регистра сбрасывается лишь в результате переполнения регистра.*

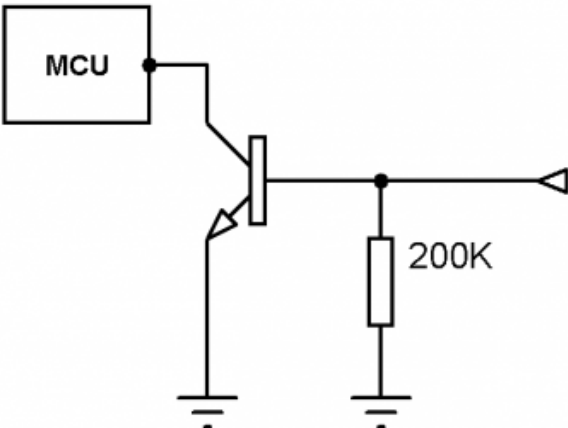
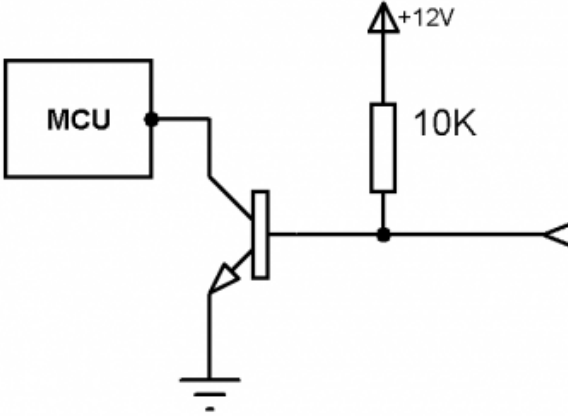
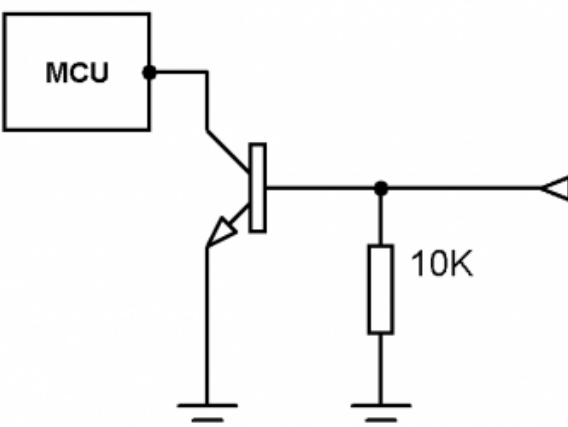
## 4.8 Эквивалентные принципиальные электрические схемы

Эквивалентные электрические принципиальные схемы для каждого возможного состояния канала ввода-вывода контроллера приведены в таблице:

Состояние канала	Эквивалентная электрическая принципиальная схема
------------------	--

«ВОЗДУХ»	
«ПЛЮС (1mA)»	
«ПЛЮС (100mA)»	
«МИНУС (1mA)»	
«МИНУС (100mA)»	
«ПОДТЯЖКА ВОЗДУХ»	



	
«ПОДТЯЖКА ПЛЮС»	
«ПОДТЯЖКА МИНУС»	

## 4.9 Электрическая защита

Для каналов работающих в конфигурациях дискретного выхода и широтно-импульсного выхода, чей ток в состоянии «ВКЛ» или «ВЫКЛ» существенно превысит значение 100мА, системное программное обеспечение контроллера реализует автоматическую защиту от короткого замыкания. Защита реализована в виде переключения замкнутого канала в нейтральный режим, последующего периодического импульсного тестового включения и автоматического возврата к заданной пользователем конфигурации при обнаружении устранения замыкания. Данная защита работает в полностью автоматическом режиме и не требует конфигурации пользователем.



# 5 CANNY 7, Драйвер высокочастотного широтно-импульсного модулятора (ВЧ ШИМ)

## 5.1 Общее описание

Два из одиннадцати каналов ввода-вывода (Канал №1 и Канал №2) CANNY 7 поддерживают работу в режиме высокочастотного широтно-импульсного модулятора. Каналы могут быть задействованы независимо друг от друга и иметь независимые настройки скважности сигнала и подтяжки линии, однако, период высокочастотного ШИМ является параметром, общим для обоих каналов. В режиме ВЧ ШИМ, временные параметры ШИМ – период и скважность задаются в диапазоне от 2 до 20000 микросекунд, с шагом 1 микросекунда.

В режиме ВЧ ШИМ канал имеет фиксированную полярность импульсов — GND 100мА. Генерация может вестись как в режиме открытого коллектора – подтяжка линии отсутствует или внешняя, так и в режиме с внутренней подтяжкой к +12В (задается установкой значения в соответствующем регистре). В данном режиме канал работает асинхронно функциональной диаграмме, что позволяет добиться максимальной стабильности временных параметров генерируемого сигнала.

Драйвер высокочастотного широтно-импульсного модулятора (ВЧ ШИМ) в своей работе использует ресурсы Драйвера каналов ввода-вывода имея, при этом, более высокий приоритет. Таким образом, при активации драйвера ВЧ ШИМ, используемые им каналы контроллера становятся недоступны для драйвера ввода-вывода.

*Примечание: В режиме высокочастотного широтно-импульсного модулятора электрическая защита канала от короткого замыкания находится в отключенном состоянии! Перегрузка или короткое замыкание каналов контроллера находящихся в режиме ВЧ ШИМ может привести к выходу контроллера из строя!*

## 5.2 Регистры драйвера

Ниже приведено описание допустимых значений регистров управления работой драйвера высокочастотного широтно-импульсного модулятора.

Регистр	Ожидаемые значения
Регистр периода ВЧ ШИМ, мкс	1...20000 = задать период в микросекундах генератора ВЧ ШИМ обоих каналов. Значение превышающее 20000 будет принято как 20000.



# 6 CANNY 7, Драйвер UART - RS232 - Modbus

## 6.1 Общее описание

Два из одиннадцати каналов ввода-вывода (Канал №9 и Канал №10) CANNY 7 поддерживают работу в режиме приема/передачи данных последовательных протоколов UART, RS-232 и могут быть использованы для связи контроллеров друг с другом или с внешним оборудованием поддерживающим данные протоколы связи. Каналы могут быть задействованы независимо друг от друга и иметь индивидуальные настройки скорости передачи данных, типа и конфигурации используемого протокола, подтяжки линии.

Реализация **UART** в контроллерах CANNY7 позволяет организовать последовательный прием и передачу данных по одному проводу в полудуплексном режиме. Таким образом CANNY7 может иметь 2 независимых подключения с использованием протокола UART. Контроль состояния канала передачи данных должен осуществляться пользователем из функциональной диаграммы. Если канал свободен, то устройство может начать передачу данных, в противном случае устройство должно дожидаться освобождения линии.

Реализация протокола **RS-232** в контроллерах CANNY7, при использовании обоих каналов UART данных, позволяет организовать обмен данными с другим RS-232 устройством в дуплексном режиме, т.е. по одному каналу выполнять отправку данных, а по другому одновременно осуществлять прием данных.

Протокол **Modbus** в контроллерах CANNY7 реализуется как поверх UART, так и поверх RS-232. В качестве ADU (Application Data Unit) используется компактный двоичный вариант - Modbus RTU. Проверка целостности данных осуществляется с помощью автоматически рассчитываемой контрольной суммы (CRC). Размер пакета ограничен 32 байтами включая CRC.

*Примечание: Для корректной работы всех протоколов на базе UART/RS-232 необходимо, чтобы контакты GND устройств, совершающих обмен данными, были приведены к единому потенциалу ("общая земля").*

*Примечание: В реализации UART активным уровнем линии является потенциал GND 100mA, пассивным - положительный потенциал заданный внутренней или внешней подтяжкой канала контроллера. В реализации RS-232 — потенциалы обратные.*

Драйвер UART / RS232 / Modbus в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвер дискретного ввода-вывода. Таким образом, при активации драйвера UART / RS232 / Modbus, для задействованных в его работе каналов, изменение значений в связанных с ними

регистрах драйвера дискретного ввода-вывода будет проигнорировано контроллером.

## 6.2 Регистры драйвера

Ниже приведено описание допустимых и возвращаемых значений регистров управления работой драйвера.

Регистры конфигурации драйвера UART / RS-232 / Modbus.

Регистр	Ожидаемые значения
Регистр конфигурации UARTx	1...N = установить конфигурацию канала драйвера UART контроллера, определяющую текущий режим и параметры его работы (задается специальной константой из справочника констант); 0 = отключить канал от драйвера UART, вернуть управление каналом драйверу каналов ввода-вывода и разрешить изменения его состояния из функциональной диаграммы.
Регистр установки таймаута приема сообщения UARTx, бит	1...65535 = прекращение приема данных, если в течении времени, за которое может быть принято указанное число бит данных, на линии не было зафиксировано ни одного изменения потенциала и линия находится в пассивном состоянии; 0 = использовать значение по умолчанию, задаваемое в конфигурации канала (13).

Конфигурация драйвера UART определяется константой, представляющей комбинацию параметров, определяющих скорость, режим, дополнительные параметры передачи данных и потенциал линии в пассивном режиме.

Параметр	Перечень допустимых значений
Скорость передачи данных, бод	110; 150; 300; 600; 1200; 1800; 2400; 4800; 9600; 19200; 38400; 57600
Режим работы	UART; RS-232
Подтяжка в режиме UART	плюс; воздух
Направление передачи в режиме RS-232	прием; передача
Количество бит данных	8; 9
Контроль четности	N (no) — нет; O (odd) — нечетный; E (even) - четный
Количество стоповых бит	1; 2

Именованные константы, представляющие доступные пользователю комбинации параметров конфигурации UART, содержатся в разделе «Конфигурация UART / RS-232» справочника констант CannyLab, доступ к которому осуществляется через контекстное меню констант на функциональной диаграмме.

Регистры диагностики драйвера UART / RS-232 / Modbus.

Регистр	Возвращаемые значения	
Регистр переполнения буфера UARTx	1	= буфер UART переполнен;
	0	= переполнения не зафиксировано.
Регистр ошибки приема UARTx	1	= во время приема данных UART произошла ошибка;
	0	= драйвер работает в нормальном режиме.
Регистр готовности буфера передачи данных UARTx	1	= буфер передачи данных драйвера UART свободен;
	0	= буфер передачи данных драйвера UART занят, передача данных невозможна.

Регистры приема драйвера UART / RS-232 / Modbus.

Регистр	Возвращаемые значения	
Регистр наличия принятых данных UARTx	1	= сообщение получено и помещено в буфер приема соответствующего канала драйвера UART;
	0	= в буфере приема соответствующего канала драйвера UART отсутствуют актуальные данные.
Регистр признака RTU буфера приема данных UARTx	1	= полученное по соответствующему каналу сообщение UART является корректным сообщением Modbus RTU, контрольная сумма корректна;
	0	= полученное по соответствующему каналу сообщение UART не является корректным сообщением Modbus RTU.
Регистр отсутствия активности драйвера UARTx	1	= активность соответствующего канала драйвера UART отсутствует, линия находится в пассивном режиме;
	0	= зафиксирована активность на линии соответствующего канала драйвера UART.
Регистр длины принятого сообщения UARTx	0...32	= значение, равное количеству байт данных, в принятом по соответствующему каналу драйвера

	UART пакете данных.
Регистр принятого сообщения UARTx D1:D0 ... Регистр принятого сообщения UARTx D31:D30	0...0xFFFF = значения соответствующих байт данных приемных буферов UART каждого канала, по два байта на регистр.

Регистры передачи драйвера UART / RS-232 / Modbus.

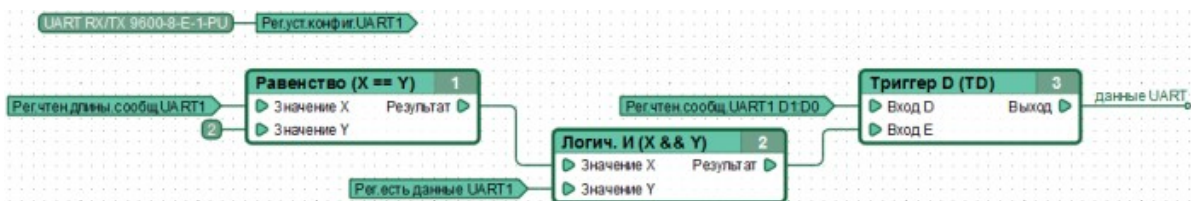
Регистр	Ожидаемые значения	
Регистр начала передачи UARTx	≥ 1	= загрузить данные из регистров передачи в буфер передачи соответствующего канала драйвера UART;
	0	= не загружать данные в буфер передачи соответствующего канала драйвера UART.
Регистр признака RTU буфера передачи данных UARTx	≥ 1	= команда драйверу дописать к сообщению в буфере передачи соответствующего канала драйвера UART контрольную сумму в формате Modbus RTU, сформировав для отправки пакет данных в соответствии с протоколом Modbus RTU;
	0	= передавать данные буфера передачи соответствующего канала драйвера «как есть».
Регистр длины сообщения передачи UARTx	0...32	= количество байт данных, которое будет необходимо передать в линию, при получении команды на отправку соответствующего канала драйвера UART.
Регистр сообщения передачи UARTx D1:D0 ... Регистр сообщения передачи UARTx D31:D30	0...0xFFFF = значения соответствующих байт данных для передачи по соответствующему каналу драйвера UART, по два байта на регистр.	

## 6.3 Работа контроллера в режиме UART

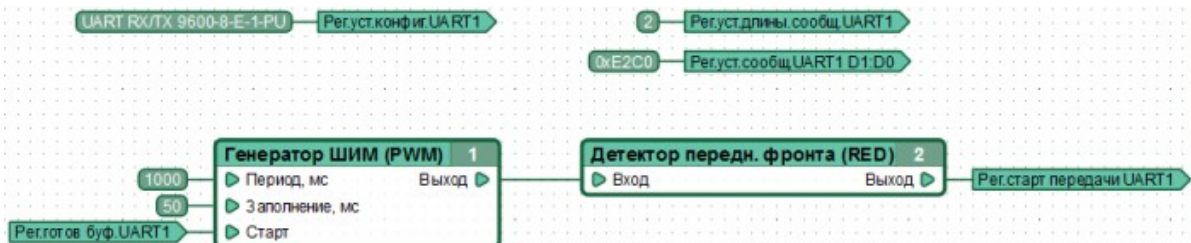
Работая в режиме UART контроллер может осуществлять полудуплексный прием/передачу данных по одному проводу.

Пример функциональной диаграммы для получения данных по UART.





Пример функциональной диаграммы для передачи данных по UART.

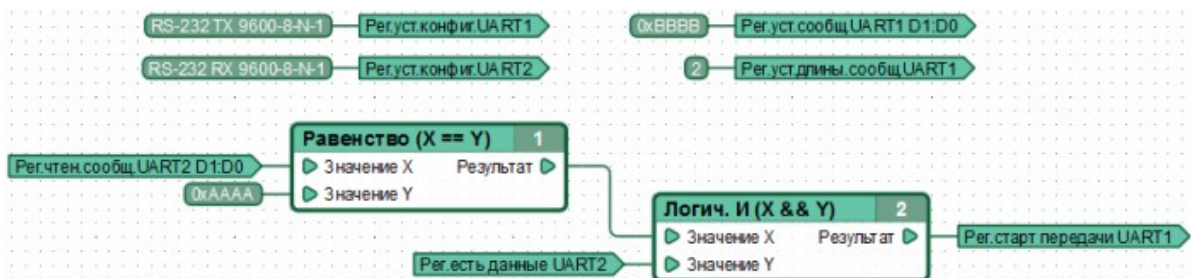


*Примечание: Особое внимание следует обратить на то, что для избежания коллизий, при отправке данных, необходимо строго контролировать регистр готовности буфера передачи данных канала UART: если буфер не готов, значит в данный момент драйвером выполняется прием данных — отправка данных должна быть отложена.*

## 6.4 Работа контроллера в режиме RS-232

Работая по протоколу RS-232 контроллер может использовать каждый из своих каналов передачи данных только в симплексном (однаправленном) режиме. При использовании сразу обоих каналов UART, которые работают независимо друг от друга, возможно организовать дуплексный режим обмена информацией по двум проводам: один канал — только прием, второй — только передача.

Пример функциональной диаграммы для работы с RS-232. Получая данные по каналу UART2, при условии, что значение полученных байтов D1:D0 равно «0xAAAA», контроллер отправляет, по каналу UART1, 2 байта данных, содержащих значение «0xBBBB».

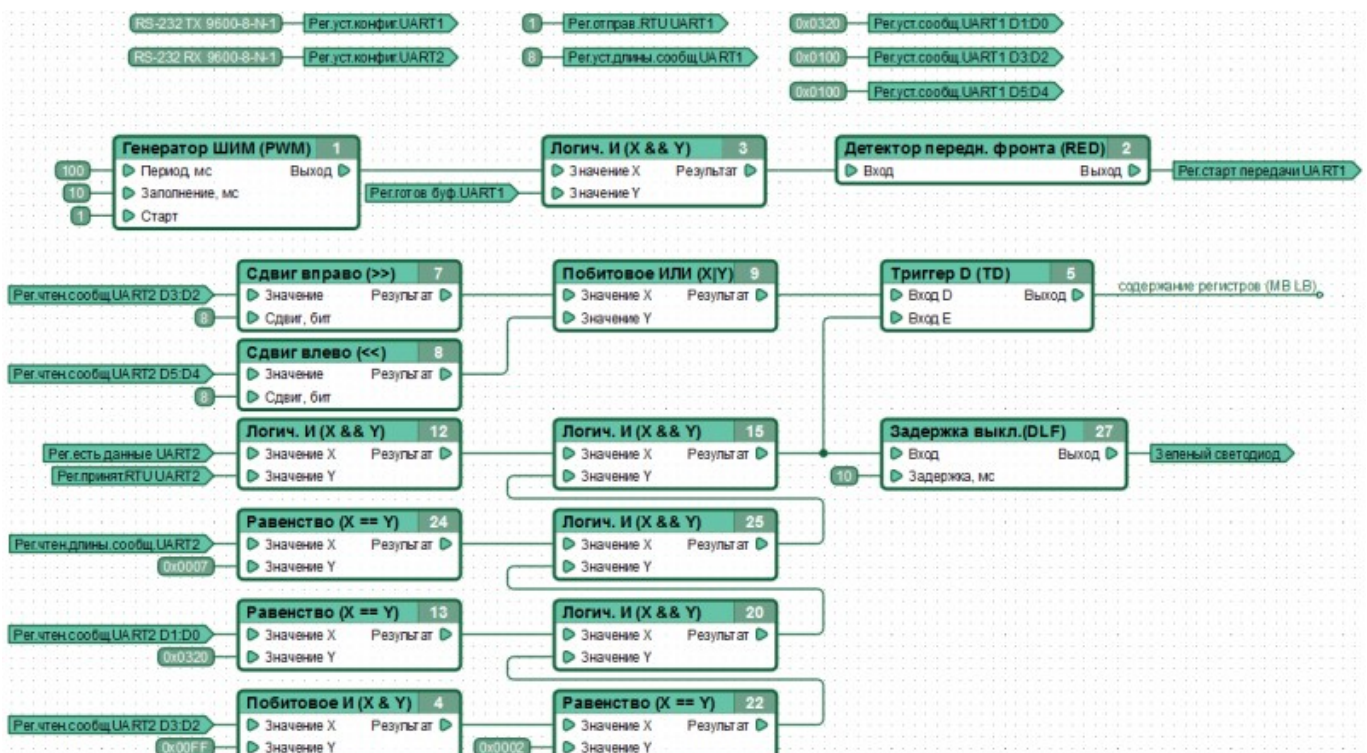


## 6.5 Реализация Modbus RTU

Драйвер UART/RS-232 включает в себя функционал автоматического формирования и проверки контрольной суммы по стандарту Modbus RTU, что упрощает включение контроллера в сеть работающую по данному протоколу. Управлять формированием контрольной суммы передаваемых сообщений и её проверкой при приёме можно через соответствующие регистры драйвера.

*Примечание: При работе в режиме Modbus регистр чтения длины UARTx содержит значение с учетом принятых байт контрольной суммы (CRC), т. е. на 2 байта больше чем длина полезной нагрузки сообщения. При отправке пакета Modbus значение регистра установки длины сообщения UARTx также должно быть увеличено на 2 байта для возможности размещения и пересылки CRC.*

Пример функциональной диаграммы работы контроллера в режиме MASTER-узла Modbus поверх RS-232. Выполняя диаграмму, контроллер периодически отправляет SLAVE-узлу с адресом 0x20 запрос на получение от него значения из Modbus-регистра данных с адресом 0x0001. Получение ответа на свой запрос MASTER сопровождает коротким включением своего зеленого светодиода.



Строка запроса, при обмене данными между устройствами, будет выглядеть так:

-> 20 03 00 01 00 01 D3 7B

Адрес опрашиваемого устройства: 0x20 (байт b0 регистра приема сообщения UART2 D1:D0).

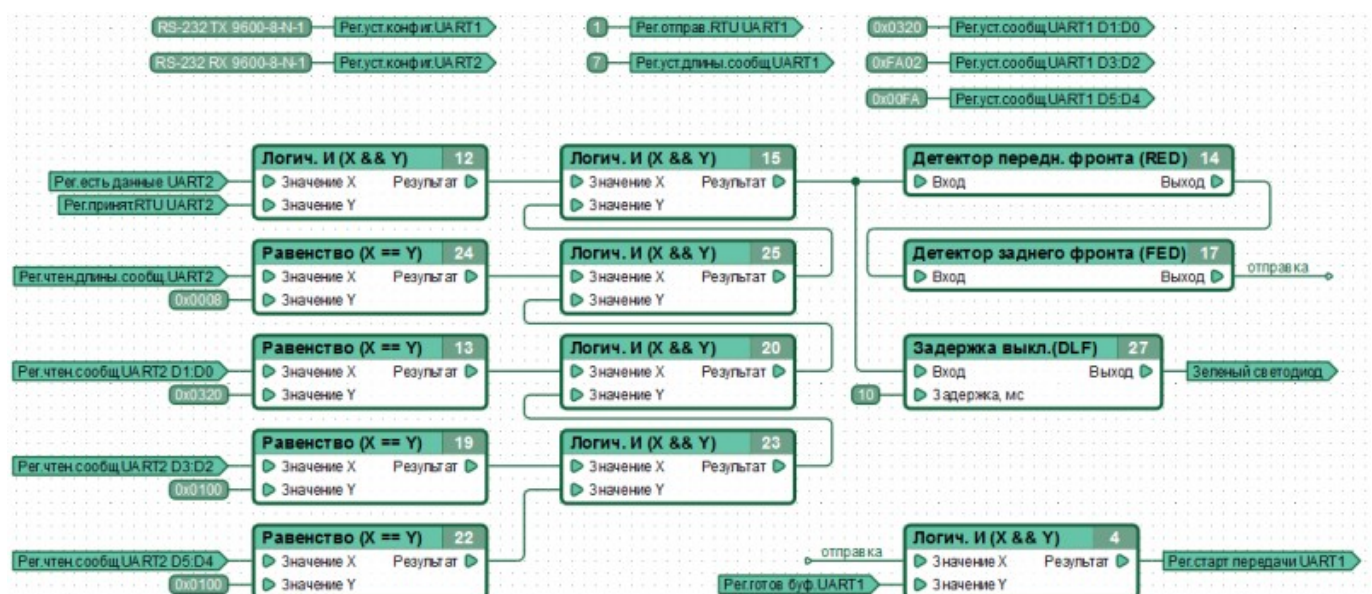
Функция: 0x03 - чтение значений из нескольких регистров хранения (байт D1 регистра приема сообщения UART2 D1:D0).

Номер первого запрашиваемого регистра: 0x0001 (байты D2 и D3 регистра приема сообщения UART2 D3:D2).

Число запрашиваемых регистров: 0x0001 (байты D4 и D5 регистра приема сообщения UART2 D5:D4).

Контрольная сумма: 0xD37B (байты D6 и D7 регистра приема сообщения UART2 D7:D6 — добавляются драйвером автоматически).

Пример функциональной диаграммы работы контроллера в качестве SLAVE-узла Modbus поверх RS-232. Получая от MASTERa запрос на передачу данных, контроллер в ответ передает состояние запрошенного регистра. Контроллер передает 2 байта данных (+2 байта CRC), т.к. регистры Modbus имеют разрядность 16 бит.



Строка ответа, при обмене данными между устройствами, будет выглядеть так:

<- 20 03 02 00 10 C6 A0

Адрес отвечающего устройства: 0x20 (байт D0 регистра сообщения передачи UART1 D1:D0).

Функция: 0x03 - результат чтения значений из нескольких регистров хранения (байт D1 регистра сообщения передачи UART1 D1:D0).

Число возвращаемых байт: 0x02 (байт D2 регистра сообщения передачи UART1 D3:D2).

Значение старшего байта запрашиваемого регистра: 0x00 (байт D3 регистра сообщения передачи UART1 D3:D2).

Значение младшего байта запрашиваемого регистра: 0x10 (байт D4 регистра сообщения передачи UART1 D5:D4).

Контрольная сумма: 0xC6A0 (байт D5 регистра приема сообщения UART1 D5:D4 и байт D6 регистра приема сообщения UART1 D7:D6 соответственно — добавляются драйвером автоматически).

# 7 CANNY 7, Драйвер CAN

## 7.1 Общее описание

Два специальных внешних вывода контроллера CANNY 7, расположенные на 4х контактном разъеме: CAN-H и CAN-L, предназначены для подключения к цифровой информационной **шине CAN**.

## 7.2 Регистры драйвера

Ниже приведено описание допустимых значений регистров управления работой драйвера CAN.

Регистры конфигурации драйвера CAN позволяют установить параметры работы контроллера в качестве узла шины CAN:

Регистр	Ожидаемые значения
Регистр установки конфигурации CAN	1...N = активизация драйвера и установка скорости приема/передачи CAN-сообщений (задается специальной константой из справочника констант); 0 = драйвер отключен.
Регистр установки фильтра приема CAN IDL №0 ... Регистр установки фильтра приема CAN IDL №15	0...0xFFFF = установить значение фильтра для младшей части идентификатора CAN-сообщения (биты 0...10 идентификатора стандартного формата или биты 0...15 идентификатора расширенного формата);
Регистр установки фильтра приема CAN IDH №0 ... Регистр установки фильтра приема CAN IDH №15	0...0x1FFF = установить значение фильтра для старшей части идентификатора CAN-сообщения (биты 16...28 идентификатора расширенного формата);
Регистр установки режима пассивного приема	≥ 1 = включен режим пассивного приема (listen only) сообщений CAN;



сообщений CAN	0	= включен режим нормального приема-передачи (normal) сообщений CAN;
Регистр режима фильтрации приема данных CAN	$\geq 1$	= режим фильтрации принимаемых сообщений CAN включен;
	0	= режим фильтрации принимаемых сообщений CAN отключен;
Регистр запрета автоматической повторной отправки сообщения CAN при ошибке передачи	$\geq 1$	= автоматическая повторная отправка сообщений CAN при ошибке передачи выключена (запрещена);
	0	= автоматическая повторная отправка сообщений CAN при ошибке передачи включена (разрешена).

*Примечание: В режиме пассивного приема сообщений CAN (listen only) в отличие от нормального режима CAN (normal) драйвер выполняет прием данных из CAN-шины, но при этом не отправляет подтверждение их приема и не переводит сеть в состояние ошибки при обнаружении таковой. Таким образом контроллер остается незаметным для остальных устройств на шине, никак себя не проявляя. Для нормальной работы сети, в ней должны находиться минимум два устройства работающие в режиме normal.*

*Примечание: При включенном режиме фильтрации CAN драйвер будет принимать только те сообщения, идентификаторы которых совпадают с указанными в регистрах установки фильтра приема сообщений CAN значениями, игнорируя все остальные.*

*Примечание: При включенном режиме запрета автоматической повторной отправки сообщения CAN при ошибке передачи, драйвер будет предпринимать только однократные попытки отправки сообщения CAN, вне зависимости от успешности его получения приемником, предотвращая, таким образом, возникновение ошибки приема (зависания) шины в случае отсутствия в сети приемника данного сообщения, иначе, в соответствии со стандартом CAN, сообщение будет отправляться до получения подтверждения его успешного приема хотя бы одним приемником.*

Конфигурация драйвера CAN, определяется константой, задающей скорость приема/передачи данных.

Параметр	Перечень допустимых значений
Скорость приема/передачи данных, Кбит/с	10; 20; 33; 50; 83; 95.2; 100; 125; 250; 500; 1000

Именованные константы, определяющие конфигурацию CAN-драйвера, содержатся в разделе «Конфигурация CAN» справочника констант CannyLab, доступ к которому осуществляется через контекстное меню входа функционального блока, имеющего тип «Константа».

Регистры диагностики драйвера CAN позволяют пользователю определить состояние драйвера в тот или иной момент выполнения диаграммы.

Регистр	Возвращаемые значения	
Регистр отсутствия активности драйвера CAN	1	= активность драйвера CAN отсутствует, шина бездействует, прием данных не осуществляется;
	0	= регистрируется активность CAN.
Регистр переполнения буфера приема CAN	1	= ошибка, буфер CAN переполнен;
	0	= переполнение буфера приема отсутствует.
Регистр ошибки приема CAN	1	= уровень ошибок приема CAN превысил допустимый порог;
	0	= уровень ошибок приема CAN ниже допустимого порога.
Регистр готовности буфера передачи данных CAN	1	= буфер передачи данных драйвера CAN свободен и готов к загрузке новых сообщений;
	0	= буфер передачи данных драйвера CAN не готов.

Регистры приема драйвера CAN позволяют получить доступ к значениям, полученным по шине.

Регистр	Возвращаемые значения	
Регистр наличия принятых данных CAN	1	= в буфере приема драйвера CAN находится полученное сообщение, данное значение появляется в регистре на один цикл выполнения диаграммы сообщая об актуальности данных, находящихся в буфере приема;
	0	= в буфере приема драйвера CAN нет актуальных данных.
Регистр принятого сообщения CAN IDL	0...0xFFFF = значение младшей части идентификатора полученного CAN-сообщения.	
Регистр принятого сообщения CAN IDH	0...0x1FFF = значение старшей части идентификатора полученного CAN-сообщения.	
Регистр принятого сообщения CAN ERL	0...0xC000 = значение, равное количеству байт данных в принятом сообщении, признаки EXT и RTR (см. примечание).	
Регистр принятого сообщения CAN D1:D0 ...	0...0xFFFF = значения соответствующих байт данных принятого сообщения CAN, по два байта на регистр.	

Регистр принятого сообщения CAN D7:D6	
---------------------------------------	--

*Примечание: Регистр принятого сообщения CAN ERL, помимо числа байт в принятом сообщении 0...8 в младших битах, содержит в своих старших битах информацию о специальных признаках сообщения: бит 15 - признак EXT и бит 14 признак RTR. Где EXT = 1 при приеме сообщения в расширенном формате, EXT = 0 при стандартном формате сообщения; RTR = 1 при приеме удаленного запроса данных, EXT = 0 при приеме обычного сообщения.*

Регистры передачи сообщений CAN используются для размещения в буфере передачи драйвера данных, подлежащих отправке.

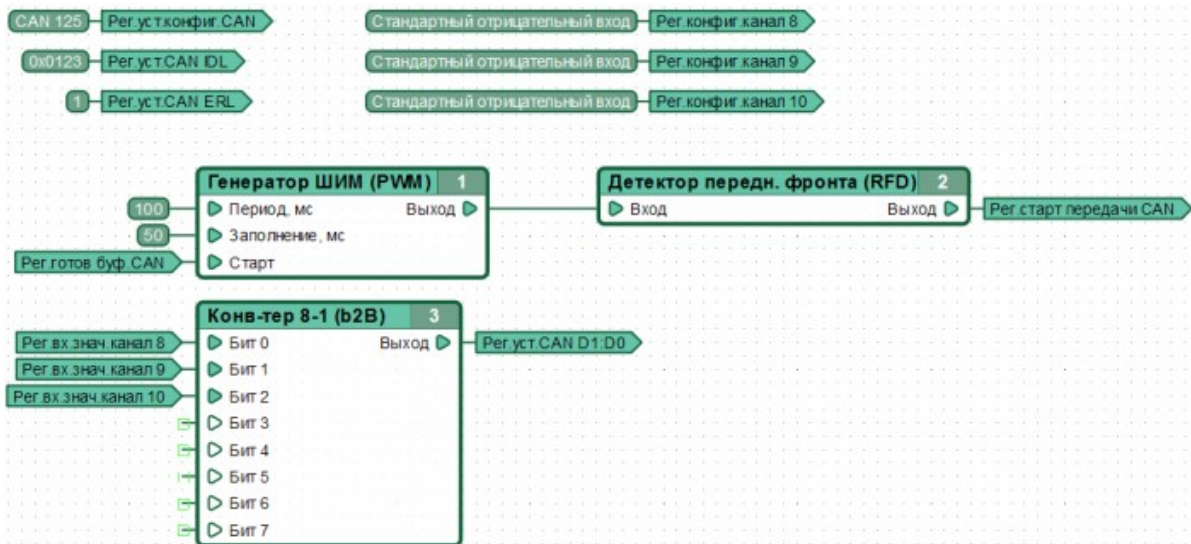
Регистр	Ожидаемые значения
Регистр начала передачи CAN	$\geq 1$ = загрузить данные из регистров передачи в буфер передачи драйвера CAN; 0 = не загружать данные в буфер передачи драйвера CAN.
Регистр сообщения передачи CAN IDL	0...0xFFFF = значение младшей части идентификатора передаваемого CAN-сообщения.
Регистр сообщения передачи CAN IDH	0...0x1FFF = значение старшей части идентификатора передаваемого CAN-сообщения.
Регистр сообщения передачи CAN ERL	0...0xC000 = значение, равное количеству байт данных в передаваемом сообщении, признаки EXT и RTR (см. примечание).
Регистр сообщения передачи CAN D1:D0 ... Регистр сообщения передачи CAN D7:D6	0...0xFFFF = значения соответствующих байт данных передаваемого сообщения CAN, по два байта на регистр.

*Примечание: Регистр сообщения передачи CAN ERL, помимо числа байт в передаваемом сообщении 0...8 в младших битах, содержит в своих старших битах информацию о специальных признаках сообщения: бит 15 - признак EXT и бит 14 признак RTR. Где EXT = 1 при передаче сообщения в расширенном формате, EXT = 0 при стандартном формате сообщения; RTR = 1 при передаче удаленного запроса данных, EXT = 0 при передаче обычного сообщения.*

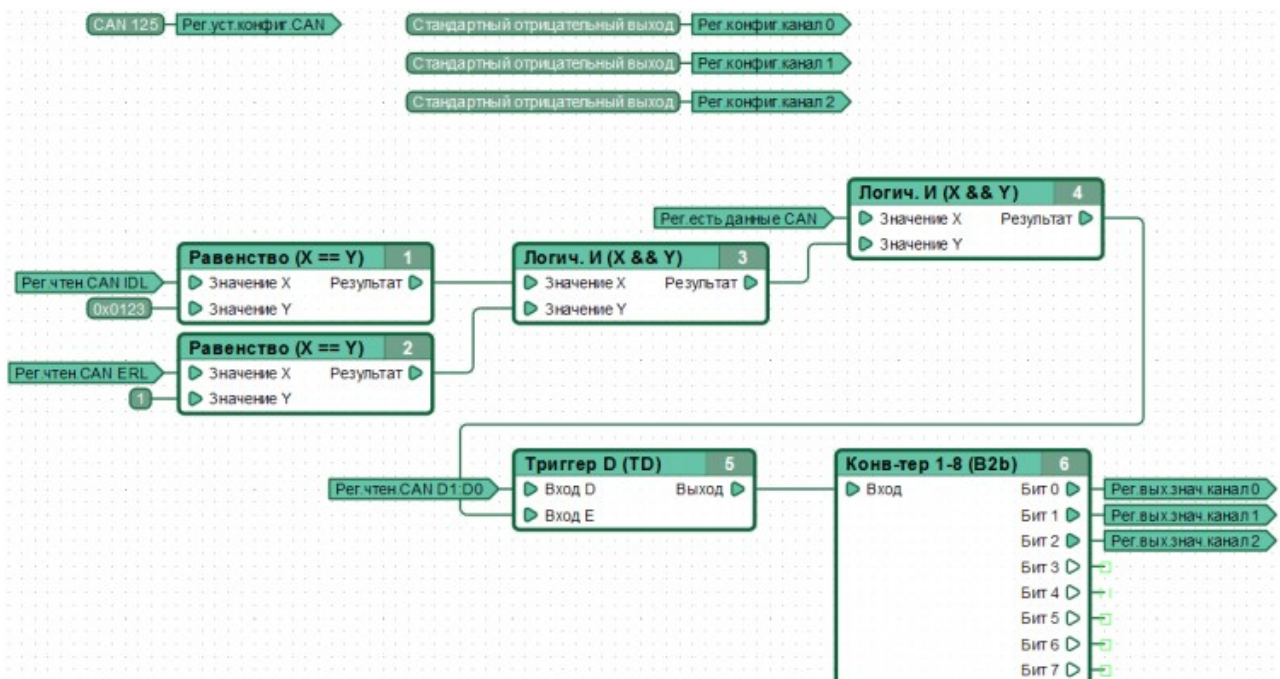


## 7.3 Примеры

Пример функциональной диаграммы отправки данных в шину CAN. Выполняя диаграмму контроллер, с периодичностью 1 раз в 100мс, передает в шину на скорости 125 кБод данные о состоянии трех своих входов, используя сообщения стандартного формата с идентификатором 0x123, содержащие один байт данных.

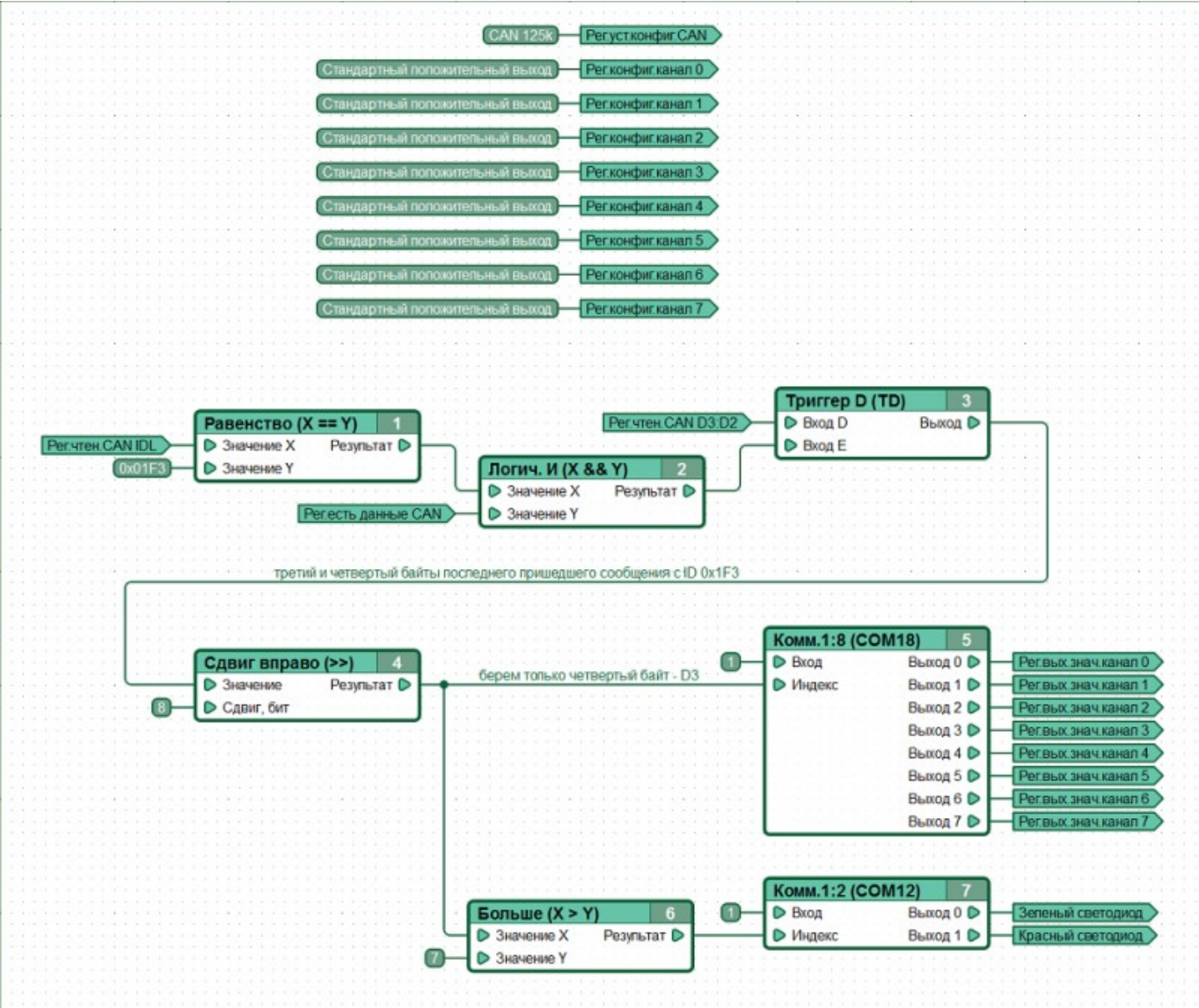


Пример функциональной диаграммы приема данных из шины CAN. Выполняя диаграмму контроллер, получая по шине сообщения стандартного формата с идентификатором 0x0123, устанавливает на трех своих выходах состояние в соответствии с полученным значением.



Пример функциональной диаграммы приема данных из шины CAN. Выполняя диаграмму контроллер, получая по шине сообщения стандартного формата с идентификатором

0x01F3, анализирует содержащееся в одном из байтов сообщения значение и устанавливает на своих выходах и встроенном светодиоде соответствующие состояния.



# 8 CANNY 7, Драйвер LIN

## 8.1 Общее описание

Два из одиннадцати каналов ввода-вывода CANNY 7, которые могут быть переданы под управление драйвера UART/RS-232 (Канал №9 и Канал №10), могут быть использованы для организации приема-передачи данных как два независимых канала драйвера **LIN**.

Каналы драйвера LIN могут подключаться как вместе так и по отдельности, иметь индивидуальные настройки скорости передачи данных, подтяжки линии и типа узла сети MASTER или SLAVE.

Драйвер LIN в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвер дискретного ввода-вывода. Таким образом, при активации драйвера LIN, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера дискретного ввода-вывода будет проигнорировано контроллером.

## 8.2 Регистры драйвера

Ниже приведено описание допустимых значений регистров управления работой драйвера LIN.

Регистры конфигурации драйвера LIN.

Регистры конфигурации драйвера LIN позволяют установить параметры работы контроллера в качестве узла шины LIN:

Регистр	Ожидаемые значения
Регистр конфигурации LINx	1...N = установить конфигурацию канала драйвера LIN контроллера, определяющую текущий режим и параметры его работы (задается специальной константой из справочника констант);  0 = отключить канал от драйвера LIN, вернуть управление каналом драйверу каналов ввода-вывода и разрешить изменения его состояния из функциональной диаграммы.

Конфигурация канала для работы в данном режиме, определяется константой, представляющей собой комбинацию параметров: версии протокола, скорости передачи данных, режима работы и наличия внутренней «подтяжки» контакта контроллера.

Параметр	Перечень допустимых значений
Версия протокола LIN	1.3; 2.0
Скорость передачи данных, бод	2400; 9600; 19200
Режим работы	MASTER; SLAVE
Подтяжка	плюс; воздух

Именованные константы, представляющие доступные пользователю комбинации параметров конфигурации LIN, содержатся в разделе «Конфигурация LIN» справочника констант CanppuLab, доступ к которому осуществляется через контекстное меню входа функционального блока, имеющего тип «Константа».

Регистры диагностики драйвера LIN.

Регистр	Возвращаемые значения	
Регистр переполнения буфера LINx	1	= буфер соответствующего канала драйвера LIN переполнен;
	0	= переполнения соответствующего канала драйвера LIN не зафиксировано.
Регистр ошибки приема LINx	1	= во время приема сообщения по соответствующему каналу драйвера LIN произошла ошибка;
	0	= драйвер работает в нормальном режиме.
Регистр готовности буфера передачи данных LINx	1	= буфер передачи данных соответствующего канала драйвера LIN свободен и готов к работе;
	0	= буфер передачи данных драйвера LIN занят.
Регистр успешной отправки данных драйвера LINx	1	= при работе в режиме SLAVE, признак успешной отправки данных из буфера передачи соответствующего канала драйвера LIN в ответ на запрос MASTER-узла;
	0	= при работе соответствующего канала драйвера LIN в режиме SLAVE, запроса от MASTER-узла не поступало.
Регистр флагов успешной отправки дополнительных буферов LINx SLAVE	1...15	= при работе в режиме MULTISLAVE, признак успешной отправки данных из дополнительного буфера передачи соответствующего канала драйвера LIN в ответ на запрос MASTER-узла; при успешной отправке значение "1" присваивается биту, номер которого совпадет с номером отправленного дополнительного буфера SLAVE, т.е. при успешной отправке буфера №0 - регистр

	<p>флага = 1 = 0x01 = 0b0001, буфера №1 - регистр  флага = 2 = 0x02 = 0b0010, буфера №2 - регистр  флага = 4 = 0x04 = 0b0100, буфера №3 - регистр  флага = 8 = 0x08 = 0b1000;</p> <p>0 = при работе соответствующего канала драйвера LIN в режиме MULTISLAVE, запроса на получение данных от узлов, номера которых присвоены дополнительным буферам LIN SLAVE, от MASTER-узла не поступало.</p>
--	---

Регистры приема драйвера LIN.

Регистр	Возвращаемые значения	
Регистр наличия принятых данных LINx	1	= сообщение успешно получено и доступно в регистрах буфера приема соответствующего канала драйвера LIN;
	0	= в буфере приема соответствующего канала драйвера LIN отсутствуют актуальные данные.
Регистр отсутствия активности драйвера LINx	1	= активность соответствующего канала драйвера LIN отсутствует, линия находится в пассивном режиме;
	0	= зафиксирована активность на линии соответствующего канала драйвера LIN.
Регистр принятого сообщения LINx LEN	0...8	= значение, равное количеству байт данных в пакете, принятом по соответствующему каналу драйвера LIN.
Регистр принятого сообщения LINx ID	0...63	= значение идентификатора полученного по соответствующему каналу драйвера LIN сообщения. (см. примечание)
Регистр принятого сообщения LINx D1:D0 ... Регистр принятого сообщения LINx D7:D6	0...0xFFFF = значения соответствующих байт данных приемных буферов LIN каждого канала, по два байта на регистр.	

*Примечание: В регистре идентификатора принятого сообщения LIN отображаются только младшие 6 бит идентификатора: 4 бита адреса устройства и 2 бита использовавшихся в LIN версии 1.1 для кодирования длины сообщения, а в более поздних версиях расширяющие адрес. 10 старших бит регистра идентификатора принятого сообщения LIN всегда равны нулю.*

Регистры передачи драйвера LIN.

Регистр	Ожидаемые значения	
Регистр начала передачи LINx	≥ 1	= загрузить данные из регистров передачи в буфер передачи соответствующего канала драйвера LIN;
	0	= не загружать данные в буфер передачи соответствующего канала драйвера LIN.
Регистр номера дополнительного буфера передачи LINx SLAVE	0...3	= при работе в режиме MULTISLAVE, номер дополнительного буфера передачи SLAVE соответствующего канала драйвера LIN, заполнение которого осуществляется на данном цикле выполнения диаграммы.
Регистр сообщения передачи LINx LEN	0...8	= количество байт данных, которое будет необходимо передать в линию, при получении команды на отправку соответствующего канала драйвера LIN.
Регистр сообщения передачи LINx ID	0...63	= значение идентификатора LIN-сообщения для передачи по соответствующему каналу драйвера LIN.
Регистр сообщения передачи LINx D1:D0 ... Регистр сообщения передачи LINx D7:D6	0...0xFFFF	= значения соответствующих байт данных для передачи по соответствующему каналу драйвера LIN, по два байта на регистр.

*Примечание: При передаче LIN-сообщения, драйвер отбрасывает все кроме младших 6 бит значения установленного в регистре идентификатора LIN-сообщения, автоматически генерирует два бита четности и дополняет ими идентификатор, в соответствии с требованиями стандарта.*

*Примечание: Если, работая в режиме SLAVE, регистр номера дополнительных буферов LIN SLAVE не использован, т.е. дополнительные буферы LIN SLAVE не заполнены, то драйвер работает в обычном SLAVE-режиме, с единственным буфером передачи LIN.*

*Примечание: Если, работая в режиме SLAVE, дополнительные буферы LIN SLAVE заполнены с помощью регистра номера дополнительных буферов LIN SLAVE, то драйвер работает в режиме MULTISLAVE.*

*Примечание: При работе в режиме MULTISLAVE, дополнительные буферы LIN SLAVE заполняются последовательно, по одному за один цикл выполнения диаграммы.*

*Примечание: При работе в сетях LIN версии ниже 1.3, будьте внимательны при формировании исходящих сообщений. Драйвер позволяет использовать комбинации значений длины и идентификатора передаваемого сообщения недопустимые в этом стандарте.*



## 8.3 Работа контроллера в режиме MASTER

Для перевода канала драйвера LIN в режим MASTER, необходимо в соответствующий каналу драйвера «Регистр конфигурации LINx» передать значение константы, соответствующей выбранному режиму работы.

Для получения MASTERом данных от SLAVE-узла, необходимо передать в шину LIN соответствующий запрос: отправить заголовок сообщения, содержащий идентификатор ведомого узла, от которого запрашиваются данные. Длина сообщения передачи LIN LEN должна быть установлена равной нулю, в регистр старта передачи должно быть записано ненулевое значение. При получении ответа от ведомого устройства, он будет помещен в Регистры принятого сообщения соответствующего канала драйвера LIN с одновременной установкой признака в Регистре наличия принятых данных LIN этого канала.

Пример функциональной диаграммы для получения MASTERом данных от SLAVE-устройства. MASTER каждые 100мс отправляет в шину запрос на получение данных от ведомого устройства с идентификатором 0x02. Получив ответ, контроллер запоминает 2 первых байта данных, в D-триггере.

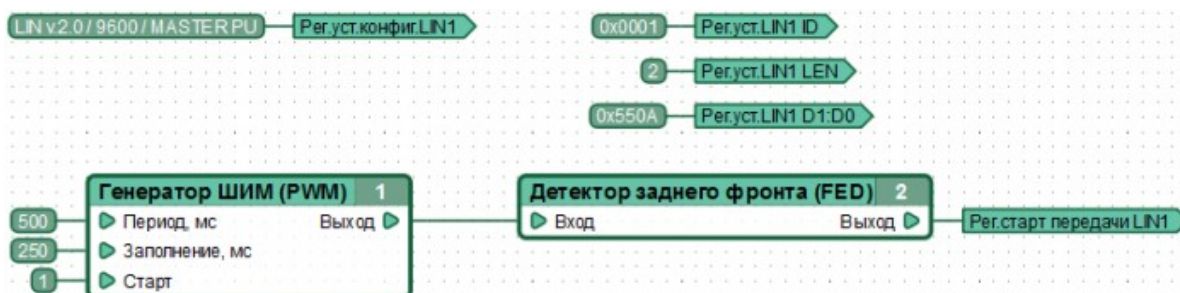


Для передачи данных в режиме MASTER необходимо заполнить регистры идентификатора, и данных передаваемого сообщения LIN, установив при этом в Регистре передаваемого сообщения LIN LEN, значение равное количеству байт данных передаваемого сообщения, которое должно быть больше нуля но меньше девяти. Команда драйверу на отправку сообщения посылается путем установки ненулевого значения в «Регистр начала передачи сообщения LINx», в результате чего, содержимое регистров данных копируется в буфер передачи LIN-сообщения, если он свободен, драйвер немедленно приступит к отправке сообщения.

*Примечание: С целью предотвращения потери данных, перед отправкой нового сообщения LIN рекомендуется убедиться в готовности буфера передачи*

соответствующего канала драйвера LIN к передаче очередного сообщения, проверив значение в Регистре готовности буфера передачи данных LINx.

Пример функциональной диаграммы отправки данных в шину LIN ведущим устройством. MASTER каждые 500мс отправляет в шину сообщение с идентификатором 0x01, содержащее 2 байта данных (0x0A и 0x55).



*Примечание: Если в регистре начала передачи сообщения LINx постоянно установлено ненулевое значение, то попытки копирования данных в буфер передачи для отправки сообщения LIN будут предприниматься на каждом цикле выполнения диаграммы. Во избежание переполнения буфера передачи LIN иницируйте начало передачи данных единичными импульсами, используя, например, функциональные блоки детекторов фронта.*

## 8.4 Работа контроллера в режиме SLAVE

Работая в режиме SLAVE, узел сети LIN успешно принимает любые данные передаваемые по сети, но не может сам передавать данные в LIN, не получив на то соответствующего запроса от MASTER.

Для перевода канала контроллера в режим SLAVE, необходимо в соответствующий каналу «Регистр конфигурации LINx» передать значение константы, соответствующей выбранному режиму работы.

Для обеспечения успешной отправки данных узлом находящемся в режиме SLAVE по запросу MASTER-узла, необходимо заранее, не дожидаясь запроса, подготовить данные к передаче. Для этого необходимо заполнить все необходимые регистры сообщения передачи драйвера LIN, а именно: регистр идентификатора LIN ID, регистр длины сообщения LIN LEN и разместить передаваемую информацию в регистрах данных LIN. Разрешение на отставку сообщения дается путем установки значения «1» в «Регистр начала передачи сообщения LINx», в результате чего содержимое регистров данных копируется в буфер передачи LIN-сообщения, если он свободен. Отправка однажды помещенных в буфер передачи данных будет выполняться драйвером автоматически, при каждом получении запроса от MASTERa.

*Примечание: Если в регистре начала передачи сообщения LINx постоянно установлено*



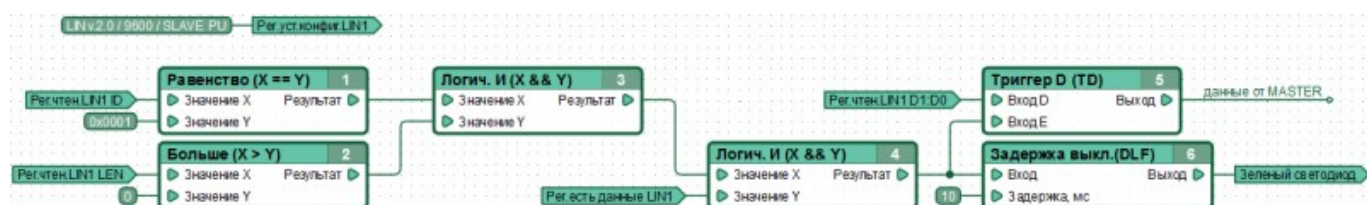
ненулевое значение, то попытки копирования данных в буфер передачи для отправки сообщения LIN будут предприниматься на каждом цикле выполнения диаграммы. Во избежание переполнения буфера передачи LIN иницилируйте начало передачи данных единичным импульсом, используя, например, функциональные блоки детекторов фронта.

Контролировать поступление запроса от MASTER и успешную отправку автоматического ответа на него драйвером узла находящегося в режиме SLAVE можно по появлению значения «1» в Регистре успешной отправки данных драйвера LINx. Обновлять данные находящиеся в буфере передачи возможно после его освобождения. Состояние буфера передачи можно отслеживать по значению соответствующего регистра драйвера.

Пример функциональной диаграммы передачи данных LIN узлом в режиме SLAVE.



Пример функциональной диаграммы получения SLAVEом данных от ведущего устройства. Получая сообщение с идентификатором 0x01, содержащее какие-либо данные, контроллер сохраняет первые два байта данных сообщения в D-триггере.



## 8.5 Работа контроллера в режиме MULTISLAVE

Работая в режиме MULTISLAVE, каждый интерфейс LIN контроллера может выступать в качестве нескольких (максимум четырех) узлов сети LIN, т.е. отвечать на запросы MASTERa по нескольким ID.

Переход драйвера соответствующего интерфейса LIN в режим MULTISLAVE осуществляется автоматически при заполнении данными дополнительных буферов LINx SLAVE.

При получении от MASTERa запроса на получение данных от узла LIN-сети, идентификатор которого присвоен одному из дополнительных буферов LINx SLAVE,

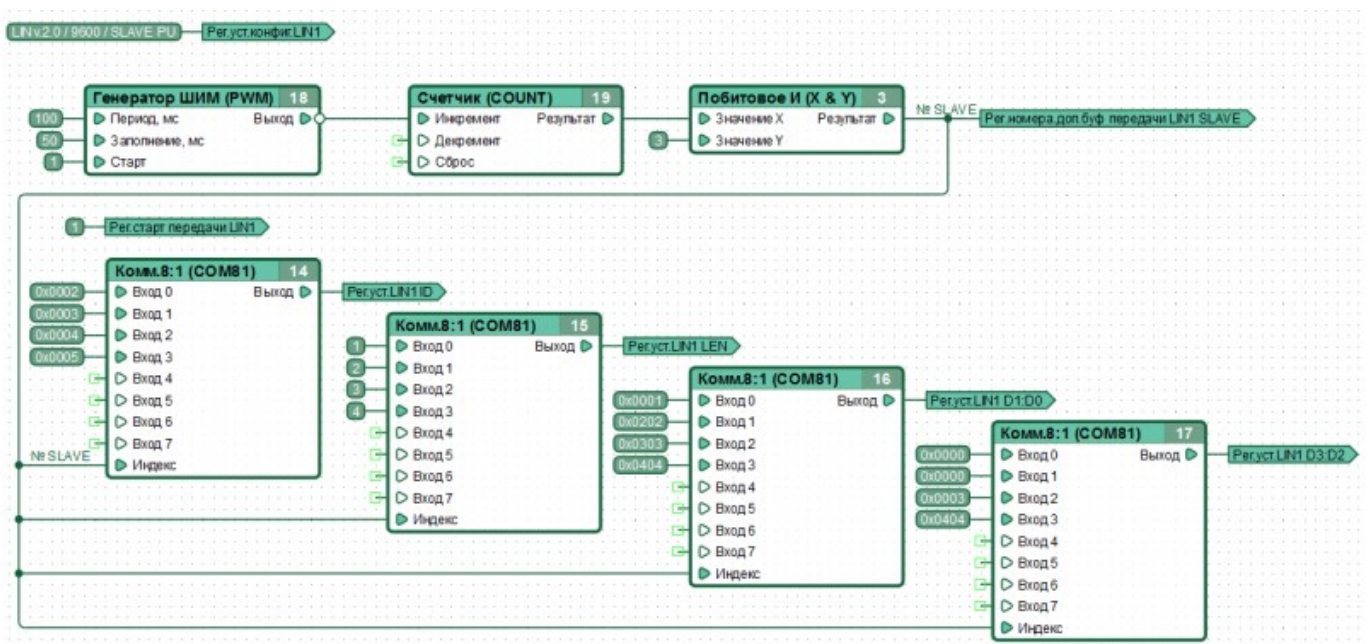
данные будут переданы автоматически, асинхронно с работой функциональной диаграммы.

Информация о успешной отправке данных из дополнительных буферов LINx SLAVE доступна пользователю через *"Регистр флагов отправки данных дополнительных буферов LINx SLAVE"*.

Заполнение данными дополнительных буферов LINx SLAVE происходит через общие для всех буферов регистры: *Регистр начала передачи LINx*, *Регистр сообщения передачи LINx LEN*, *Регистр сообщения передачи LINx ID* и регистры *Регистр сообщения передачи LINx Dy:Dz*.

При этом данные и ID сообщения помещаются в тот буфер, номер которого (от 0 до 3) установлен в *Регистре номера дополнительного буфера передачи LINx SLAVE* на момент записи в *Регистр начала передачи LINx* значения 1.

Пример работы контроллера в режиме MULTISLAVE, в котором после подачи питания выполняется заполнение и постоянное обновление 4х дополнительных буферов передачи LIN1 SLAVE. Заполнение буферов выполняется последовательно, по одному на каждом цикле выполнения диаграммы:



Фактически, работа интерфейса LIN в режиме SLAVE представляет собой работу в режиме MULTISLAVE с одним задействованным буфером передачи - буфером №0.

## 8.6 Режим экономии энергии (режим пониженного энергопотребления)

Для экономии энергии спецификацией LIN предусмотрена возможность перехода всех узлов сети LIN в режим пониженного энергопотребления. Чаще всего, такой переход инициируется ведущим устройством путем отправки соответствующего сообщения в шину, либо осуществляется ведомыми устройствами самостоятельно, при отсутствии активности на шине в течение определенного интервала времени. Выход из режима сна может быть инициирован любым узлом шины. MASTER-узел инициирует пробуждение шины обычным началом опроса SLAVE-узлов. Пробуждение же по инициативе SLAVE-узла, осуществляется кратковременной установкой на линии потенциала GND.

В драйвере LIN контроллеров CANNY 7 для пробуждения шины MASTER-устройству достаточно начать передачу данных или запросов ведомым устройствам. SLAVE-узлы отправляют запрос на пробуждение шины передачей сообщения с любым идентификатором, но имеющим нулевую длину, т. е. «Регистр сообщения передачи LINx LEN» = 0.

# 9 CANNY 7, Драйвер I<sup>2</sup>C

## 9.1 Общее описание

В качестве линий связи (SDA и SCL) может быть назначена любая пара каналов контроллера CANNY 7. При этом, данные каналы должны быть подтянуты к напряжению 5В резисторами номиналом от 1 кОм до 10 кОм снаружи. Особенность реализации протокола I<sup>2</sup>C в контроллерах CANNY7 состоит в том, что CANNY7 может выступать только в качестве ведущего (Master) узла сети и обмен данными между устройствами, который может быть как одно- так и двунаправленным, происходит отдельными сеансами, с максимальной длиной сообщения I<sup>2</sup>C внутри одного сеанса равной 16 байтам, т. е. открытие одновременно несколько сеансов с разными устройствами не допускается. Скорость обмена фиксированная и составляет 100 кбит/с. Общее число ведомых устройств на линии может достигать нескольких десятков.

Драйвер I<sup>2</sup>C в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвер дискретного ввода-вывода. Таким образом, при активации драйвера I<sup>2</sup>C, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера дискретного ввода-вывода будет проигнорировано контроллером.

## 9.2 Регистры драйвера I<sup>2</sup>C

Ниже приведено описание допустимых значений регистров управления работой драйвера I<sup>2</sup>C.

Регистры конфигурации драйвера I<sup>2</sup>C.

Регистр	Ожидаемые значения				
Регистр адреса I <sup>2</sup> C	2...254 = четное число, адрес slave-устройства, с которым будет производиться обмен данными по шине I <sup>2</sup> C.				
Регистр активации драйвера I <sup>2</sup> C	<table><tr><td>≥ 1</td><td>= активировать драйвер I<sup>2</sup>C, передать ему управление каналами контроллера, используемыми в качестве линий SDA и SCL;</td></tr><tr><td>0</td><td>= деактивировать драйвер I<sup>2</sup>C, вернуть управление каналами драйверу ввода-вывода и разрешить изменения их состояний из функциональной диаграммы.</td></tr></table>	≥ 1	= активировать драйвер I <sup>2</sup> C, передать ему управление каналами контроллера, используемыми в качестве линий SDA и SCL;	0	= деактивировать драйвер I <sup>2</sup> C, вернуть управление каналами драйверу ввода-вывода и разрешить изменения их состояний из функциональной диаграммы.
≥ 1	= активировать драйвер I <sup>2</sup> C, передать ему управление каналами контроллера, используемыми в качестве линий SDA и SCL;				
0	= деактивировать драйвер I <sup>2</sup> C, вернуть управление каналами драйверу ввода-вывода и разрешить изменения их состояний из функциональной диаграммы.				
Регистр номера канала SDA драйвера I <sup>2</sup> C	0...10 = установить номер канала контроллера для использования драйвером I <sup>2</sup> C в качестве линии				

	передачи данных.
Регистр номера канала SCL драйвера I <sup>2</sup> C	0...10 = установить номер канала контроллера для использования драйвером I <sup>2</sup> C в качестве линии передачи тактирующих импульсов.

*Примечание: Для линии SDA и линии SCL должны быть назначены отдельные каналы контроллера.*

Регистры диагностики драйвера I<sup>2</sup>C.

Регистр	Возвращаемые значения
Регистр ошибки приема/передачи данных I <sup>2</sup> C	1 = во время приема или отправки сообщения I <sup>2</sup> C произошла ошибка; 0 = драйвер работает в нормальном режиме.

Регистры приема драйвера I<sup>2</sup>C.

Регистр	Возвращаемые значения
Регистр наличия принятых данных I <sup>2</sup> C	1 = сообщение успешно получено и доступно в регистрах буфера приема драйвера I <sup>2</sup> C; 0 = в буфере приема драйвера I <sup>2</sup> C отсутствуют актуальные данные.
Регистр длины принимаемого сообщения I <sup>2</sup> C	0...16 = значение, равное количеству байт, которые должны быть приняты в сообщении I <sup>2</sup> C.
Регистр принятых данных I <sup>2</sup> C D1:D0 ... Регистр принятых данных I <sup>2</sup> C D15:D14	0...0xFFFF = значения соответствующих байт данных приемного буфера I <sup>2</sup> C, по два байта на регистр.

Регистры передачи драйвера I<sup>2</sup>C.

Регистр	Ожидаемые значения
Регистр начала обмена данными I <sup>2</sup> C	≥ 1 = загрузить данные из регистров передачи в буфер передачи / загрузить данные из буфера приема в регистры принятых данных драйвера I <sup>2</sup> C; 0 = не загружать данные в буфер передачи / не считывать данные из буфера приема драйвера I <sup>2</sup> C.
Регистр длины передаваемого сообщения I <sup>2</sup> C	0...16 = количество байт сообщения I <sup>2</sup> C, которое будет необходимо передать при получении команды на отправку данных.

Регистр передаваемого сообщения I <sup>2</sup> C D1:D0 ...	0...0xFFFF = значения соответствующих байт сообщения I <sup>2</sup> C для передачи, по два байта на регистр.
Регистр передаваемого сообщения I <sup>2</sup> C D15:D14	

## 9.3 Особенности работы драйвера I<sup>2</sup>C

Обмен данными в сети I<sup>2</sup>C с использованием контроллера CANNY7 определяется комбинацией значений регистров длины передаваемого и принимаемого сообщения, установленных пользователем (смотри таблицу).

Значение регистра длины передаваемого сообщения I <sup>2</sup> C	Значение регистра длины принимаемого сообщения I <sup>2</sup> C	Направление обмена данными
> 0	= 0	Только передача данных от CANNY7 (Master) ведомому (Slave) устройству с адресом, указанным в соответствующем регистре.
> 0	> 0	Передача данных от CANNY7 (Master) ведомому (Slave) устройству с адресом, указанным в соответствующем регистре, и получение от него ответных данных.
= 0	> 0	Только прием данных CANNY7 (Master) от ведомого (Slave) устройства с адресом, указанным в соответствующем регистре.

Пример функциональной диаграммы получения данных от датчика температуры TCN75. В процессе работы контроллер выступает в качестве ведущего (Master) узла шины I<sup>2</sup>C, выполняя опрос датчика температуры 2 раза в секунду.



# 10 CANNY 7, Драйвер Dallas 1-Wire

## 10.1 Общее описание

Контроллер CANNY7 может быть использован в качестве ведущего (MASTER) узла в однопроводной сети передачи данных **Dallas 1-Wire**, при этом он имеет возможность только отправлять запросы на получение данных от ведомых устройств.

Для подключения контроллера CANNY7 к шине 1-Wire может использоваться любой из его каналов ввода-вывода. При этом, данный канал должен быть снаружи подтянут к напряжению 5В резистором номиналом от 3 кОм до 7 кОм. В контроллерах CANNY7 предусмотрена возможность обращения к конкретному устройству на шине 1-Wire по его адресу, что позволяет организовать работу с контроллером нескольких ведомых устройств по одному каналу, в том числе, используя несколько каналов контроллера, возможно его последовательное подключение к нескольким шинам 1-Wire.

Драйвер Dallas 1-Wire в своей работе использует ресурсы каналов контроллера, но имеет более высокий приоритет чем драйвера ввода-вывода. Таким образом, при активации драйвера Dallas 1-Wire, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера ввода-вывода будет проигнорировано контроллером.

Ведомое устройство должно иметь постоянное, а не паразитное питание.

## 10.2 Регистры драйвера 1-Wire

Ниже приведено описание регистров управления работой драйвера 1-Wire.

Регистры конфигурации драйвера 1-Wire.

Для активации драйвера 1-Wire необходимо установить номер канала контроллера, подключенного к шине 1-Wire, с которым предполагается начать работу, в соответствующий регистр драйвера.

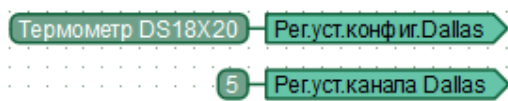
*Примечание: В каждый отдельный момент времени драйвер может работать только с одной из подключенных к нему шин 1-Wire, при этом допускается динамическое переключение между шинами в процессе выполнения функциональной диаграммы.*

Управление отправкой запросов на получение данных от ведомых устройств и контроль получения их ответов (опросов датчиков) выполняется с помощью соответствующих регистров драйвера 1-Wire.



*Примечание: Во избежание потери данных при работе с несколькими шинами Dallas, выполняйте переключение между ними только после получения от ведомых устройств ответов на запрос контроллера и обработки или сохранения полученных данных.*

Пример конфигурации канала №5 для работы с датчиком температуры DS18X20.



Регистры конфигурации драйвера 1-Wire.

Регистр	Ожидаемые значения
Регистр конфигурации Dallas	1...N = активизация драйвера и установка типа подключаемого устройства (задается специальной константой из справочника констант); 0 = деактивировать драйвер Dallas 1-Wire.
Регистр номера канала Dallas	0...10 = установить номер канала контроллера используемого для работы с шиной 1-Wire.

Конфигурация канала для работы в данном режиме задается следующим именованными константами:

Параметр	Перечень допустимых значений
Тип устройства 1-Wire	Датчик температуры DS18X20, Считыватель CP-Z в режиме эмуляции DS1990A, Считыватель DS1990A

Регистры передачи драйвера 1-Wire.

С помощью регистров передачи драйвера 1-Wire пользователь может задать адрес опрашиваемого ведомого устройства и отправить запрос на получение данных.

Регистр	Ожидаемые значения
Регистр начала передачи Dallas	$\geq 1$ = отправить ведомому устройству запрос на получение данных; 0 = не отправлять ведомому устройству запрос на получение данных.
Регистр установки 64-битного ROM-кода адресата передачи Dallas SN0:FC	0...0xFFFF = значение регистра: младшая часть содержит идентификатор семейства устройства (FC), старшая часть содержит первый байт уникального адреса устройства (SN0).

Регистр установки 64-битного ROM-кода адресата передачи Dallas SN2:SN1	0...0xFFFF = значение регистра: младшая часть содержит второй байт уникального адреса устройства (SN1), старшая часть содержит третий байт уникального адреса устройства (SN2).
Регистр установки 64-битного ROM-кода адресата передачи Dallas SN4:SN3	0...0xFFFF = значение регистра: младшая часть содержит четвертый байт уникального адреса устройства (SN3), старшая часть содержит пятый байт уникального адреса устройства (SN4).
Регистр установки 64-битного ROM-кода адресата передачи Dallas CRC:SN5	0...0xFFFF = значение регистра: младшая часть содержит шестой байт уникального адреса устройства (SN5), старшая часть содержит контрольную сумму ROM-кода адресата (CRC).

*Примечание: Регистры установки 64-битного ROM-кода адресата передачи Dallas используются только для выбора устройства на шине 1-Wire по его уникальному номеру (ROM-коду), в случае подключения к одной шине нескольких устройств. При работе с единственным устройством на шине, в случае если его ROM-код неизвестен, установите значение «0» во все регистры ROM-кода адресата передачи Dallas.*

Регистры приема драйвера 1-Wire.

В процессе работы, при получении от ведомых устройств ответов на запросы контроллера, данные, в зависимости от конфигурации драйвера 1-Wire, размещаются в его соответствующие регистры приема.

Общие регистры приема.

Регистр	Возвращаемые значения	
Регистр наличия принятых данных Dallas	1	= данные от ведомого устройства успешно получены и доступно в регистрах чтения драйвера Dallas;
	0	= в буфере приема драйвера отсутствуют актуальные данные.

Прием в режиме CZIP / DS1990A.

Регистр	Возвращаемые значения
Регистр чтения 64-битного ROM-кода устройства Dallas SN0:FC	0...0xFFFF = значение регистра: младшая часть содержит идентификатор семейства ключа (FC), старшая часть содержит первый байт уникального адреса ключа (SN0).

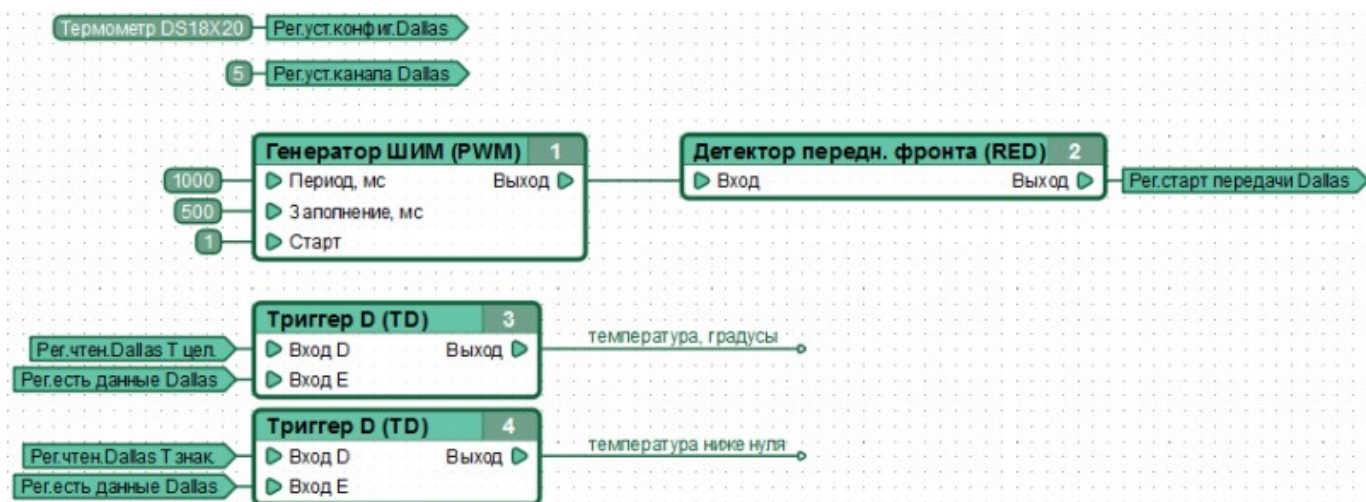
Регистр чтения 64-битного ROM-кода устройства Dallas SN2:SN1	0...0xFFFF = значение регистра: младшая часть содержит второй байт уникального адреса ключа (SN1), старшая часть содержит третий байт уникального адреса ключа (SN2).
Регистр чтения 64-битного ROM-кода устройства Dallas SN4:SN3	0...0xFFFF = значение регистра: младшая часть содержит четвертый байт уникального адреса ключа (SN3), старшая часть содержит пятый байт уникального адреса ключа (SN4).
Регистр чтения 64-битного ROM-кода устройства Dallas CRC:SN5	0...0xFFFF = значение регистра: младшая часть содержит шестой байт уникального адреса ключа (SN5), старшая часть содержит контрольную сумму ROM-кода ключа (CRC).

Прием в режиме DS18X20.

Регистр	Возвращаемые значения
Регистр чтения показаний температуры: целые градусы Цельсия	0...125 = модуль целой части значения измеренной датчиком температуры.
Регистр чтения показаний температуры: десятитысячные доли градуса Цельсия	0...9999 = модуль десятичной части значения измеренной датчиком температуры.
Регистр чтения показаний температуры: знак (0 = выше нуля; 1 = ниже нуля)	0 = измеренная датчиком температура не ниже нуля; 1 = измеренная датчиком температура ниже нуля.

*Примечание: При приеме в режиме DS18X20 период опроса датчика температуры, т. е. отправки ему запросов с помощью регистра начала передачи Dallas, должен быть не менее 750мс.*

Пример функциональной диаграммы работы с термодатчиком DS18X20.



Опрос термодатчика, подключенного к каналу №5 контроллера, осуществляется 1 раз в секунду. При получении данных от DS18X20, в регистре наличия принятых данных драйвера Dallas появляется значение «1» и данные из регистров чтения показаний температуры, с помощью D-триггеров, сохраняются в соответствующие именованные сети для дальнейшей обработки.

# 11 CANNY 7, Параметры пользовательской конфигурации

## 11.1 Общее описание

Параметры пользовательской конфигурации могут быть заданы конечным пользователем контроллера в момент загрузки в него программного обеспечения с использованием Исполняемого файла автономной загрузки ПО в контроллер. После загрузки ПО и запуска контроллера в автономном режиме, установленные пользователем таким образом данные, становятся доступны функциональной диаграмме в соответствующих регистрах контроллера.

Грамотное использование пользовательских параметров существенно повышает гибкость и универсальность решений на базе контроллера, позволяя конечному пользователю, не имеющему навыков работы с CannyLab, вносить безопасные изменения в работу алгоритма контроллера используя простой пользовательский интерфейс.

## 11.2 Регистры параметров пользовательской конфигурации

Возможно задать до 16 пользовательских параметров, которые будут доступны в 16 соответствующих регистрах контроллера.

Регистр	Ожидаемые значения
Регистр параметра пользовательской конфигурации №0 ... Регистр параметра пользовательской конфигурации №15	0...65535 = значение соответствующего пользовательского параметра.

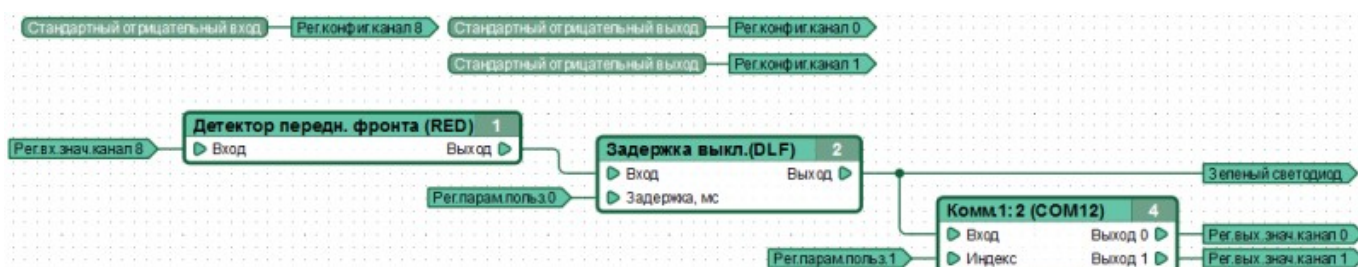
Значения в данных регистрах установятся при старте контроллера, после записи в него ПО посредством исполняемого файла автономной загрузки, и будут оставаться неизменными (константными) на протяжении всего времени работы функциональной диаграммы, не изменяясь даже при сбросе контроллера. Изменить значения данных регистров можно лишь стерев или перезаписав память контроллера новым ПО.

Значения регистров соответствующих параметрам не перечисленным в исполняемом файле автономной устанавливается равным нулю.

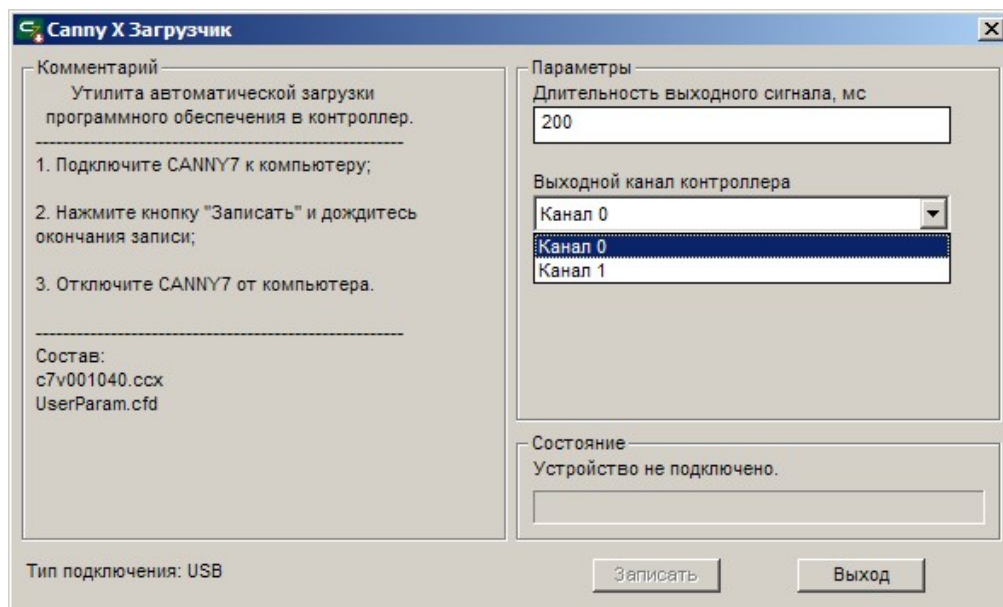
*Примечание: При записи контроллера из среды CannyLab значения всех регистров параметров пользовательской конфигурации устанавливаются равным нулю.*

## 11.3 Пример использования параметров пользовательской конфигурации

Создавая в среде CannyLab исполняемый файл автономной загрузки ПО в контроллер, указав файл системного ПО контроллера и файл, содержащий приведенную ниже диаграмму, задайте два пользовательских параметра: Имя «Длительность выходного сигнала,мс», Тип «Число» и Имя «Выходной канал контроллера», Тип «Список». В список значений параметра «Выходной канал контроллера» добавьте две строки: Название «Канал 0», Значение «0» и «Канал 1», Значение «1».



Запустите созданный таким образом исполняемый файл автономной загрузки ПО, установите требуемые значения параметров и запишите ПО в контроллер.



Выполняя диаграмму контроллер, в момент получения на входе канала №8 отрицательного потенциала, устанавливает на заданном пользователем в соответствующем параметре канале потенциал «GND» и удерживает его заданное пользователем время. Для наглядности, в диаграмме реализована индикация состояния выходного канала контрольным светодиодом.



# 12 CANNY 7, Энергонезависимая память (ЭНП)

## 12.1 Общее описание

Для исключения потери критически важной информации (состояния контроллера, состояния внешних устройств и т. п.) при сбросе питания, в контроллере CANNY7 предусмотрено наличие энергонезависимой памяти. Сохраненные в ней значения будут доступны после восстановления питания контроллера в специальных регистрах.

Пользователю доступны 64 шестнадцатибитные ячейки энергонезависимой памяти, доступ к которым осуществляется с помощью соответствующих регистров чтения и записи.

*Примечание: Работа с энергонезависимой памятью не требует какой-либо специальной предварительной конфигурации.*

## 12.2 Регистры энергонезависимой памяти

Ниже приведено описание допустимых значений регистров установки энергонезависимой памяти контроллера. Они используются для сохранения информации в ячейках ЭНП.

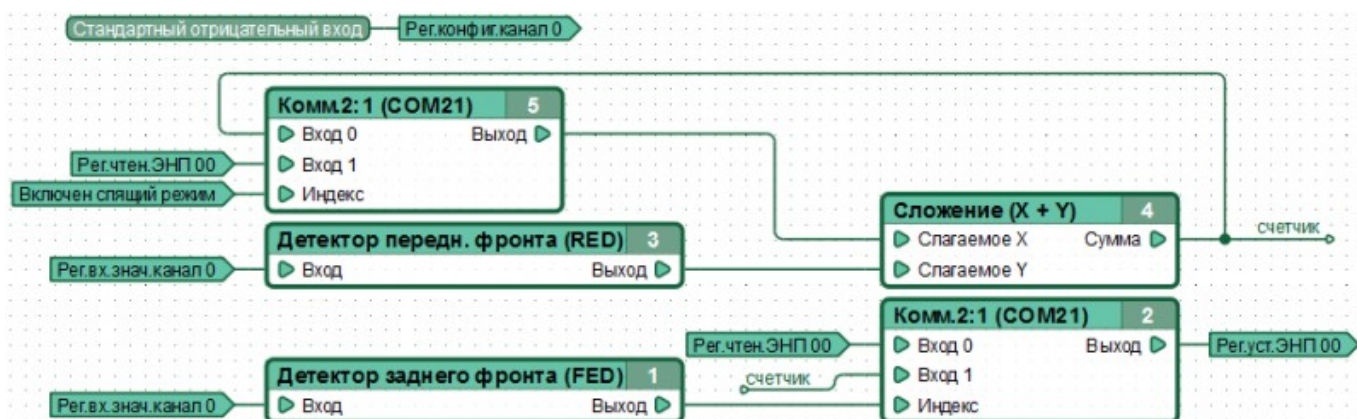
Регистр	Ожидаемые значения
Регистр установки энергонезависимой памяти №00 ... Регистр установки энергонезависимой памяти №63	0...65535 = сохраняемое значение.

Ниже приведено описание допустимых значений регистров чтения энергонезависимой памяти контроллера. Они используются для сохранения информации в ячейках ЭНП.

Регистр	Возвращаемые значения
Регистр чтения энергонезависимой памяти №00 ... Регистр чтения энергонезависимой памяти	0...65535 = хранимое значение.



Пример функциональной диаграммы работы с ячейками энергонезависимой памяти.



Количество нажатий кнопки, подключенной к каналу №0 контроллера, суммируется с ранее сохраненными в именованной сети «счетчик» значениями. Значение сети «счетчик» сохраняется в энергонезависимой памяти при отпускании кнопки. После выключения и восстановления питания контроллера, сохраненное в ячейке энергонезависимой памяти значение автоматически читается и передается обратно в именованную сеть «счетчик». Таким образом удастся избежать потери информации о количестве нажатий на данную кнопку при отключении питания контроллера.

*Примечание: Процесс сохранения данных в ЭНП требует времени, т. е. не происходит мгновенно.*

*Примечание: Количество циклов перезаписи информации в энергонезависимой памяти ограничено. Драйвер работы с памятью CANNY7 организован таким образом, что ее ресурс существенно увеличен. Тем не менее, избегайте постоянного сохранения в ЭНП ненужных данных или сохранения данных на каждом цикле выполнения диаграммы, выполняйте сохранение информации по определенному условию (смотри пример выше).*

# 13 CANNY 7, Драйвер пульта ИК ДУ

## 13.1 Общее описание

Контроллер CANNY7 позволяет принимать и передавать команды инфракрасных пультов дистанционного управления (ИК ДУ) в широко распространенных форматах NEC, extended NEC и Samsung. Работа драйвера возможна в трех режимах: только прием, только передача или прием/передача, в том числе в разных форматах. Для приема и передачи используются два любых канала контроллера.

При передаче команд ИК ДУ, используемый для этого канал контроллера CANNY7 выдает только модулирующий сигнал. Для формирования пакетов импульсов контроллеру требуется наличие несущей частоты, источником которой может выступать как один из каналов ВЧ ШИМ CANNY7, так и внешний генератор ШИМ.

Прием команд ИК ДУ требует наличия внешнего демодулятора, например TSOP1736 или аналогичного.

Драйвер пульта ИК ДУ в своей работе использует ресурсы Драйвера каналов ввода-вывода контроллера, но имеет более высокий приоритет чем драйвера ввода-вывода. Таким образом, при активации драйвера пульта ИК ДУ, для задействованных в его работе каналов, изменение значений в связанных с ними регистрах драйвера ввода-вывода будет проигнорировано контроллером.

## 13.2 Регистры драйвера пульта ИК ДУ

Для работы контроллера CANNY7 с приемниками и передатчиками ИК-сигналов может использоваться любой из его каналов ввода-вывода, при этом тот или иной канал контроллера, в каждый отдельный момент, может работать либо только на прием, либо только на передачу. Драйвер пульта ИК ДУ предусматривает возможность организации одновременного приема и передачи данных по двум независимым каналам. Кроме того, возможно подключение к CANNY7 нескольких приемников/передатчиков ИК-сигналов.

Регистры конфигурации драйвера пульта ИК ДУ.

Регистр	Ожидаемые значения
Регистр конфигурации ИК-порта	1...N = установить конфигурацию канала драйвера пульта ИК ДУ контроллера, определяющую текущий режим и параметры его работы (задается специальной константой из справочника)

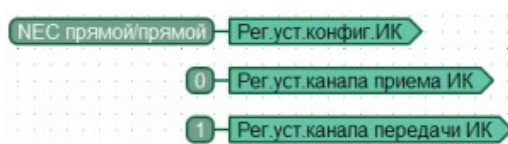
	0	констант); = отключить канал от драйвера пульта ИК ДУ, вернуть управление каналом драйверу каналов ввода-вывода и разрешить изменения его состояния из функциональной диаграммы.
Регистр номера канала приема ИК-порта	0...10	= установить номер канала контроллера, используемый драйвером пульта ИК ДУ для приема данных.
Регистр номера канала передачи ИК-порта	0...10	= установить номер канала контроллера, используемый драйвером пульта ИК ДУ для передачи данных.

*Примечание: В конфигурации «Прием и передача» для приема и передачи данных должны быть назначены отдельные каналы контроллера.*

Конфигурация драйвера задается именованными константами, представляющими комбинацию параметров, определяющих тип сигнала, определяющим электрические потенциалы исходного состояния канала и состояния канала при передаче данных, наличие и потенциал внутренней «подтяжки» канала контроллера.

Параметр	Перечень допустимых значений
Стандарт	NEC, Samsung
Направление потока данных	Прием, передача, прием и передача
Тип сигнала	Прямой («1»: плюс, «0»: минус), инверсный («1»: минус, «0»: плюс)
Внутренняя подтяжка	Без подтяжки, подтяжка плюс, подтяжка минус

Пример конфигурации драйвера пульта ИК ДУ контроллера для работы в качестве приемника/передатчика ИК-сигналов, при этом для приема данных используется канал №0, а для передачи - канал №1.



Регистры диагностики драйвера пульта ИК ДУ.

Регистр	Возвращаемые значения
Регистр отсутствия активности драйвера ИК	1 = активность драйвера ИК ДУ на соответствующем канале отсутствует; 0 = зафиксирована активность драйвера ИК ДУ на соответствующем канале.

Регистр переполнения буфера драйвера ИК	1	= буфер соответствующего канала драйвера пульта ИК ДУ переполнен;
	0	= переполнения соответствующего канала драйвера пульта ИК ДУ не зафиксировано.
Регистр готовности буфера передачи ИК	1	= буфер передачи данных драйвера пульта ИК ДУ свободен и готов к работе;
	0	= буфер передачи данных драйвера пульта ИК ДУ занят.

Регистры приема драйвера пульта ИК ДУ.

Регистр	Возвращаемые значения	
Регистр наличия принятых данных ИК	1	= сообщение успешно получено и доступно в регистрах буфера приема драйвера пульта ИК ДУ;
	0	= в буфере приема драйвера пульта ИК ДУ отсутствуют актуальные данные.
Регистр принятого сообщения ИК D1:D0 Регистр принятого сообщения ИК D3:D2	0...0xFFFF = значения соответствующих байт данных приемного буфера драйвера пульта ИК ДУ, по два байта на регистр.	

Регистры передачи драйвера пульта ИК ДУ.

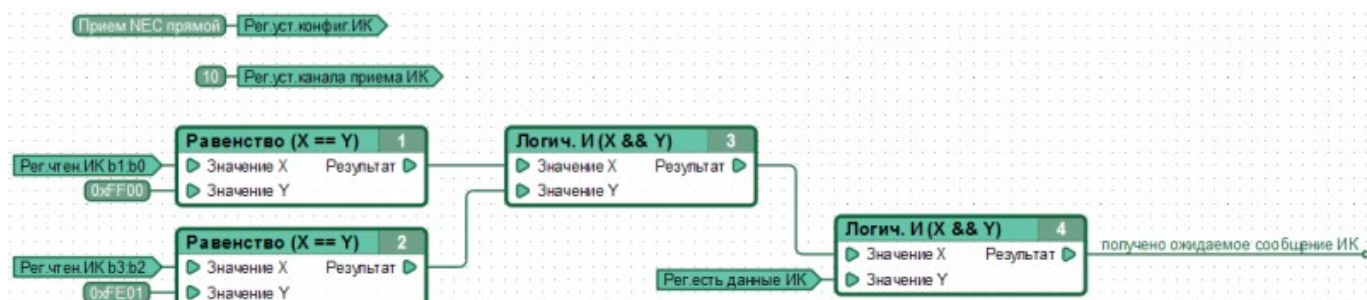
Регистр	Ожидаемые значения	
Регистр начала передачи ИК	$\geq 1$	= загрузить данные из регистров передачи в буфер передачи драйвера пульта ИК ДУ;
	0	= не загружать данные в буфер передачи драйвера пульта ИК ДУ.
Регистр сообщения передачи ИК D1:D0 Регистр сообщения передачи ИК D3:D2	0...0xFFFF = значения передаваемых байт сообщения драйвера пульта ИК ДУ, по два байта на регистр.	

Специальная команда стандарта NEC – «повтор команды», кодируется значениями D1:D0 = 0xFFFF и D3:D2 = 0xFFFF как при приеме так и при передаче.

*Примечание: При отправке ИК-сообщений, каждая последующая команда должна передаваться только после освобождения буфера передачи, т. е. при наличии в регистре готовности буфера передачи ИК-сообщения значения «1». Команда повтора предыдущей команды должна отправляться сразу после отправки основной команды, не дожидаясь освобождения буфера*

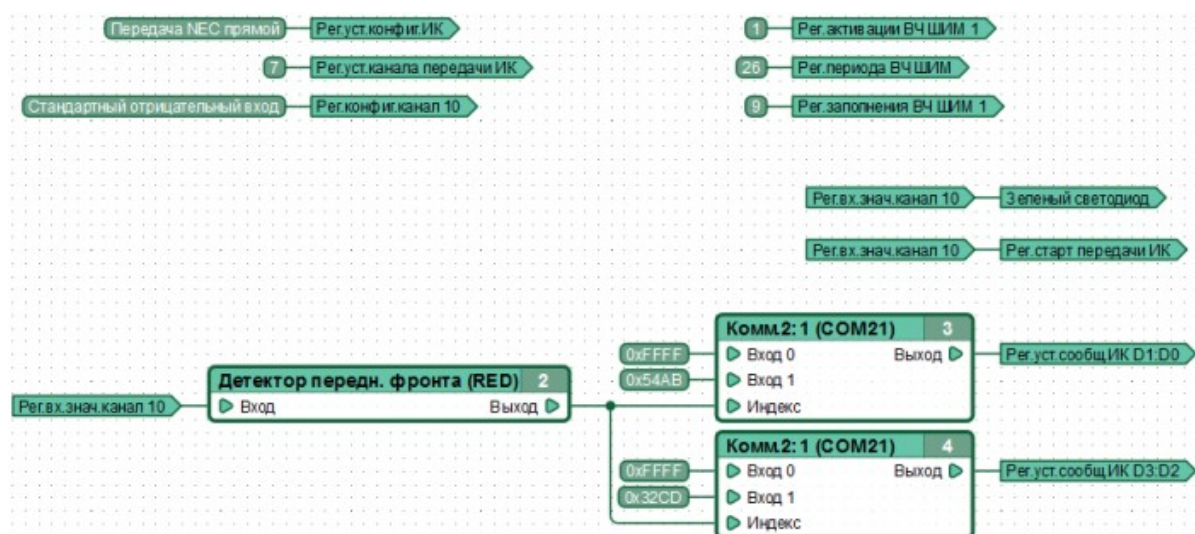
передачи, при этом отправка должна выполняться непрерывно, как можно чаще, все время, пока требуется передача подтверждения.

Пример функциональной диаграммы приема контроллером команд от внешнего ИК-пульта ДУ.



Контроллер ожидает получения сообщения, содержащего номер адреса 0x00 и команду с кодом 0x01. Проверка полученных от пульта ИК ДУ команд на соответствие ожидаемым выполняется путем их сравнения с константами, содержащими в младшем байте требуемое значение, а в старшем — его инверсную версию.

Пример функциональной диаграммы эмуляции контроллером CANNY7 ИК-пульта ДУ, т.е. передачи ИК-команд управляемому устройству.



При появлении на входе канала №10 значения «1» (нажатия управляющей кнопки), контроллер отправляет ИК-сообщение внешнему устройству. В сообщении содержится адрес 0xAB и команда 0xCD. При длительном сохранении значения «1» на входе канала №10 (удержании кнопки) контроллер отправляет сообщение стандарта NEC «повтор команды». Специальная команда стандарта NEC — «повтор команды», кодируется значениями D1:D0 = 0xFFFF и D3:D2 = 0xFFFF как при приеме так и при передаче.

*Примечание: В примере отправки ИК-сообщения также реализовано использование CANNY7 для формирования несущего высокочастотного сигнала установленного стандартом NEC, с помощью канала №1 контроллера*

*работающего в режиме ВЧ ШИМ с периодом 26мкс и заполнением 9мкс.*