УТВЕРЖДАЮ

Генеральный директор

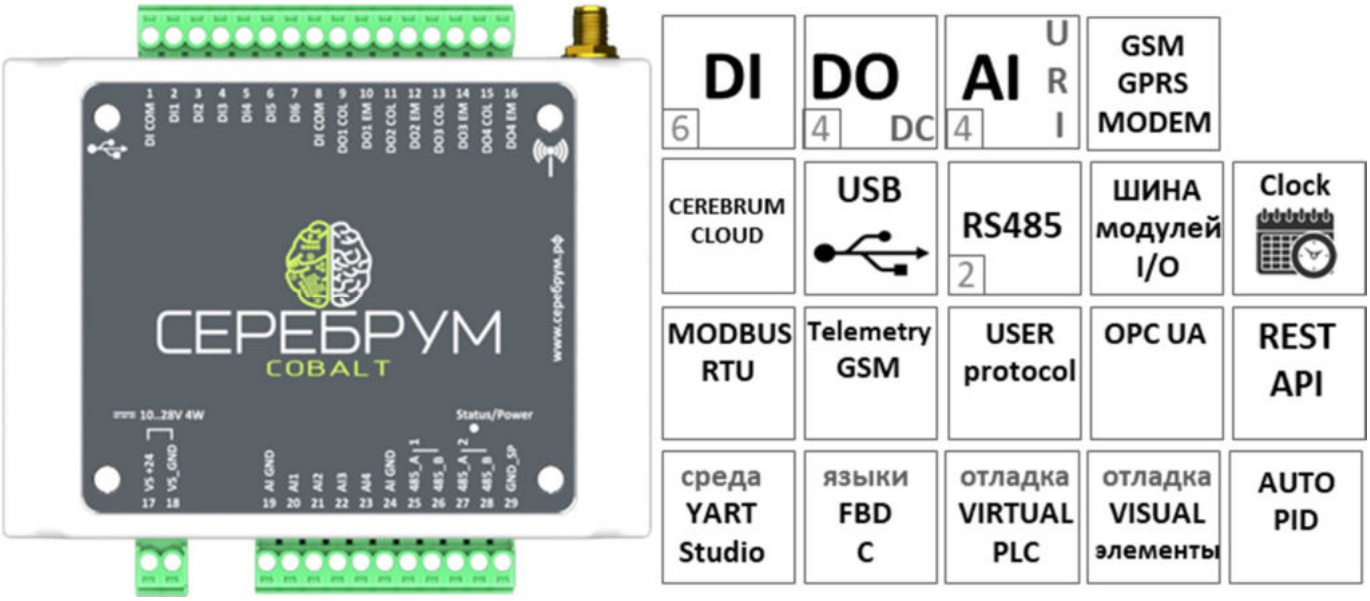
ООО «ТД «СЕРЕБРУМ»

Т.Ю. Селенкина

Программируемый логический контроллер

СЕРЕБРУМ COBALT

Техническое руководство



Санкт-Петербург

2020

Оглавление

2. Предупреждения

3. Список изменений

4. Общая информация

Контроллер COBALT — промышленный логический контроллер (ПЛК) со встроенным GPRS модемом системой удаленного мониторинга и управления СЕРЕБРУМ.

COBALT предназначен для использования в качестве промышленного контроллера в локальных и распределенных системах управления с возможностью подключения удаленных модулей ввода/вывода через Modbus RTU.

Встроенный GPRS модем позволяет интегрировать COBALT с системой телеметрии СЕРЕБРУМ без необходимости подключения дополнительного оборудования.

Программирование и настройка контроллера выполняется в среде Yart Studio, соединение с которой осуществляется через встроенный USB порт.

COBALT поддерживает одновременную работу с серверами телеметрии СЕРЕБРУМ, ModbusRTU Master/Slave и до семи подключенных модулей расширения через шину YART-BUS.

Встроенное программное обеспечение контроллера автоматически устанавливает и поддерживает Интернет соединение с серверами удаленного мониторинга и управления.

5. Технические характеристики

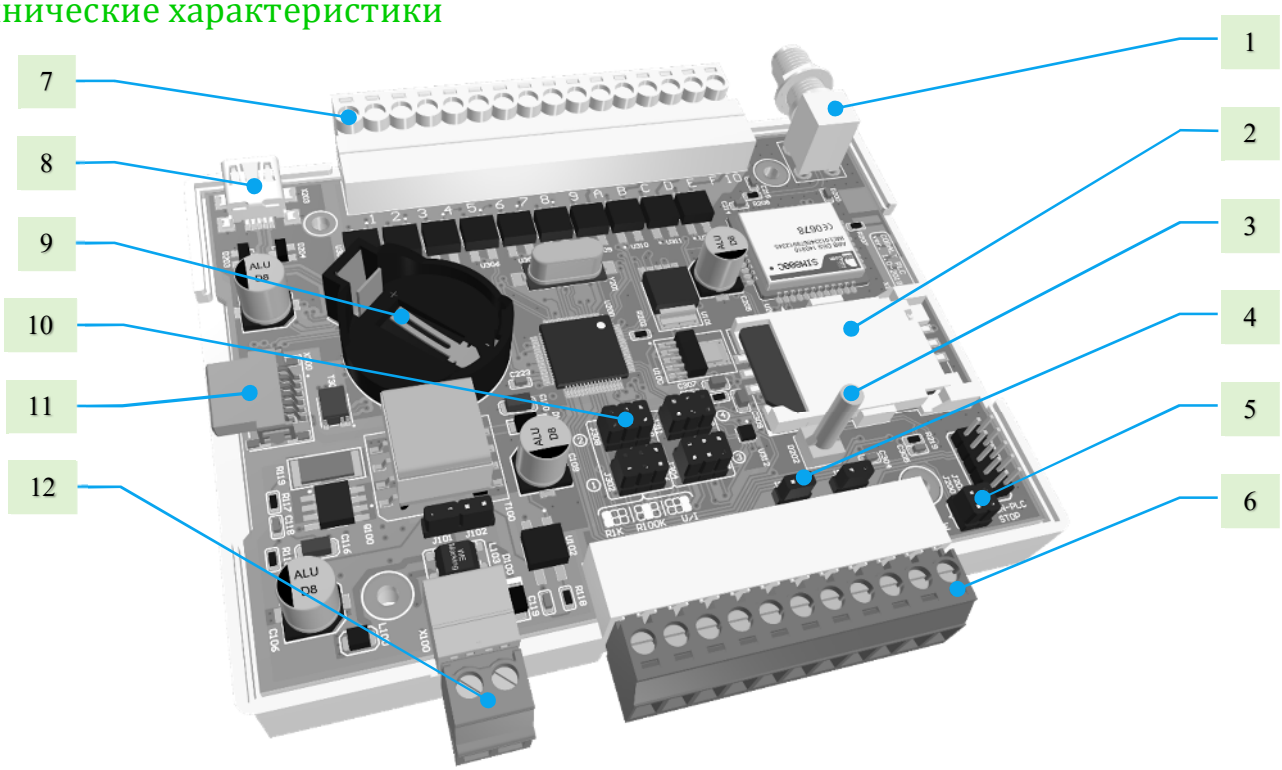


Рисунок 1. Плата контроллера COBALT

1. Разъем подключения антенны GSM
2. Слот SIM карты
3. Статусный индикатор
4. Переключатели «терминаторов» RS-485
5. Переключатели принудительной остановки и обновления СПО контроллера
6. Клеммник 3
7. Клеммник 2
8. Разъем USB для подключения к ПК
9. Держатель батареи
10. Переключатели режимов аналоговых входов
11. Разъем шины YART BUS
12. Клеммник 1

Таблица 1. Технические характеристики COBALT

Параметр		Характеристики
Напряжение питания		от 10 до 28 VDC
Потребляемая мощность, Вт		Не более 3Вт
Габариты, мм		100x103x20
Степень защиты корпуса по ГОСТ 14254-96		IP20
Подключение антенны		SMA
Тип разъемов клеммника		Разъемные, под винт, максимальное сечение провода 1.5 мм2
Программирование		ФБД, C-YART, среда программирования YART Studio
Память	Программ	384 KB энергонезависимая память (более 500 типов блоков, 5000 вызовов блоков)
	RAM	32 KB память для переменных пользователя
	FRAM	7 KB энергонезависимая память для переменных пользователя
	BRAM	отсутствует
	RAM BIT	2 KBIT память для переменных пользователя
	FRAM BIT	2 KBIT энергонезависимая память для переменных пользователя
uSD карта памяти		отсутствует

Выполнения цикла программы		от 1 ms
Входы	Дискретные	6DI x 7-28 VDC, гальванически изолированы от цепей процессора
	Аналоговые	4(2)AI x 16 бит. Режимы - U/R/DI (0-10VDC/0-400kOm).
Выходы	Дискретные	4DO x оптотранзистор, до 0,1 А нагрузка, до 30VDC. Гальванически изолированы от цепей процессора и между собой
	Аналоговые	отсутствуют
Коммуникационные порты	mini-USB	MODBUS RTU, порт отладки
	RS485-1	MODBUS RTU, протоколы библиотеки, пользовательский, гальванически изолирован от питания
	RS485-2	MODBUS RTU, протоколы библиотеки, пользовательский, гальванически изолирован от питания
	YART-PORT	Шина модулей расширения, период опроса всех модулей менее 30 ms, максимальное число подключаемых модулей – 7
Часы реального времени (RTC)		С питанием от батарейки (точность хода при 25 ºС – не более ±2 с в сутки)
Диапазон температур эксплуатации		-40°С ... 55°С
GPRS модем	Работа в сетях	850/900/1800/1900 МГц, GPRS B multi-slot 12/10
	Выходная мощность	– Класс 4 (2 Вт @ 850/900 МГц) – Класс 1 (1 Вт @ 1800/1900 МГц)

	Скорость обмена	12-й класс GPRS: до 85.6 кбит/сек (скачивание/выгрузка)
	Сертификаты	CE, FCC
SMS		есть

6. Подключение

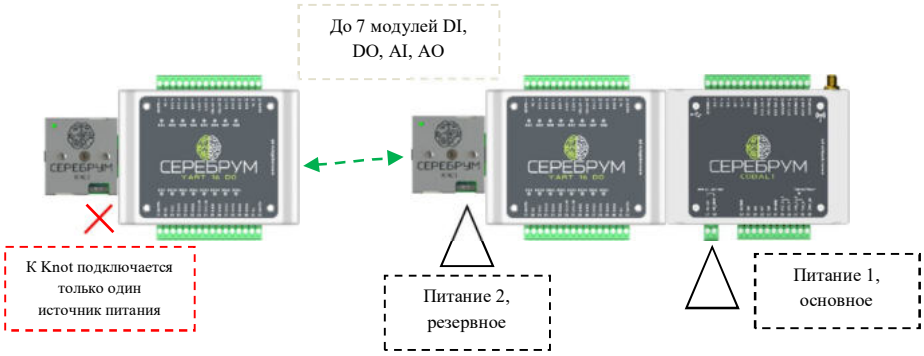


Рисунок 2. Подключение источников питания

Для работы контроллера требуется наличие внешнего питания (см. **Таблицу №1**). Линия питания выведена на клеммы 17-19 клеммника 1, а также на выводы разъема подключения модулей расширения – YART-BUS.

На **Рисунке 2** показана схема питания контроллера.

Основной источник питания подключается к клеммнику 1. Модули расширения, подключенные к COBALT будут работать от основного источника питания через шину расширения YART-BUS.

При необходимости организации резервного питания или при отсутствии возможности подключения к клеммнику №1 питание может осуществляться через модуль Knot, подключаемый к шине расширения ПЛК.

Место подключения не регламентируется.

! Внимание

При одновременном использовании нескольких модулей Knot внешнее питание должно подключаться только к одному из них.

Таблица 3. Сигналы клеммника 1

Номер клеммы	Наименование	Назначение
17	VS+24	Питание контроллера, положительный потенциал
18	VS_GND	Питание контроллера, общий

Таблица 4. Сигналы клеммника 2

Номер клеммы	Наименование	Назначение
1	DI-COM	Общий сигнал дискретных входов
2	DI1	Дискретный вход
3	DI2	Дискретный вход
4	DI3	Дискретный вход
5	DI4	Дискретный вход
6	DI5	Дискретный вход
7	DI6	Дискретный вход
8	DI-COM	Общий сигнал дискретных входов
9	DO1-COL	Дискретный выход, коллектор
10	DO1-EM	Дискретный выход, эмиттер
11	DO2-COL	Дискретный выход, коллектор
12	DO2-EM	Дискретный выход, эмиттер
13	DO3-COL	Дискретный выход, коллектор
14	DO3-EM	Дискретный выход, эмиттер
15	DO4-COL	Дискретный выход, коллектор
16	DO4-EM	Дискретный выход, эмиттер

Таблица 3. Сигналы клеммника 3

Номер клеммы	Наименование	Назначение
19	AI GND	Общий сигнал аналоговых входов
20	AI1	Аналоговый вход
21	AI2	Аналоговый вход
22	AI3	Аналоговый вход
23	AI4	Аналоговый вход
24	AI GND	Общий сигнал аналоговых входов
25	485_A	Порт RS-485-(COM2) Линия A
26	485_B	Порт RS-485-(COM2) Линия B
27	485_A	Порт RS-485-(COM3) Линия A
28	485_B	Порт RS-485-(COM3) Линия B
29	GND_SP	Аналоговый вход 6 (U, I R)

7. Начало работы

Обмен данными между ПК и COBALT осуществляется через встроенный USB порт. В составе установочного пакета Yart Studio включены необходимые драйвера для устойчивой работы порта.

Порт USB гальванически развязан от входного питания контроллера. Однако в установленном контроллере ряд цепей (аналоговый вход, последовательные интерфейсы) могут быть гальванически связаны с важными силовыми цепями установки. Перед подключением убедитесь, что питание ПЛК\ПК имеет гальваническую развязку от основной сети питания. При необходимости используйте специализированные адаптеры USB.

8. Использование модулей расширения

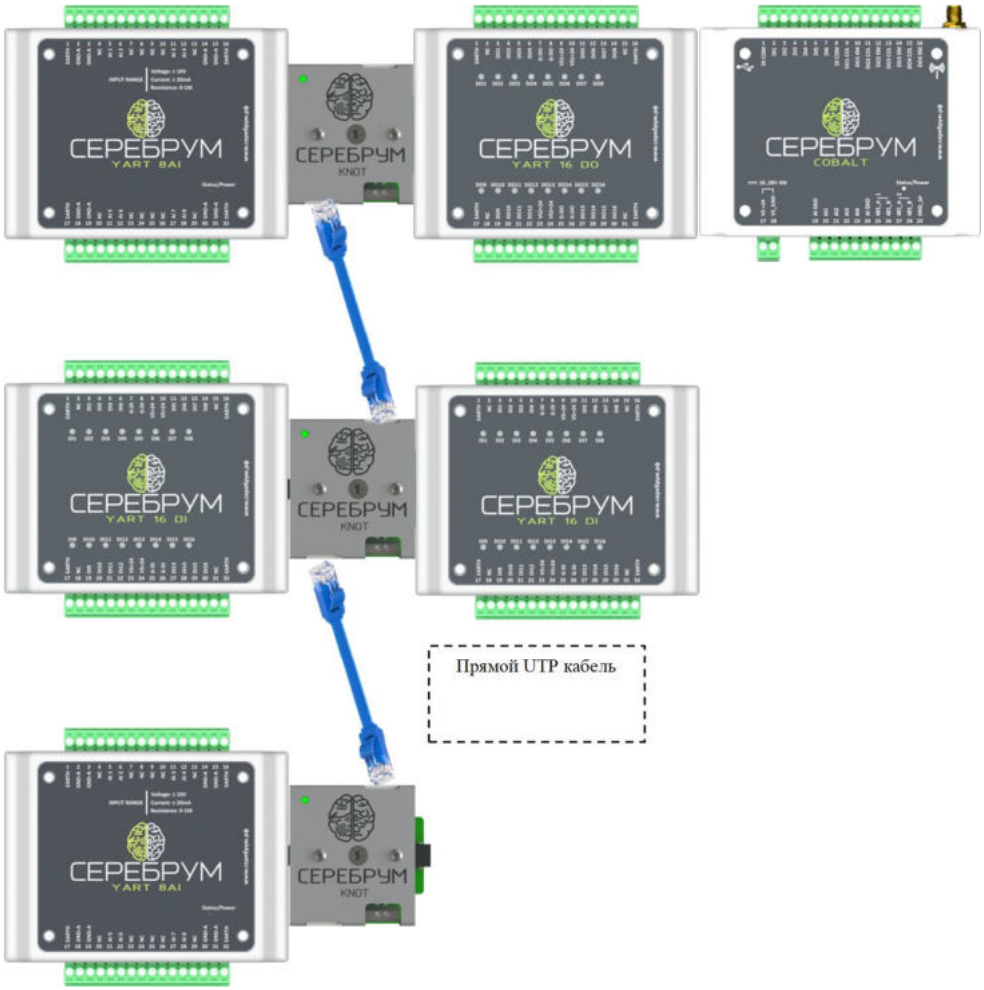


Рисунок 5. Возможная топология шины расширения

Пример возможной конфигурации шины YART-BUS представлен на **Рисунке 5**.

При проектировании конфигурации шины необходимо учитывать следующие правила:

- Контроллер COBALT поддерживает до семи различных модулей расширения на одной шине
- Суммарная длина всех соединений не должна превышать 5 метров
- На оконечном модуле должен быть включен согласующий «терминатор»
- Учитывайте падение напряжения между модулями. В среднем на каждом модуле теряется 0.1В
- При низком значении напряжения питания удаленный модуль может отключиться
- Диапазон разрешенных адресов для конфигурации модулей расширения 1 – 7
- Порядок адресации и подключения модулей может быть любым

Программная конфигурация модулей осуществляется при программе YartStudio.

При каждом запуске контроллер осуществляет сканирование и идентификацию всех возможных модулей. В процессе работы найденные модули опрашиваются, а результат работы сохраняется в специализированных регистрах процессора.

На рисунке ниже показан пример автоматически определенных модулей.

Рисунок 6. Список определенных модулей

COBALT может работать в двух режимах опроса модулей расширения:

- В ***безопасном режиме работы*** любая ошибка опроса модулей расширения вызовет аварийную остановку прикладной программы. При этом все выходы модулей будут переведены в безопасное состояние (нуль).
- В ***обычном режиме работы*** ошибка обмена с модулем расширения не вызывает остановку выполнения прикладной программы, но состояние ошибки фиксируется в соответствующем регистре процессора для анализа и диагностики.

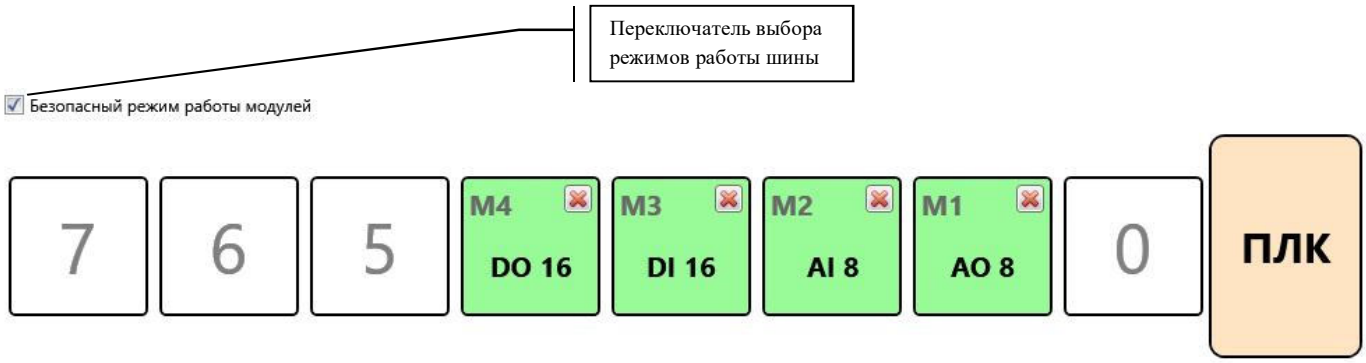


Рисунок 7. Конфигурация модулей расширения

Выбор режима работы шины расширения осуществляется в окне настроек проекта – модули расширения.

Каждый модуль расширения должен иметь уникальный адрес, который назначается вращающимся переключателем, расположенным на плате модуля.

Для COBALT доступны адреса модулей в диапазоне 1 – 7.

Пример модуля расширения и расположение переключателей показан на **Рисунке 8**.

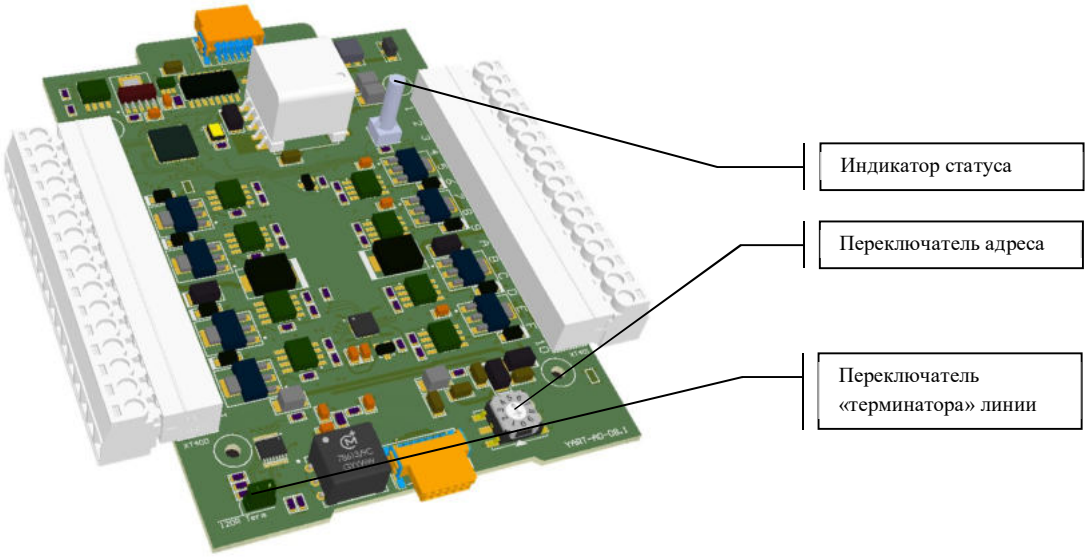


Рисунок 8. Переключатели модулей расширения

Перед началом работы убедитесь, что переключатели адресов модулей расширения установлены в правильные позиции, совпадений между ними нет.

В процессе работы в окне статуса модулей расширения Yart Studio будет отображаться текущее состояние шины расширения.

Для успешной работы отображаемое состояние модулей должно в точности соответствовать проектному.

Каждый модуль расширения оснащается статусным индикатором. Для исправного модуля расширения данный индикатор должен всегда быть включен.

9. Индикация состояний

Отображение состояния ПЛК COBALT осуществляется при помощи сдвоенного светодиодного индикатора (**поз. 3 Рис. 1**).

Красный светодиод подключен к GPRS модулю, зеленый к ПЛК.

Различные режимы отображения представлены в **Таблице 7**.

Таблица 7. Состояния встроенного индикатора

№	Режим индикатора	Состояние ПЛК
1	Горит зеленым	Выполнение прикладной программы (режим Run)
2	Мигает зеленым (t = 1 сек)	Выполнение прикладной программы остановлено
3	Мигает зеленым (t = 0.25 сек)	Аварийная остановка прикладной программы
4	Красный не горит	Модем отключен
5	Красный: 64мс включен; 800 мс - нет	Регистрация в сети GSM
6	Красный: 64мс включен; 3с - нет	Модем зарегистрирован в сети
7	Красный: 64мс включен; 300 мс - нет	Установлено подключение к сети Интернет

10. ПЛК, особенности архитектуры и память

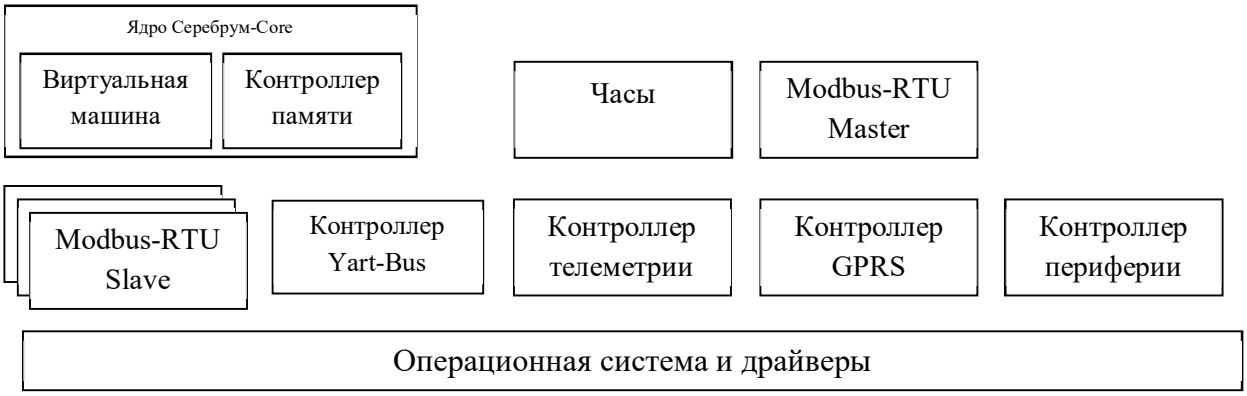


Рисунок 9. Архитектура контроллера COBALT

В основе вычислительного ядра контроллера лежит система ***Серебрум-Core***, состоящая из виртуальной машины и контроллера виртуальной памяти.

Все взаимодействие между модулями осуществляется через общее пространство памяти, в которой определены участки периферии, данных пользователя и настроек. Более подробно структура памяти рассматривается ниже.

Независимо от виртуальной машины ПЛК выполняет задачи обработки внешних интерфейсов, приёма и передачи данных, а также поддерживает функционирование встроенного сервера телеметрии.

При изменении настроек загруженного проекта все сопутствующие программы автоматически перенастраиваются.

Часть программных модулей настраиваются и обрабатываются непосредственно из программы пользователя. Например, работа программ обмена по Modbus конфигурируется через специальный API, что ускоряет и упрощает процесс настройки.

COBALT содержит встроенные периферийный модули для непосредственного подключения к линиям датчиков и управления.

10.1 Память программируемого контроллера

Вся память контроллера (**Таблица 8**) условно разделена на две основных части:

- Память программы
- Память данных

В памяти программы хранится исполняемый код виртуальной машины, настройки проекта, список переменных, а также настройки обмена и телеметрии.

При каждом старте контроллера содержимое памяти программы проверяется и, если все поля верны - даётся разрешение на исполнение кода виртуальной машины.

Память данных содержит четыре основных сегмента

10.1.1 Системная память

В системной памяти хранятся все настройки периферии контроллера, а также текущее значение данных, полученных от модулей ввода.

Чтение и запись в системную память осуществляют специальные блоки программы пользователя, обеспечивающие корректную работу с аппаратными ресурсами ПЛК

10.1.2 Оперативная память

Основной блок памяти, предназначенный для хранения данных прикладной программы, настроек программ окружения и данных последовательных портов.

Пространство оперативной памяти автоматически сбрасывается при рестарте контроллера. Исключение составляют лишь переменные, для которых определены начальные значения.

Нижнее пространство оперативной памяти выделено для хранения значений булевых переменных.

Помимо обычного режима доступа (байт, слово, двойное слово) булевые переменные имеют битовый режим доступа.

10.1.3 Энергонезависимая память (FRAM)

FRAM дает даёт возможность сохранения данных программы виртуальной машины при отсутствии основного питания контроллера.

Доступ к FRAM осуществляется аналогично оперативной памяти, однако скорость работы контроллера при этом будет снижена.

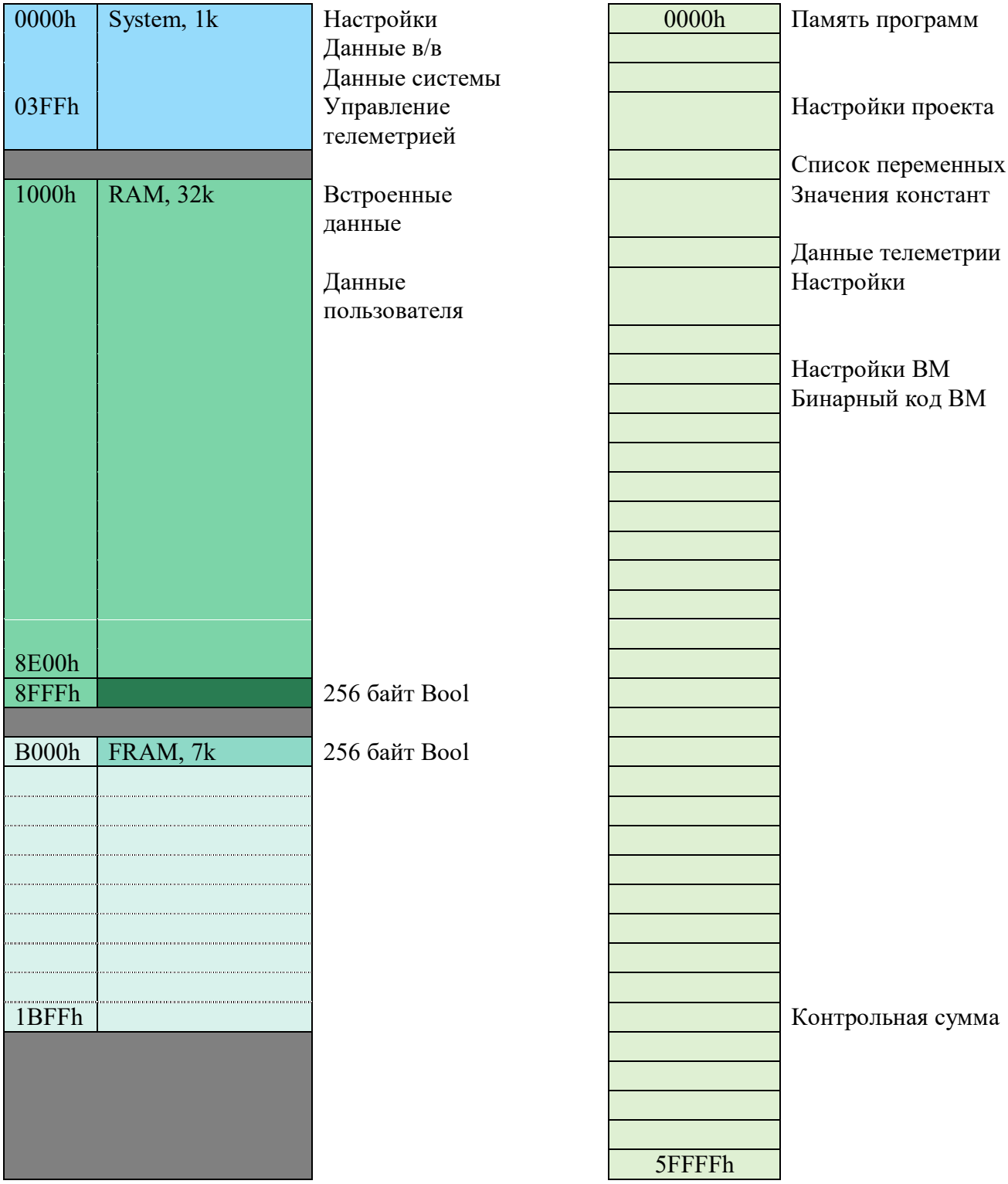
При разработке прикладных программ следует избегать постоянной перезаписи значений FRAM, поскольку контроллеру требуется дополнительное время на индексацию данных энергонезависимой памяти.

Верхние 256 байт FRAM выделены под хранение энергонезависимых булевых переменных аналогично булевым переменным оперативной памяти.

10.1.4 Батарейная память

Данный участок памяти функционально аналогичен оперативной памяти, однако значения хранящиеся в BRAM не сбрасываются при отсутствии основного питания контроллера. Работа BRAM (также к батарее подключен модуль часов) осуществляется при поддержке напряжения от встроенной батареи.

Таблица 8. Карта памяти контроллера



10.2 Типы данных, используемых в системе

- ***Bool*** – Битовая переменная.
  Данные переменные хранятся в Bool сегментах (Таблица 8).
  В C-Yart доступны предопределенные слова «***true***», «***false***» соответствующие значениям «1» и «0» соответственно.
- ***Byte*** – Один байт данных. Беззнаковый тип в диапазоне чисел 0 – 255.
- ***Short*** – Знаковый 16 – разрядный тип данных. Может принимать значения в диапазоне [−32768, +32767]
- ***Int*** - Знаковый 32 – разрядный тип данных. Может принимать значения в диапазоне [−2 147 483 648, +2 147 483 647]
- ***Float*** – вещественные 32-разрядные числа в формате с плавающей точкой (IEEE 754-1985)
- ***Символ*** – ASCII символ в кодировке IBM CP866. Поддерживаются строки длиной до 254 символов
- ***Дата*** – внутренний тип данных в BCD формате даты: DD.MM.YY
  Нумерация лет в рамках диапазона [0, 99]
- ***Время*** – внутренний формат времени в BCD формате HH:MM:SS

11. Телеметрия

Все программируемые контроллеры Серебрум изготавливаются с поддержкой системы удаленного мониторинга и управления. В зависимости от типа используемого оборудования возможно подключение через Ethernet, 2G/3G модем или оба варианта.

Все подключенные контроллеры имеют встроенную возможность передавать через защищенное соединение данные на удаленный сервер, принимать команды и отображать в пространстве программы пользователя данные о состоянии сетевого подключения.

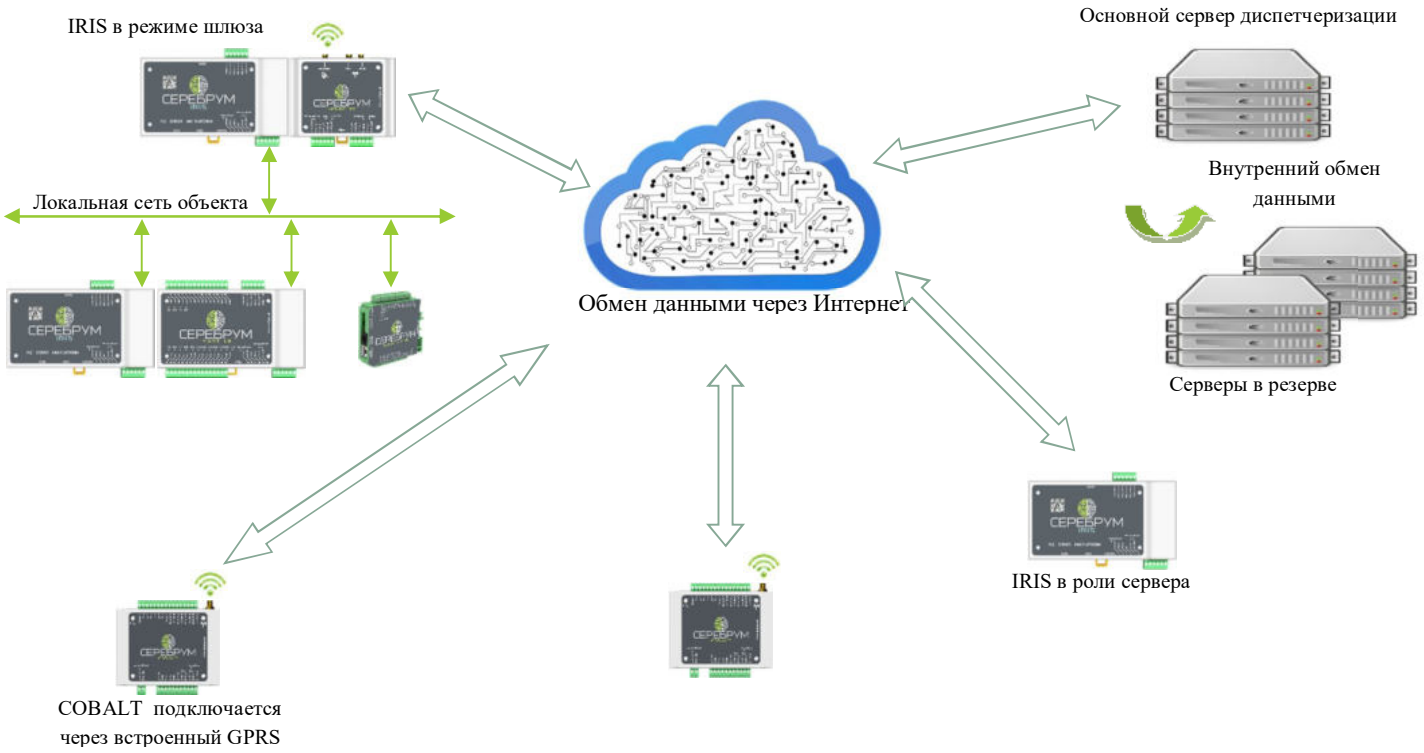


Рисунок 10. Удаленная диспетчеризация и управление Серебрум

Инфраструктура системы телеметрии Серебрум (**Рис. 10**) состоит из выделенного сервера и множества удаленных контроллеров. При необходимости вся система может быть развернута в локальной сети или сети предприятия.

Головная часть состоит из одного или нескольких серверов, которые могут быть связаны друг с другом для обеспечения горячего резервирования данных.

11.1 Подключение к удаленному серверу телеметрии

Контроллер COBALT поддерживает работу в режиме удаленного модуля телеметрии. Подключение к Интернет осуществляется через встроенный GPRS модем.

Для настройки телеметрии необходимо выполнить ряд действий.

Шаг 1. Установите (**Рис. 11**) параметры мобильной сети и настройки соединения с сервером телеметрии.

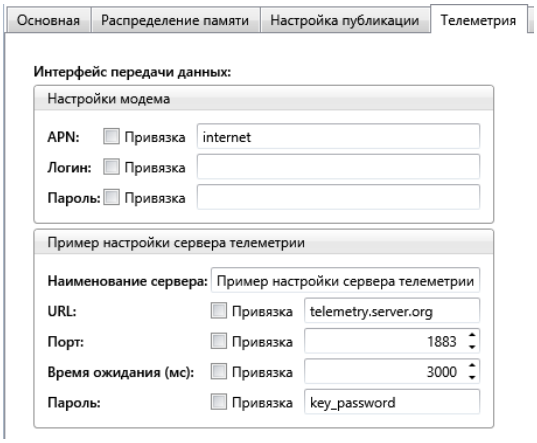


Рисунок 11. Простая настройка телеметрии

В том случае если контроллер еще на был зарегистрирован в системе – оставьте значение поля «Пароль» равным «***hitchhiker***».

В поле URL укажите адрес основного сервера телеметрии.
Порт – 1883, время ожидания – рекомендуется использовать значение 3000

В данном примере показана базовая настройка, не предполагающая смену рабочих серверов при работе контроллера.
Для динамического управления подключениями следует использовать режим привязок.

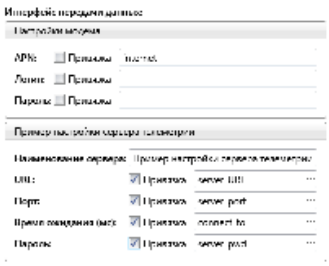


Рисунок 12. Привязка параметров подключения к серверу к переменным

На **Рисунке 12** показан пример использования переменных для настройки подключения к серверу. Значения переменных объявлены заранее, однако в процессе работы их можно изменить и дать команду повторного подключения.

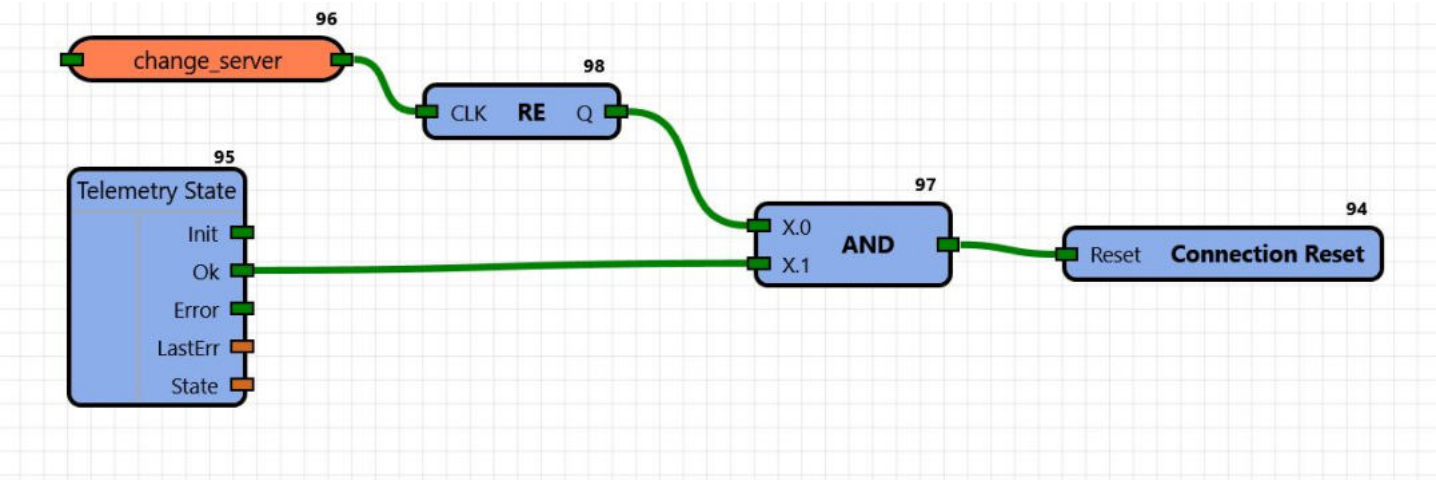


Рисунок 13. Пример команды сброса соединения с сервером

Перед каждой попыткой соединения с сервером программа телеметрии будет считывать значения переменных, указанных в виде ссылок (**Рис. 12**). Если соединение установить не удалось, программа снова считает эти данные и повторит попытку.

Если требуется установить соединение с другим сервером, то в приложении необходимо предусмотреть место, где переменные **server_URL**, **server_pwd**, **server_port** и **connect_to** будут изменены.

Для сброса и обновления данных подключения работающего соединения следует использовать блок “Connection Reset”, **Рисунок 13**.

На этом примере соединение сбросится и новые данные будут скопированы из привязанных переменных при наличии успешного соединения (блок Telemetry State) и нарастающего фронта “change_server”.

Параметры подключения можно изменять удаленно, поместив данные основных и резервных серверов в список публикуемых переменных.

Шаг 2. Определение списков удаленных переменных

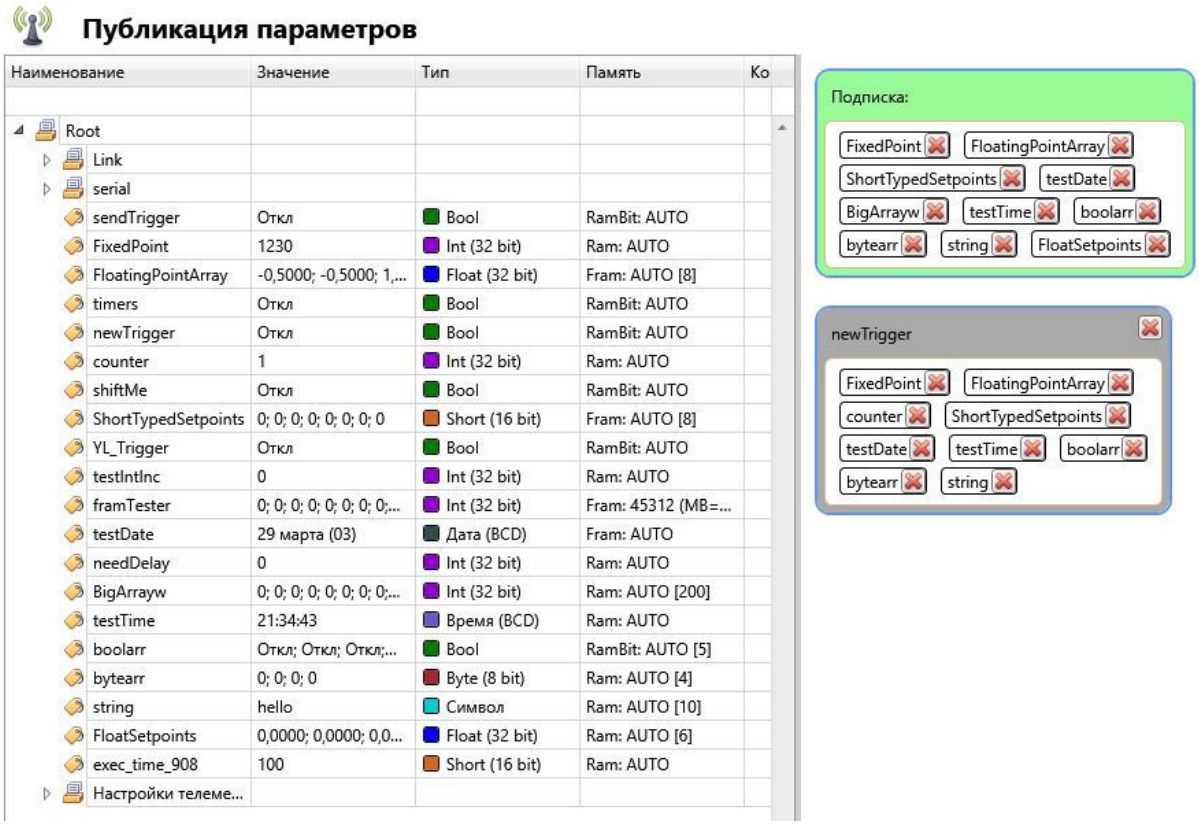


Рисунок 14. Параметры публикации

Все публикуемые переменные условно разделяются на два множества: переменные в подписке и переменные для публикации.

В списке «Подписка» (**Рис. 14**) находятся переменные, которые допускается изменять со стороны сервера – настройки, параметры работы и прочие данные.

<u>Шаг 3.</u> Публикация данных

Отправка данных на сервер осуществляется по флагам передачи. Флагом является любая переменная типа bool, которая объявлена в таком качестве. Для этого требуется перенести нужную переменную на правое поле – появится серый контейнер, куда аналогично можно помещать переменные, значения которых будут отправлены при наличии «1» в бите соответствующего флага.

В примере выше в качестве флага используется переменная “newTrigger”. После того, как программа присвоит данной переменной значение «1» на текущий сервер будут отправлены значения переменных:

FixedPoint, FloatingPointArray, counter, ShortTypedSetpoints, testDate, testTime, boolarr, bytearr, string.

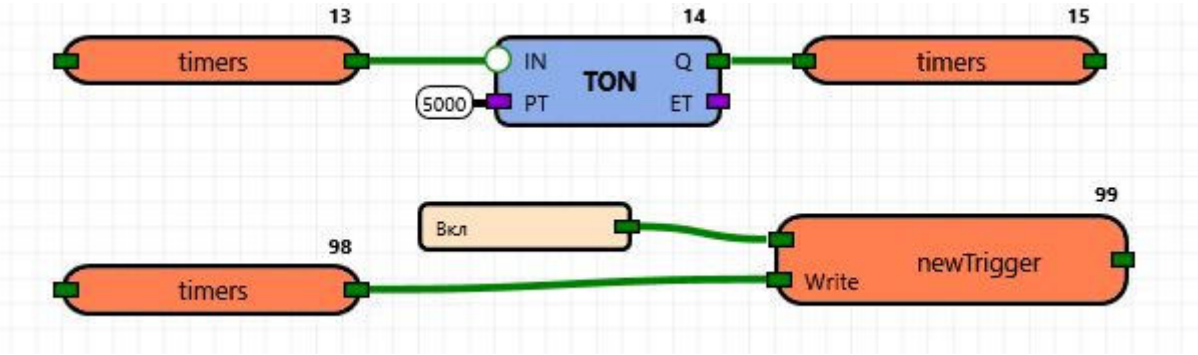


Рисунок 15. Отправка данных на сервер каждые 5 секунд

В качестве примера приложение (**Рис. 15**) отправляет данные на сервер через каждые 5000мс.

Обратите внимание, что переменной newTrigger присваивается только значение «1» - после успешной отправки данных значение переменной-флага автоматически сбросится в «0».

Переменные, перечисленные в списке «Подписка», меняются автоматически, если от сервера пришла команда на их изменение.

В случае, когда контроллеру необходимо «знать» о изменении тех или иных переменных в прикладной программе, необходимо предусмотреть дополнительные функции контроля за счет специального системного блока SubscribeEvent – данный блок выдает информацию о идентификаторе записанной переменной и флаг активации.

Одни и те же переменные могут быть перечислены в обоих множествах (подписка и передача) – контроллер будет отправлять на сервер значение тех переменных, которые также могут быть изменены удаленно.

12. Модули обмена данными

Как видно из **Рис. 9** все интерфейсные блоки реализованы отдельно от виртуальной машины что дает возможность независимой (асинхронной) работы коммуникационных модулей. Конфигурация режимов работы и получение статуса связи выполняется посредством внутреннего API, а данные для передачи передаются в/из пространство памяти контроллера независимо от работы основной программы.

В зависимости от модификации контроллера состав интерфейсных модулей может изменяться, однако основные принципы конфигурации и обмена остаются неизменными.

12.1 Адресация Modbus

Адресация Modbus привязана к физическим адресам контроллера, однако ряд системных адресов защищен от записи для обеспечения безопасной работы контроллера.

Наиболее удобный способ получения информации об адресации – это использование YartStudio.

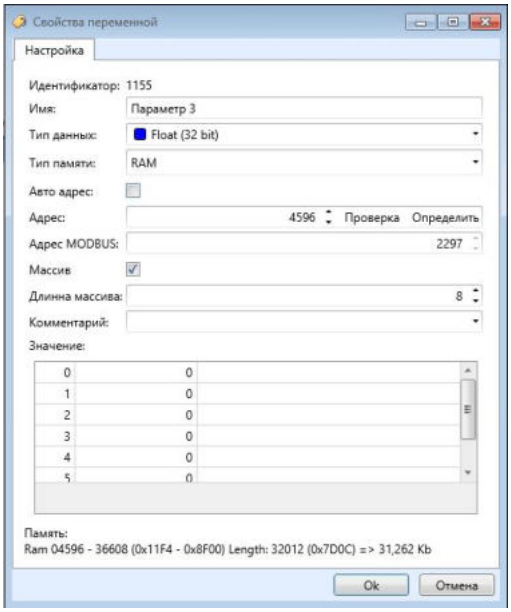


Рисунок 16. Редактирование переменных

Для точного определения любой переменной в Modbus пространстве необходимо использовать окно редактора переменных (**Рис. 16**).

YartStudio допускает два режима физического размещения переменных: автоматический и ручной (переключатель «Авто адрес»).

В автоматическом режиме размещения компилятор виртуальной машины последовательно размещает переменные в памяти данных. В этом режиме нет возможности однозначной идентификации физического адреса Modbus.

В ручном режиме пользователь сам выбирает тот адрес, где будет размещена переменная. При этом система предлагает пользователю возможность автоматизированного назначения адресов и проверки введенного адреса на пересечение с уже объявленными адресами переменных.

В поле «Адрес» указывается текущий начальный адрес размещения переменной. Кнопки «Проверить» и «Определить» позволяют осуществить проверку использованного адреса на предмет пересечений, и выбрать новый свободный в случае необходимости.

Ниже, в поле «Modbus» указан начальный адрес, по которому данная переменная будет доступна через Modbus протокол. В примере (**Рис. 16**) этот адрес равен 4596 (Holding Register).

Необходимо отметить, что стандартная спецификация Modbus не предполагает чтение переменных типа Float, более того разные платформы могут иметь различающийся формат записи чисел с плавающей точкой. Чтение Float можно реализовать как чтение двух последовательных Holding регистров и дальнейшее их объединение.

В случае линейки контроллеров Серебрум данные преобразования реализованы автоматически – достаточно прочитать/записать требуемое количество слов.

Все данные в памяти контроллера размещаются последовательно, без промежутков, поэтому в приведенном примере массив «Параметр 3» доступен по Modbus адресам 2297 – 2312.

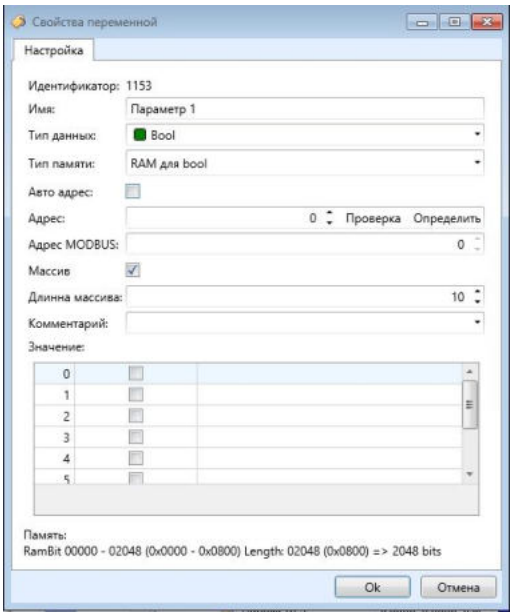


Рисунок 17. Пример с Bool

На **Рис. 17** определен массив переменных типа bool. Обратите внимание, что адрес указан как 0, поскольку доступ к bool памяти организован при помощи команд 01, 05, 15 – Coils Read/Write.

Изменение режима адресации переменной никак не сказывается на использование данной переменной в других программных модулях. Эта переменная может быть использована в модуле Modbus-RTU, телеметрии, архиве и т. д.

YartStudio позволяет осуществить экспорт списка переменных для последующей интеграции в продукты других компаний.

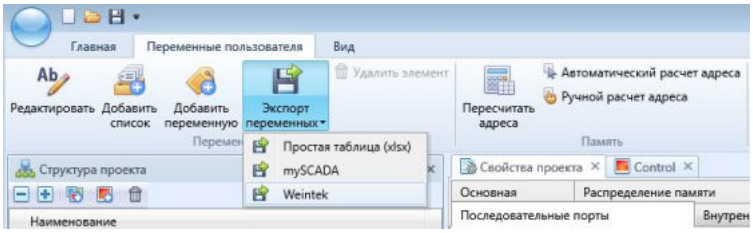


Рисунок 18. Меню экспорта списка переменных

В пункте основного меню «Экспорт переменных» можно выбрать формат выходных данных и осуществить экспорт. В зависимости от выбранного формата определенные в ручном режиме переменные будут сохранены в файл на диск.

12.2 Modbus RTU Master

Работа Modbus RTU осуществляется через последовательные порты контроллера.

В **Таблице 10** указано обозначение портов.

Порт COM1 жестко задействован для обмена со встроенным GPRS модемом и в списке не отображается.

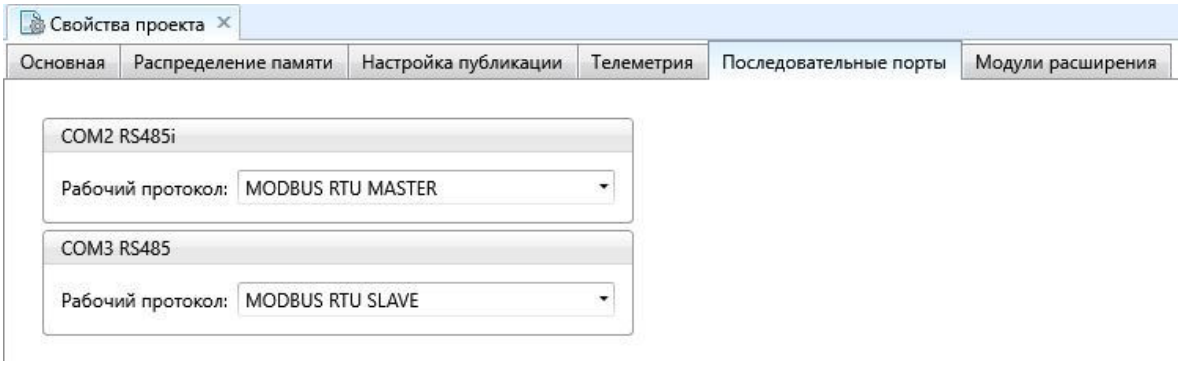


Рисунок 19. Базовая настройка последовательных портов

Для каждого порта допускается три рабочих режима, каждый из которых будет рассмотрен далее:

- Modbus RTU Master
- Modbus RTU Slave
- Пользовательский протокол

Таблица 10. Последовательные порты

Именование	Расположение	Обозначение в программе
COM1	Зарезервирован для GPRS модема	1
COM2	Изолированный (*) порт RS-485, клеммы 20, 21	2
COM3	Изолированный (*) порт RS-485, клеммы 41, 42	3

(*) последовательные порты контроллеры изолированы от цепей внешнего источника питания и модулей расширения. Между последовательными портами и аналоговыми входами цепи не изолированы.

12.2 Modbus RTU Slave

В режиме Modbus RTU Slave последовательный порт предоставляет доступ сторонним устройствам к записи/чтению данных памяти контроллера.

Для правильного использования режима Modbus RTU Slave в прикладной программе необходимо указать локальный адрес устройства и режим работы последовательного порта (**Рис. 20**).

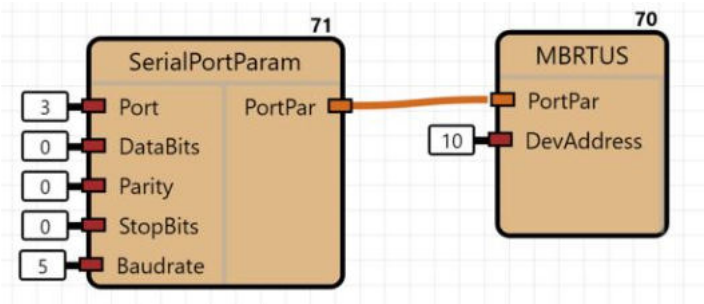


Рисунок 20. Настройка параметров Modbus RTU Slave

В блоке ***SerialPortParamCOBALT*** указывается условный номер порта (**Таблица 10**), и параметры обмена:

DataBits – Код числа бит в слове обмена

Parity – Режим четности

StopBits – Число Стоп бит

Baudrate – Скорость обмена

Подробная информация по командам содержится в описании программирования ПЛК Серебрум

В примере (**Рис. 20**) ***DataBits*** = 0 (8 бит), ***Parity*** = 0 (нет бита четности), ***StopBits*** = 0 (один стоп бит), ***Baudrate*** = 7 (скорость обмена 57600 бод).

При получении корректных Modbus RTU запросов на указанный порт COBALT автоматически сгенерирует и отправит ответ. Участия прикладной программы не требуется.

Участок кода, показанный на примере, **Рис. 20**, будет выполнен лишь в самом начале работы программы, соответственно параметры работы последовательного порта устанавливаются на всем протяжении времени жизни программы.

При необходимости изменить сетевой адрес или режим соединения потребуется перезагрузка прикладной программы контроллера.

Modbus RTU Slave контроллера COBALT поддерживает только команды чтения/записи Holding регистров (команды 0x03, 0x06, 0x10).

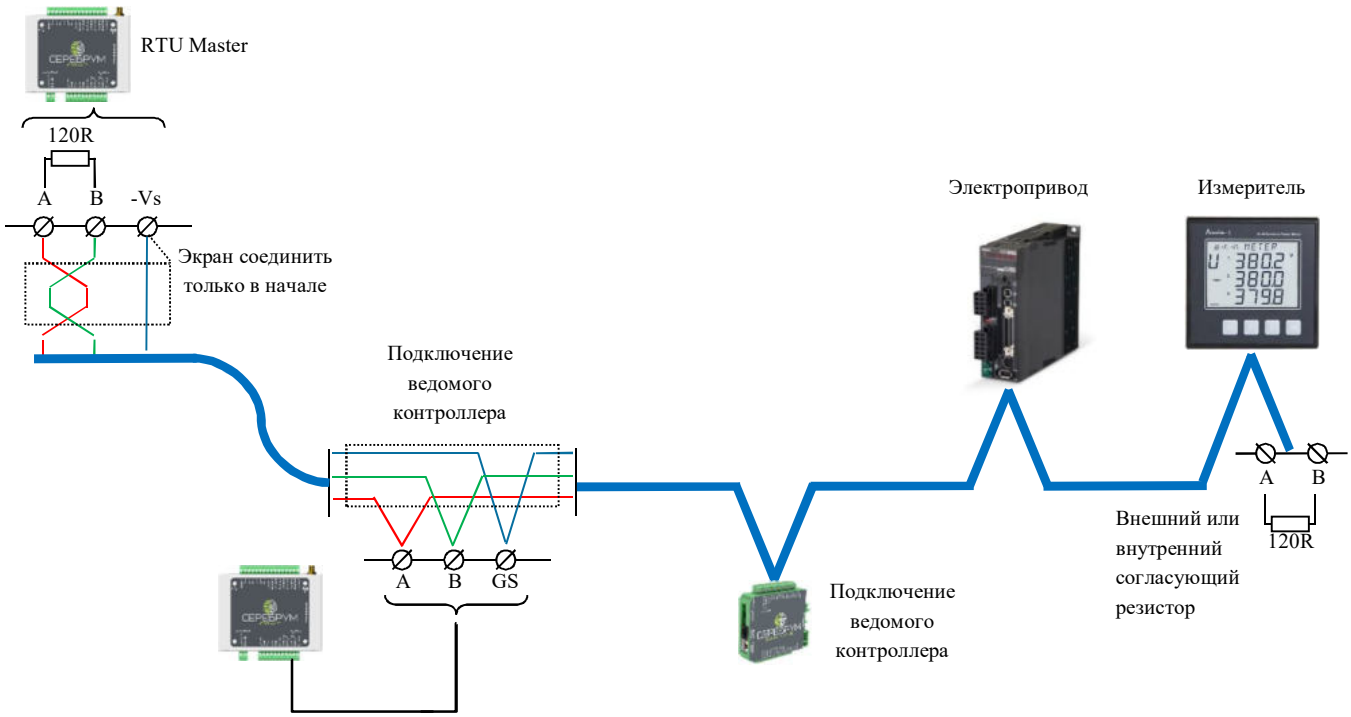


Рисунок 21. Подключение к шине Modbus RTU

На **Рисунке 21** показана приблизительная схема подключения устройств в шине RS-485 (Modbus RTU). В качестве ведущего устройства выступает контроллер COBALT – COM3.

Поскольку в данной конфигурации ведущий физически расположен в самом начале шины, то для порта COM3 включается согласующий резистор 120 Ом. Для этого устанавливается джампер (**поз. 7, Рис. 1**).

Другой контроллер COBALT подключается к шине через порт COM2.

На оконечном устройстве (в данном примере это Измеритель) также должен быть подключен согласующий резистор.

12.2.1 Modbus RTU Master

Если COM порт сконфигурирован в проекте в качестве ведущего устройства Modbus, то автоматически инициализируется программа обмена по сети Modbus RTU.

Конфигурация данной программы осуществляется через прикладную программу пользователя.

На **Рисунке 22** приведен пример программы настройки обмена через порт Modbus RTU Master.

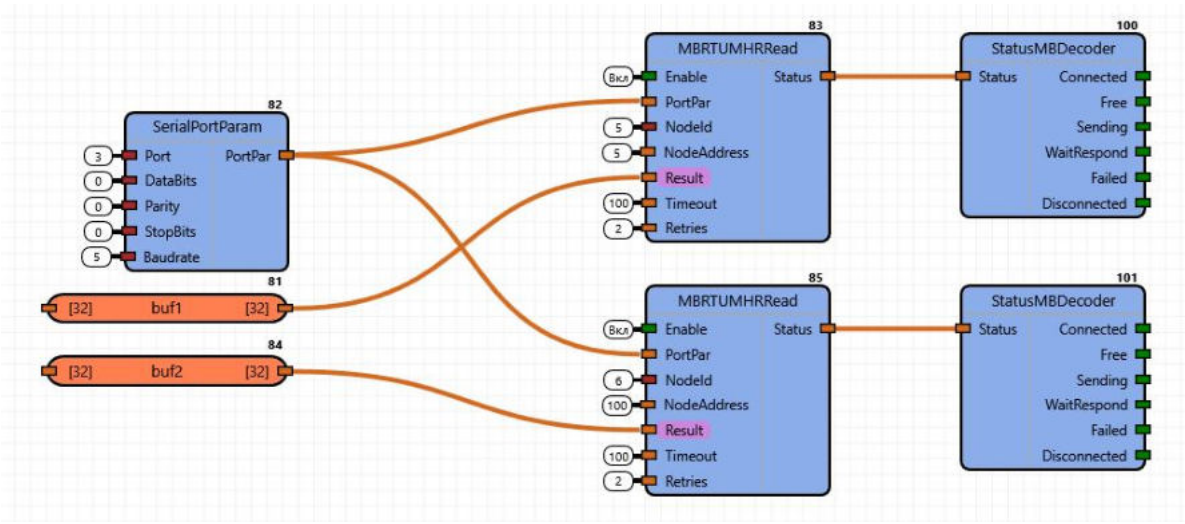


Рисунок 22. Конфигурация обмена Modbus RTU Master

Для каждого коммуникационного слота (команда для обмена) может быть определена своя уникальная настройка последовательного порта. В ряде случаев такой подход позволяет организовать несколько RTU сетей, работающих параллельно.

В примере настройки порта не изменяются и используется порт COM3, 19200 бод в формате 8N1 (блок 82 – ***SerialPortParam***).

Команда 83 – ***MBRTUMHRRead*** конфигурирует коммуникационный слот для чтения группы Holding регистров из устройства по сетевому адресу 5 (***NodeId***). Адрес первого регистра на удаленном устройстве тоже 5 (***NodeAddress***). Данные будут прочитаны в пользовательскую переменную buf1.

Обратите внимание, что длина запроса берется равной длине пользовательской переменной buf1, т. е. 32 слова.

Для данной команды определен таймаут отклика ведомого устройства (Timeout) 100 мс. Если отклик не будет получен, то программа Modbus RTU Master повторит запрос 2 (Retries) раза.

Статус выполнения команды расшифровывается в блоке ***StatusMBDecoder***.

Аналогично настроена и команда 85, за исключением того, что обращение производится к устройству по адресу 6, начальный адрес регистра 100.

Все запросы делятся по используемым ими портам. Каждый порт обрабатывается индивидуально. Запросы, запрошенные в программе для порта, следуют поочередно (**Рис. 23**).

При необходимости запрос может быть пропущен, если вход Enable при начале обработки запроса равен «0».

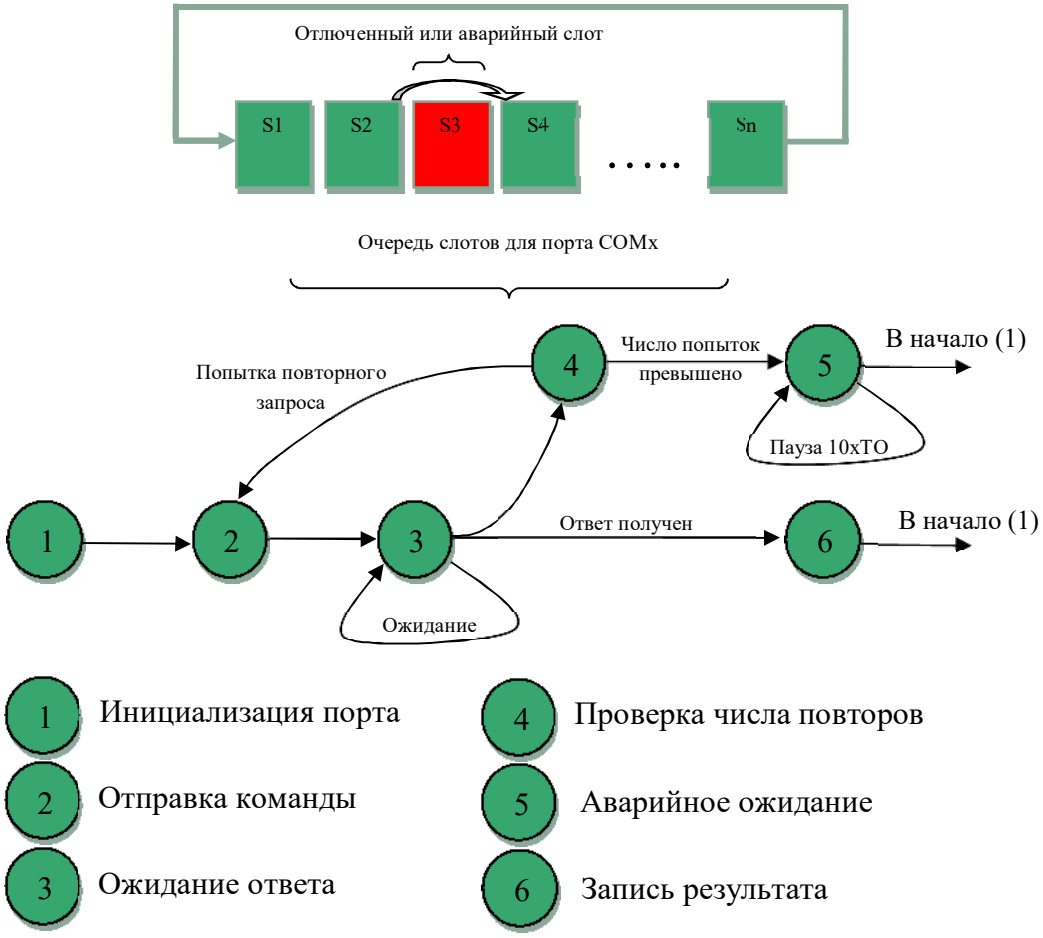


Рисунок 23. Работа слотов и диаграмма состояний коммуникационного слота.

Также слот пропускается в том случае, если все попытки получить отклик от удаленного устройства были неудачными (состояние 5). Пауза, в течение которой слот не будет задействован, как правило равна десяти таймаутам для данной команды.

При успешном завершении процедуры обмена (состояние 6) программа перейдет к обслуживанию следующего по очереди слота.

Обработка очереди запросов прекращается если контроллер переходит в состояние Стоп. Кроме того, очередь сбрасывается при каждом новом запуске контроллера.

Запрос полностью удаляется из очереди в случае, когда Enable равняется False более чем пять секунд. Поскольку суммарная длина очереди запросов ограничена 32 – таким образом возможно организовать обмен с большей суммарной длиной очереди.

12.3 Пользовательские протоколы

В контроллере COBALT предусмотрен API для самостоятельного программирования протоколов обмена через последовательные порты.

При необходимости использования собственного протокола (или какого-либо из поставляемых в составе библиотеки функциональных блоков) необходимо в окне основного режима COM портов (**Рис. 19**) выбрать пункт «Пользовательский протокол».

API рассчитан на использование языка C-Yart и содержит следующие функции:

Попытка открытия порта с заданными аргументами.

port_open (port, baud, flags)

В процессе выполнения данной команды операционная система проверят правильность конфигурации порта (режим «Пользовательский протокол»), доступность порта (порт не уже не открыт) и конфигурирует порт в соответствии с заданными в команде параметрами.

Значение параметра port должно быть равно коду порта (**Табл. 10**).

baud – значение скорости обмена: 9600, 19200, и т.д.

Поддерживаются следующие значения:

- 1200 бод
- 2400 бод
- 4800 бод
- 9600 бод
- 19200 бод
- 38400 бод
- 57600 бод
- 115200 бод

flags - битовое поле конфигурации

биты 31-3	**бит 2**	**бит 1**	**бит 0**
зарезервировано	STOPB	Parity	

STOPB: 1 – два стоповых бита; 0 – 1 стоповый бит

Parity: 0 – нет четности; 01 – Нечетный режим; 10 – Четный режим

В случае успешного открытия порта функция вернет значение 1.

В случае ошибки значения будут:

- -1 – порт уже открыт

- -2 – некорректный режим порта

Для закрытия порта используется функция ***port_close(port)***

port – аналогично ***port_open***

Все соединения сделанные ранее разрываются, параметры сбрасываются. Если порт находился в режиме приема или передачи данных, то работа будет остановлена.

port_check (port) – проверяет текущий режим порта. Функция возвращает «1», если порт открыт и «0», если порт закрыт.

port_send (port, address, len) – функция отправки сообщения через последовательный порт

port – индекс порта для отправки сообщения. Перед отправкой порт должен быть открыт

address – адрес в пространстве оперативной памяти (**Табл. 8**) где находится сообщение

len – длина сообщения в байтах

Функция возвращает:

- 0 - отправка началась
- -1 – ошибочный порт
- -5 – Оправка не удалась, передатчик занят
- -6 – Ошибка работы порта

port_receive (port, address, len)

port – индекс порта для получения сообщения. Перед приемом порт должен быть открыт

address – адрес в пространстве оперативной памяти (Табл. 8) где будет находиться сообщение

len – длина сообщения в байтах

Для информации о текущем статусе приема или передачи предусмотрены функции:

port_transmit_status (port) – проверка очереди передачи

port_receive_status (port) – проверка очереди приема

Функции возвращают число байт, оставшихся к приему или передаче.

0 – означает, что операция завершена

Отрицательное число обозначает ошибку.

stop_port_receive(port) – останавливает прием данных. ***port_receive_status()*** будет возвращать 0.

12.5 USB порт Yart Studio

Для подключения к COBALT используется встроенный USB порт (**поз. 5 Рис. 1**). При подключении к ПК в системе появится дополнительный USB-COM порт.

ОС должна определить новое устройство (***«Устройство с последовательным интерфейсом»***, **Рис. 26**).

В процессе установки Yart Studio доступен режим автоматической настройки драйвера для USB порта контроллера.

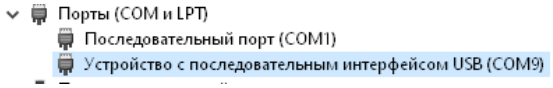


Рисунок 26. Подключение COBALT к ПК через USB порт

Доступ к данным COBALT будет осуществляться через протокол Modbus-RTU по адресу 1.

13. Работа с периферийными модулями контроллера

13.1 Аналоговые входы

В COBALT доступны четыре линии аналоговых входов, **Рис. 27**. В зависимости от конфигурации возможны следующие режимы измерения:

- Измерение входного напряжения
- Измерение входного тока (при подключении внешнего нагрузочного резистора)
- Измерение подключенного сопротивления
- Имитация дискретных входов

Переключение режимов работы аналоговых входов осуществляется при помощи переключателей, расположенных на плате контроллера. Перед переключением убедитесь, что питание контроллера и внешних цепей отключено.

На **Рис. 44** показана схема коммутации аналоговых входов.

Четыре канала могут быть сконфигурированы в дифференциальном и униполярном режиме. Дифференциальный режим предназначен для измерения величины внешнего сопротивления и мостовых схем. Измерение тока и напряжения выполняется в униполярном режиме.

Схема измерения внешнего сопротивления использует внутренние прецизионные резисторы в качестве опорных значений. В зависимости от требуемого диапазона измерения выбираются опорные резисторы 1К или 100К Ом.

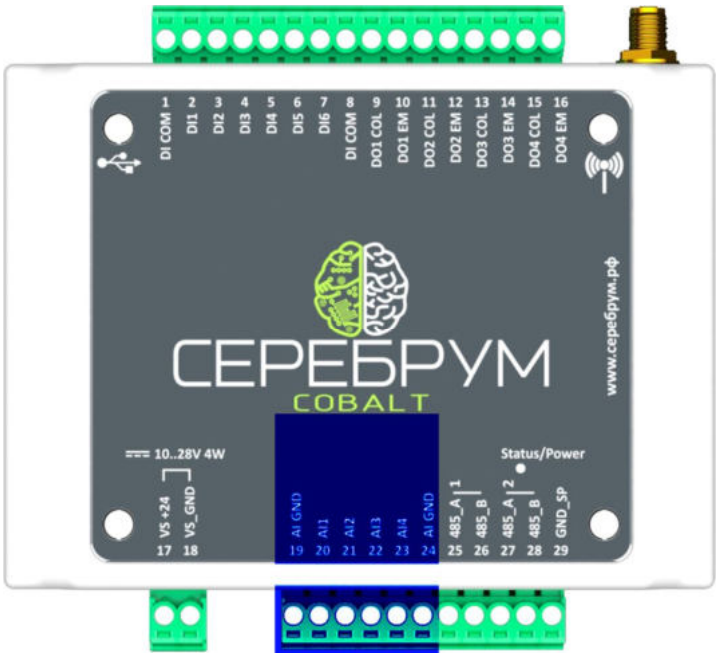


Рисунок 27. Аналоговые входы контроллера

Переключение схемы для измерения напряжения и сопротивления осуществляется автоматически.

На **Рисунке 28** показаны возможные схемы подключения входных сигналов к аналоговым входам:

На Схеме 1 измеряемое сопротивление Rm подключается к двум входам AI1 и AI2. Оставшаяся пара входов AI3:AI4 может использоваться для подключения еще одного резистора, дифференциального напряжения или для измерения двух однополярных напряжений (токов).

Переключатели режимов АЦП выбираются в соответствии с диаграммой справа. При максимальной величине измеряемого сопротивления меньше 5К Ом выбирается левый крайний переключатель (соответствует 1К Ому). В случае, когда измеряемое сопротивление может превышать значение в 5К Ом необходимо подключить опорные сопротивления в 100К Ом (средняя позиция переключателя).

В настройках аналоговых входов Yart Studio необходимо выбрать режим измерения сопротивления.

Результат преобразования записывается в AI1(3) в Омах.

Схема 2 (Рис. 28) соответствует подключениям для измерения несимметричного напряжения. Каждый из четырех каналов может быть сконфигурирован таким образом. Программные настройки Yart Studio конфигурируют внутренние алгоритмы расчета так, чтобы результат измерений был представлен в Вольтах.

Аналогичным образом осуществляется измерение входного тока. Аналоговый вход (Схема 3) конфигурируется в режиме измерения входного напряжения, формируемого на нагрузочном резисторе. Для обеспечения унификации с другими промышленными устройствами для сопротивления нагрузки рекомендуется устанавливать постоянный резистор 250 Ом.

Входы контроллера могут работать в режиме измерения дифференциального сопротивления (Схема 4). В качестве примера приведен измерительный мост (например тензопреобразователь), возбуждение которого осуществляется от входа AIN1 через последовательные резисторы 1К или 100К, в зависимости от положения встроенного переключателя.

Как правило тензопреобразователи используют большее рабочее напряжение возбуждения. В данном примере мост подключен к линии +24В через резистор 1К Ом из расчета, что падение напряжения на AIN1 составит 5В.

С учетом типового коэффициента преобразования тензомоста 2мВ/В входной диапазон получается как 5*2мВ = 10мВ. При работе АЦП с максимальным коэффициентом усиления диапазон разбивается на 10 мВ / 0.125 мВ = 80 условных отсчетов АЦП.

При проектировании схем с использованием аналоговых входов следует уделить внимание следующим особенностям схемы:

- ✓ Резистивные датчики должны измеряться в дифференциальном режиме по схеме Рис. 55
- ✓ Монтаж проводов соединений с резистивными датчиками должен осуществляться на максимальном расстоянии от силовых кабелей, особенно преобразователей частоты и выпрямителей
- ✓ Общий сигнал токовых датчиков должен быть соединен с клеммой G-AI

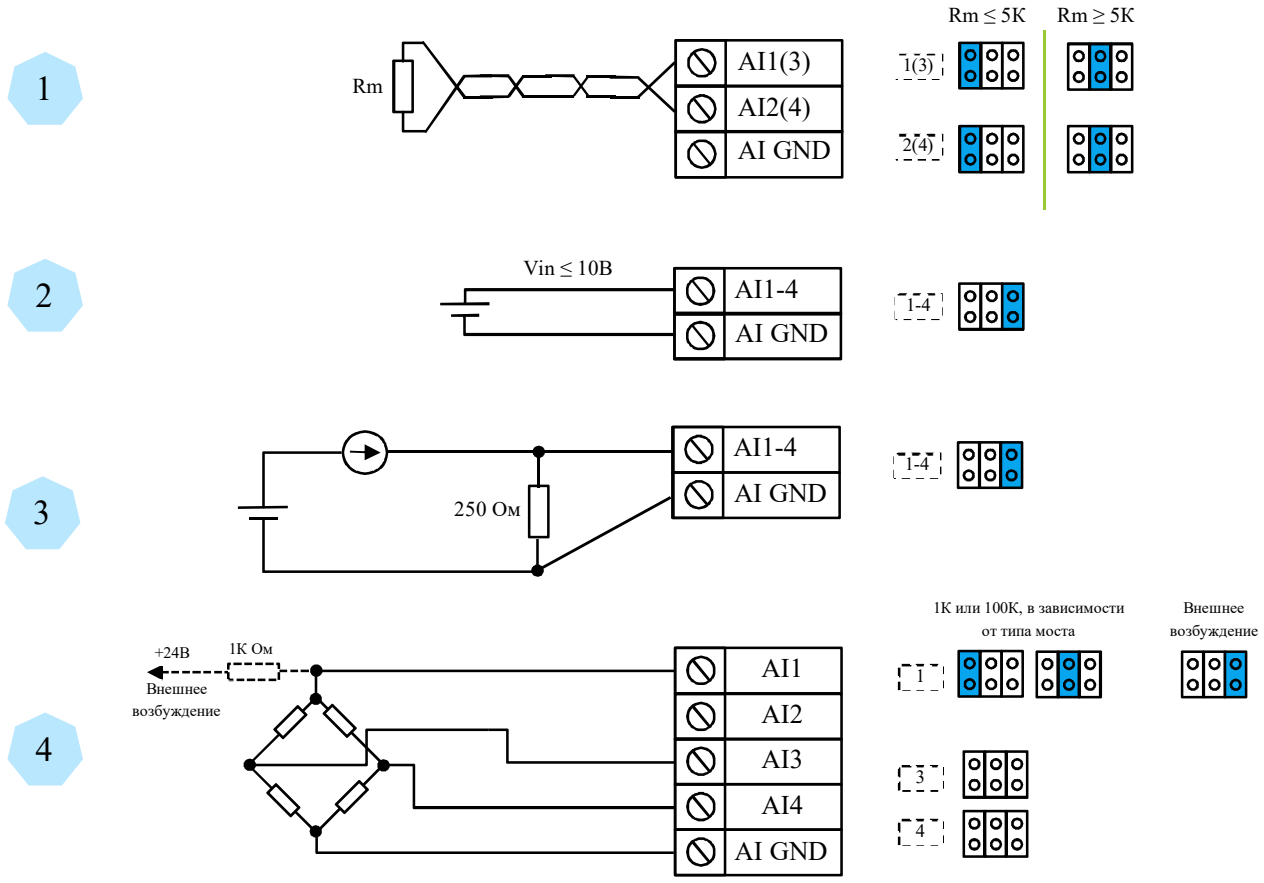


Рисунок 29. Установка режимов аналогового входа

Дальнейшая работа с аналоговыми входами выполняется в Yart Studio. Для каждого из используемых аналоговых входов необходимо выбрать требуемый режим работы, **Рис. 29**.

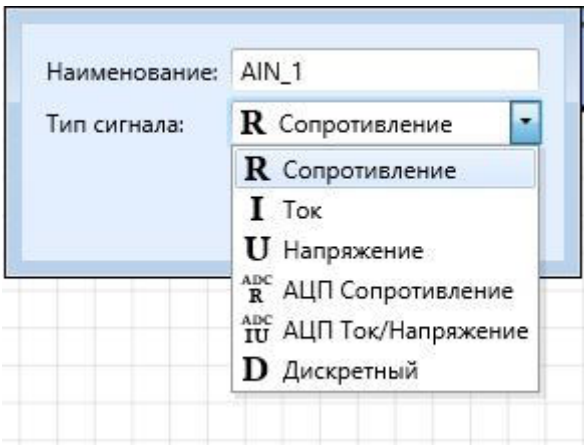


Рисунок 29. Установка режимов аналогового входа

Так, для примера выше для AI7 и AI8 необходимо установить режим «I Ток», AI6 – «D Дискретный», AI5 – «U Напряжение» и AI3 – «R Сопротивление».

Выход каждого канала – число с плавающей точкой в формате измеряемой величины, **Рис. 30**.

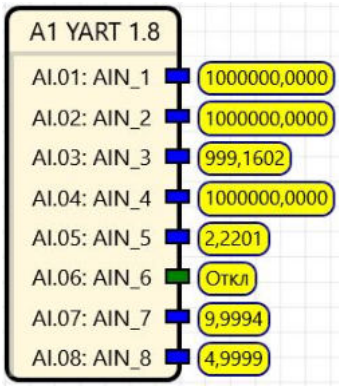


Рисунок 30. Установка режимов аналогового входа

Токовые каналы выдают значения в миллиамперах, напряжение выражается в Вольтах, величина сопротивления выражается в Омах. При использовании аналогового входа в качестве дискретного выход такого канала имеет тип Bool.

В ряде случаев удобно использовать необработанный код АЦП, для этого в списке режимов работы добавлены «ADC R АЦП Сопротивление» и «ADC IU АЦП Ток/Напряжение».

13.2 Дискретные входы

К контроллеру COBALT может подключаться до шести линий дискретных сигналов с номинальным напряжением «лог. 1» 24В. Каждый из входов (Рис. 31) может использоваться в режиме автоматического низкоскоростного счетчика импульсов (максимальная частота подсчета 100 Гц).

Дискретные входы контроллера - биполярные, поддерживается управление положительным и отрицательным потенциалами. Поскольку общий потенциал дискретных входов объединен - управляющий потенциал должен быть одинаковым для всей группы входов.

Входы DI1:DI2, DI3:DI4 и DI5:DI6 используются как входы для встроенных (аппаратных) модулей счета/захвата, позволяющих обрабатывать сигналы от высокоскоростных датчиков скорости.

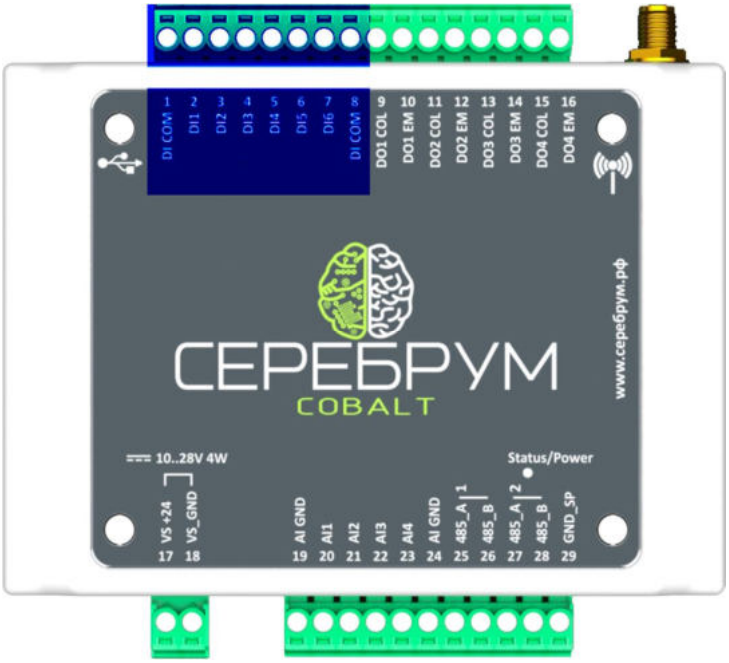


Рисунок 31. Дискретные входы контроллера

Подключение внешних сигналов к входам DIx должно выполняться по схеме, **Рис. 32**

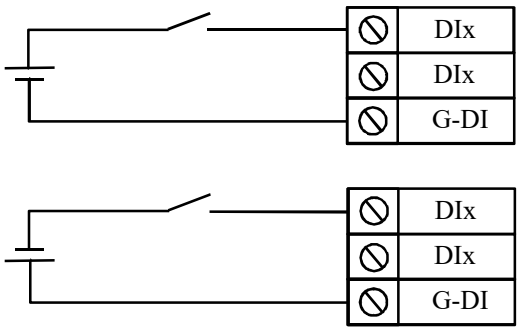


Рисунок 32. Схема подключения дискретных входов

Настройка режимов работы дискретных входов осуществляется при помощи Yart Studio (**Рис. 33**)

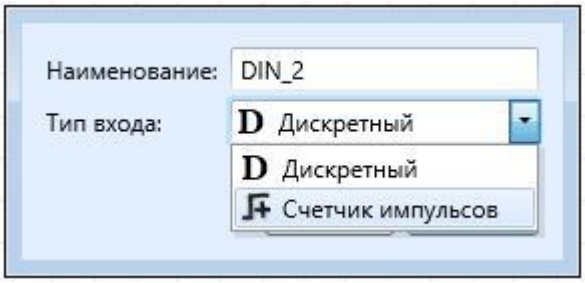


Рисунок 33. Дискретные входы контроллера

Режим «Дискретный» - обычный режим работы. Выход блока отражает текущее состояние дискретного входа.

В режиме «Счетчик импульсов» выходом блока будет являться число Int, равное числу детектированных переходов от лог. 0 к лог. 1. Для корректной работы входа в таком режиме частота на входе не должна превышать 100 Гц.

Необходимо отметить, что работа счетчика импульсов не зависит от текущего времени выполнения прикладной программы пользователя.

При использовании аппаратных счетчиков конфигурация осуществляется при помощи набора функциональных блоков, **Рис. 34**.

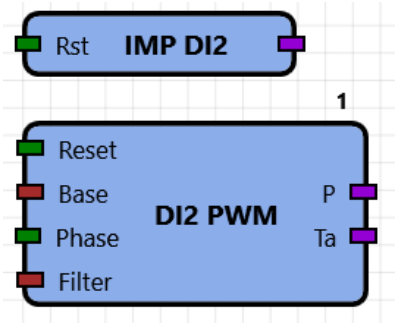


Рисунок 34. Функциональные блоки аппаратных счетчиков

Каждый из двух счетчиков настраивается для работы в одном из трех режимов

Инкрементальный энкодер. В этом режиме на каждый счетчик подключаются два сигнала: A:B к обоим каналам входов высокоскоростных счетчиков, **Рис. 35**.

В зависимости от направления вращения вала значение счетчика будет увеличиваться/уменьшаться пропорционально скорости вращения.

Аппаратные счетчики имеют разрядность 16 бит, поэтому в режиме энкодера требуется дальнейшая обработка для предотвращения переполнения значения.

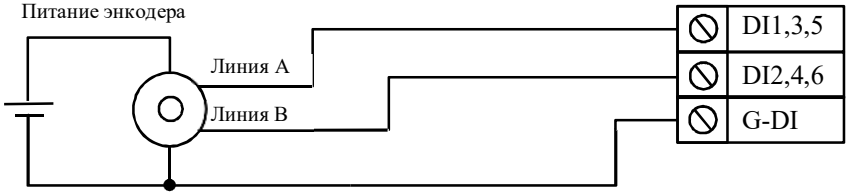


Рисунок 35. Подключение инкрементальных энкодеров

Счетчик импульсов – режим похож на режим энкодера кроме отсутствия определения направления. Контроллер считает число переходов сигнала из 0 в 1 (в зависимости от выбранной полярности). Подключение осуществляется по схеме, **Рис. 36**.

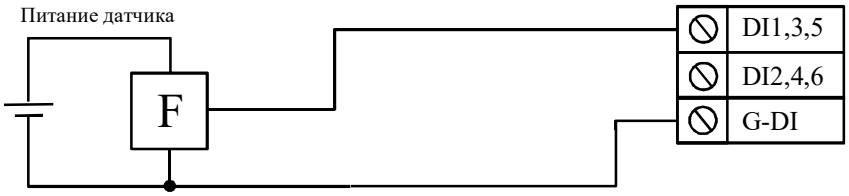


Рисунок 36. Подключение скоростных датчиков

Режим захвата. В данном режиме контроллер определяет длительность активного состояния на входе (**Рис. 37**).

На выходе формируется два значения: длительность активного состояния (Ta) и период (P).

Разрешающая способность определяется значением Base: 1 мкс, 10 мкс, 100 мкс, 1000 мкс.

Таким образом измеряемая длительность ограничивается 65.535 мс, 655.35 мс, 6553.5 мс и 65535 мс соответственно.

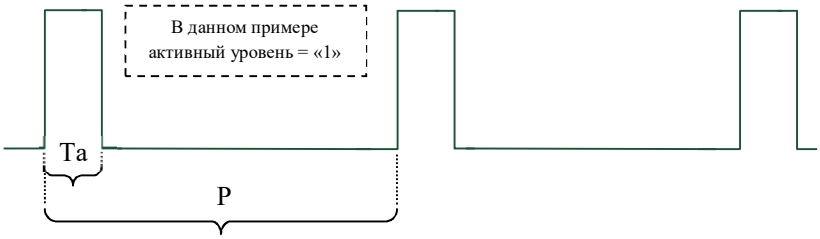


Рисунок 37. Работа модуля захвата

Значения Ta и P обновляются на каждом периоде сигнала.

13.4 Дискретные выходы

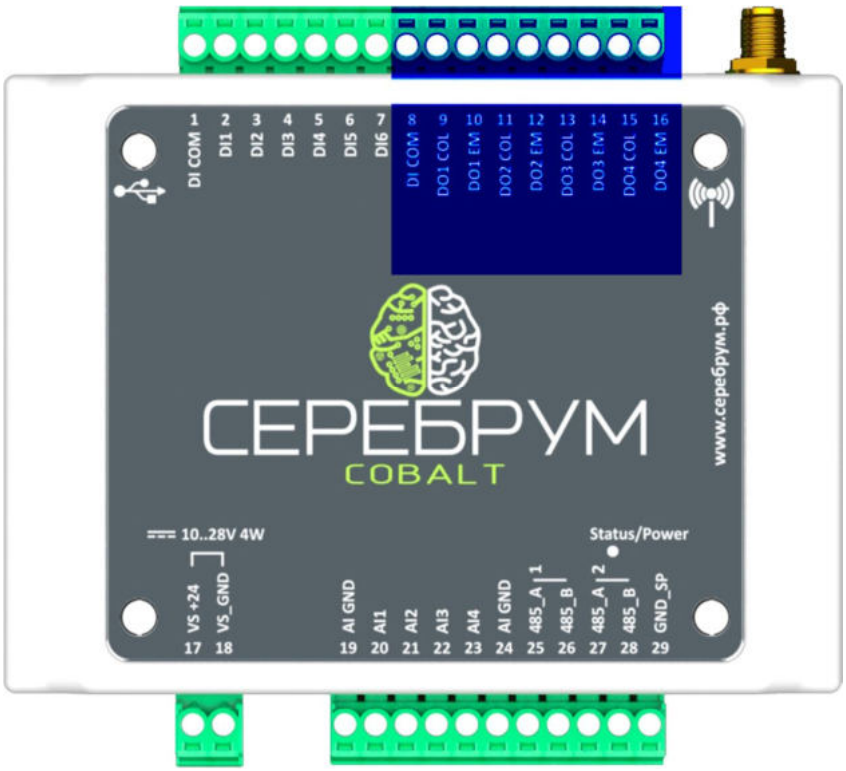


Рисунок 41. Дискретные выходы контроллера

В состав COBALT входят четыре дискретных выхода на основе оптотранзисторов. Каждый оптотранзистор имеет индивидуальную изоляцию.

На **Рисунке 42** показана предполагаемая схема подключения нагрузки к выходам контроллера.

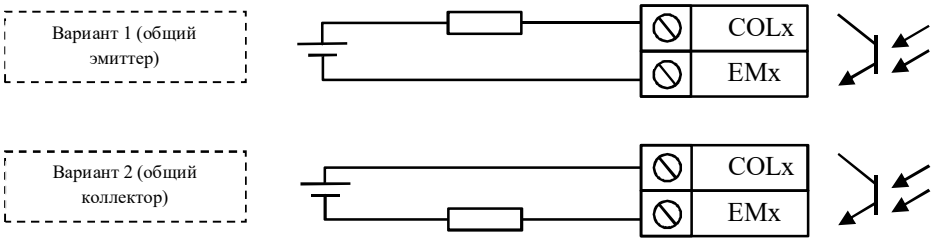


Рисунок 42. Подключение нагрузки к дискретным выходам контроллера

Каждый дискретный выход контроллера поддерживает четыре режима работы:

- дискретный выход
- выход программный ШИМ
- выход аппаратный ШИМ
- выход управления сервоприводом

Настройка режима осуществляется в Yart Studio в разделе «Состав оборудования» - «Выход» - «Дискретный», **Рисунок 43**.

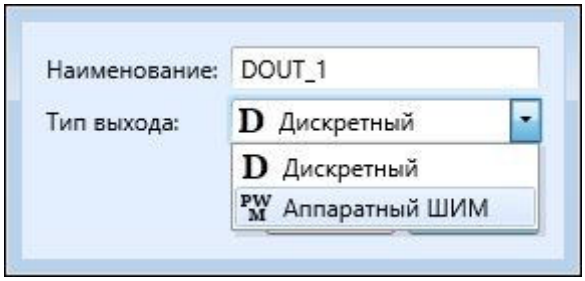


Рисунок 43. Конфигурация дискретного выхода

Режим работы «Дискретный» предполагает прямую трансляцию значения Bool на физический выход контроллера: False – выключено, True – включено.

В режиме «Программный ШИМ» на вход функционального блока подается массив из двух Int. Для удобства формирования ШИМ в состав библиотеки включены блоки формирования массива управления (PWM, 2PWM, 3PWM).

На **Рисунке 44** показан пример использования ШИМ.

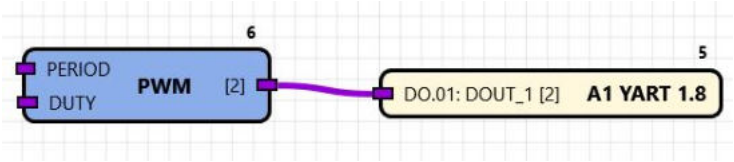


Рисунок 44. Формирование ШИМ на выходе ПЛК

Значение PERIOD соответствует периоду ШИМ и задается в миллисекундах. DUTY – величина длительности активного уровня сигнала (в мс).

В режиме аппаратного ШИМ работа выхода аналогична работе “Программного ШИМ” но разрешающая способность равна 10 мкс. Значения периода и скважности ограничиваются 16 разрядами. Таким образом максимальный период выхода ограничивается 655.35 мс.

13.5 Приводной контроллер

В режиме управления сервоприводом контроллер автоматически рассчитывает профиль скорости двигателя и формирует сигналы управления Step/Dir.

К одному контроллеру может быть подключено до двух приводов, в соответствии со схемой (**Рис. 45**).

Выходы 1-2 управляют осью №1, выходы 3-4 управляют осью №2. При этом выходы 1 и 3 должны подключаться к входам STEP, а выходы 2 и 4 управляют направлением вращения и подключаются к входам DIR.

Обе оси контроллера работают одновременно и синхронно. Режимы работы и координаты перемещений задаются при помощи прикладной программы пользователя.

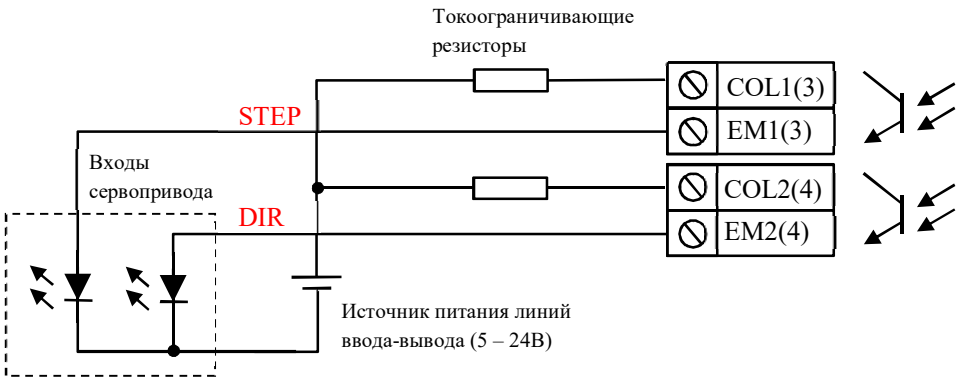


Рисунок 45. Подключение сервопривода к контроллеру Cobalt

Для упрощения программирования в контроллере приняты следующие базовые единицы:

- Единицы расстояния – шаги двигателя
- Скорость – условное количество шагов двигателя в секунду
- Ускорение – изменение величины скорости двигателя за секунду

Начальная конфигурация контроллера движения осуществляется блоком **SetMotion** (**Рис. 46**).

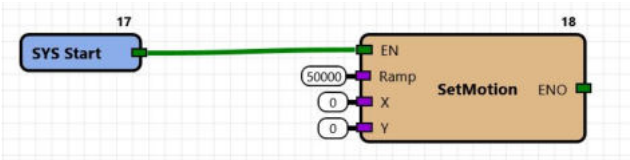


Рисунок 46. Подключение сервопривода к контроллеру Cobalt

Вход Ramp задает ускорение для работы осей контроллера. В данном случае 50000 шагов/с^2.

Входы X и Y задают текущее положение осей. В примере выше, позиция осей сбрасывается в нулевое положение, однако при необходимости на вход блока **SetMotion** можно передавать фактическое положение приводов от энкодеров.

Работа контроллера движения осуществляется автоматически, по положениям, определяемым в двух буферах позиционирования.

В один момент времени активен только один буфер, из которого контроллер автоматически забирает данные для следующего перемещения. Все перемещения задаются в режиме Старт-Стоп, таким образом привода всегда достигают целевых точек перемещения.

На Рисунке 47 схематично отображена схема движения приводов. Движение начинается из точки с координатами {0,0} в точку {10000, 5000} со скоростью 1000 шагов/сек. Необходимо отметить, что в данном случае скорость 1000 задается для обобщенного движения по обоим осям. Таким образом фактические скорости будут равны 894 и 447 имп/сек.

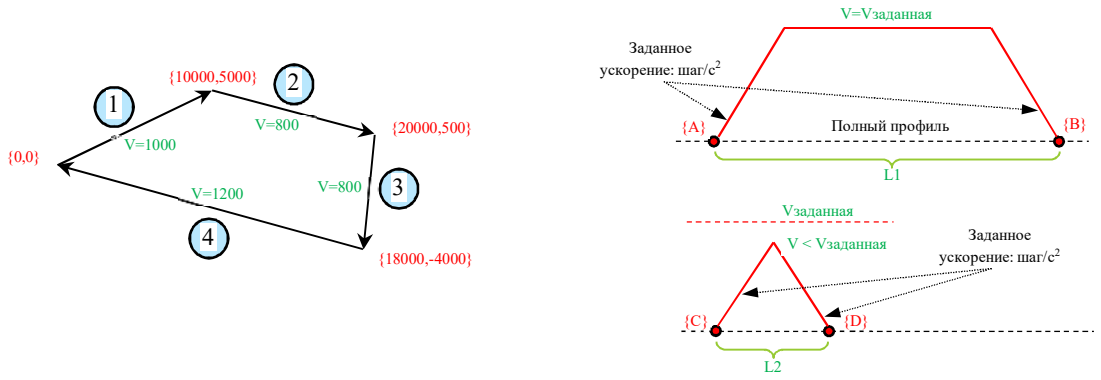


Рисунок 47. Работа контроллера движения

Всего в приведенной диаграмме приводы должны будут выполнить четыре перемещения, вернувшись в начальную точку с координатами {0,0}.

В правой части рисунка показаны возможные графики изменения маршевой скорости в зависимости от заданного режима движения.

Верхняя диаграмма показывает, что заданная скорость достигается – профиль скорости состоит из трех основных этапов: ускорение, перемещение на заданной скорости и замедление.

Для нижней диаграммы параметры движения заданы таким образом, что скорость, определенная в программе движения, не может быть достигнута. В этом случае профиль скорости будет состоять только из этапов разгона и торможения, а максимальная скорость перемещения будет определена таким образом, чтобы к моменту остановки привода пройденное расстояние было равно заданному L2.

В целом все профили движения в контроллере рассчитываются исходя из того, чтобы обе приводные оси одновременно выполнили перемещение в заданные координаты.

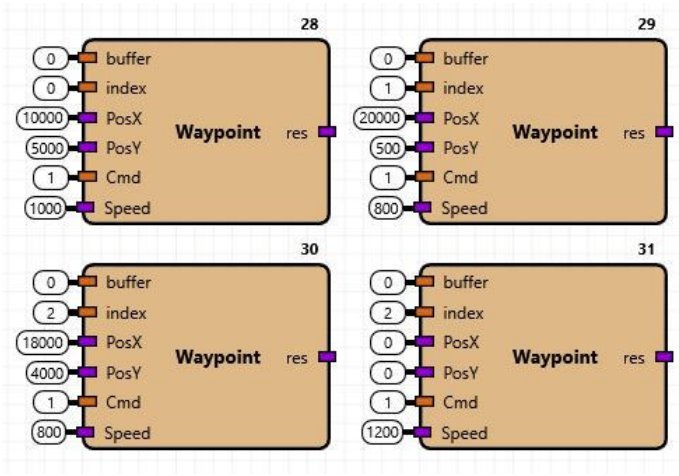


Рисунок 48. Формирование программы движения

На **Рисунке 48** показан пример, формирующий программу движения из примера на **Рис. 47.** Начало движения предполагается из точки {0,0} и при первом перемещении приводы переместятся в точку с координатами {10000, 5000}.

Каждый последующий этап перемещения начинается только после окончания предыдущего. Кроме того, профиль скорости рассчитывается динамически перед началом движения.

Как видно из примера, для задания узловой точки программы движения необходимо задать следующие поля:

- buffer – индекс буфера в который записывается путевая точка (0 или 1)
- index – индекс путевой точки в самом буфере (0-199)
- PosX – конечная позиция движения по оси 1
- PosY – конечная позиция движения по оси 2
- Cmd – команда для контроллера перемещения: 1 – перемещение в режиме Старт-Стоп
- Speed – маршевая скорость. Задается для суммарного перемещения по обоим осям управления

Начало движения осуществляется при помощи блока StartMotion (**Рис. 49**). В качестве входных аргументов задается индекс буфера, содержащего программу движения, и суммарное количество запрограммированных перемещений.

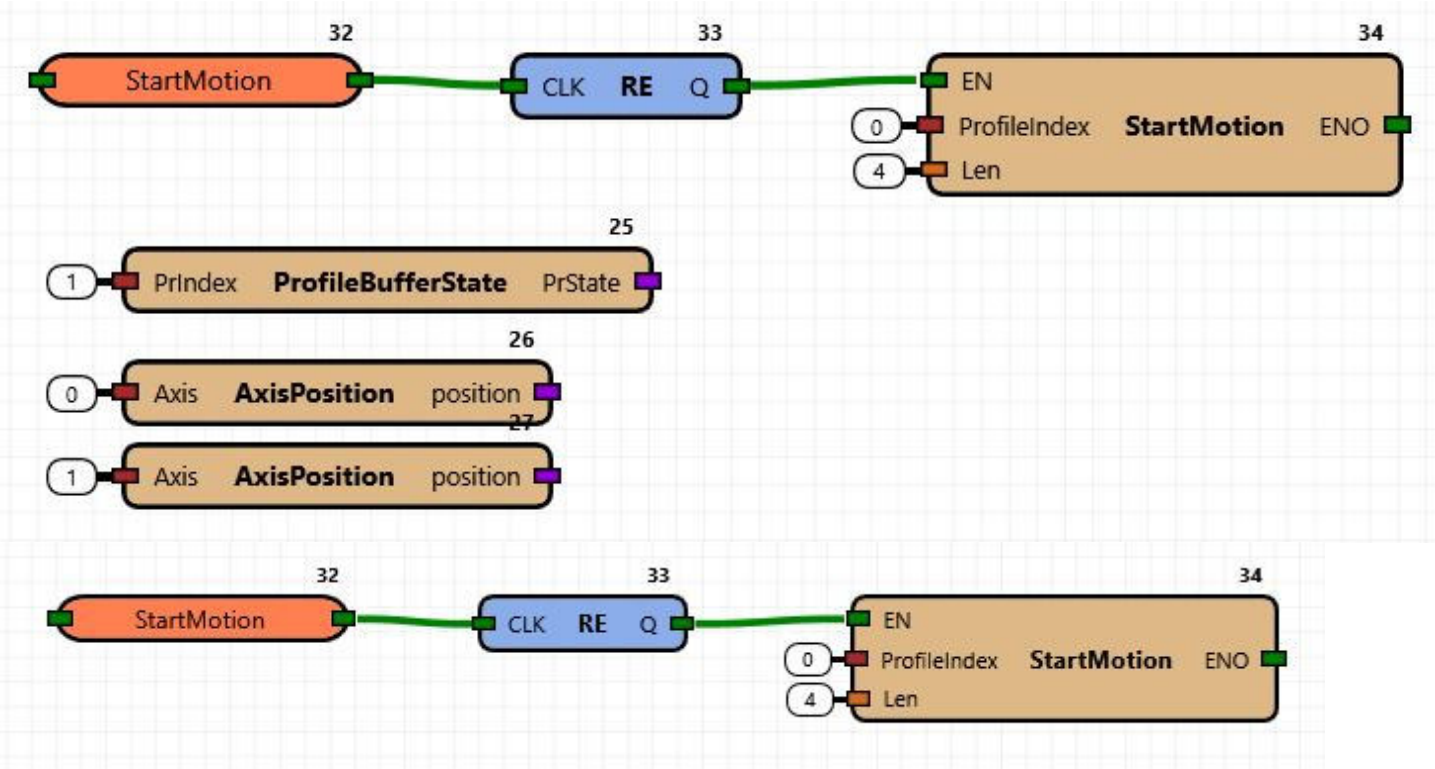


Рисунок 49. Начало движения по программе

Ход исполнения программы позиционирования доступен при помощи функциональных блоков **ProfileBufferState** и **AxisPosition**, **Рисунок 49.**

Блок **ProfileBufferState** возвращает текущий активный номер в заданном буфере управления. Так, для примера выше блок **ProfileBufferState** должен будет вернуть число 5 (число шагов +1) после того, как приводы вернуться обратно в точку с координатами {0,0}.

После этого прикладная программа может снова активировать перемещение, запрограммированное в любом из доступных буферов движения.

Блок **AxisPosition** возвращает текущее положение для заданной оси управления.

13.6 Настройка часов

В COBALT встроены часы реального времени с резервным питанием от литиевой батареи 3В (CR2032). Работа часов не зависит от выполнения прикладной программы пользователя за исключением использования команд программной установки времени.

Значение текущего времени/даты доступно в программе пользователя (блоки GET DATE, GET_TIME).

При подключении к ПЛК текущее время/дата контроллера доступно для чтения и установки.

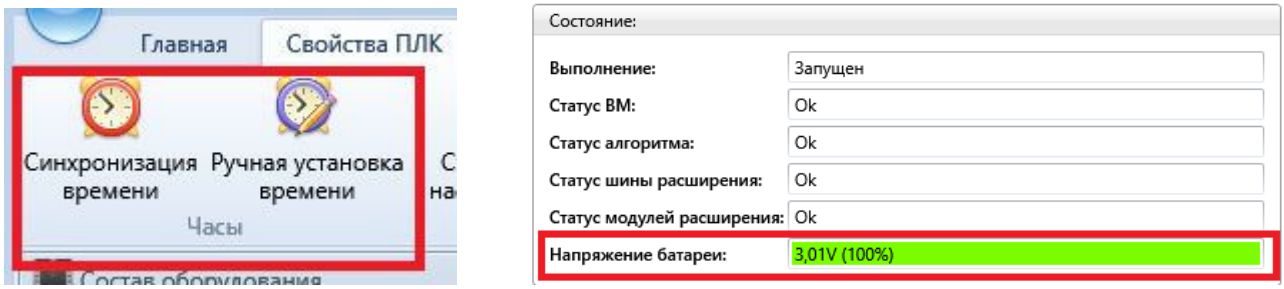


Рисунок 50. Подключение нагрузки к дискретных выходам контроллера

Состояние батареи отображается в поле «Напряжение батареи» раздела «Состояние».

При падении напряжения батареи ниже 2.5В батарею необходимо заменить. Для замены снимите верхнюю крышку контроллера, выкрутите болты крепления платы и снимите плату с пластикового основания. Замените батарею на новую и осуществите сборку контроллера в обратной последовательности.

Расчетный срок эксплуатации новой батареи – 20 лет.

Величина напряжения доступна для контроля из пользовательской программы через блок U BATTERY, **Рисунок 51**.

Рисунок 51. Подключение нагрузки к дискретных выходам контроллера

14. Работа с внешними периферийными модулями

Настройка работы внешних периферийных модулей производится в разделе «Состав оборудования». В данном разделе отображаются модули, добавленные в разделе настройки проекта – «Модули расширения» (**Рис. 7**).

Для модулей аналогового ввода необходим также выбор основного режима работы.

В процессе работы контроллера производится опрос внешних модулей расширения: модули ввода опрашиваются до старта очередного цикла программы, в модули вывода данные заносятся после выполнения программы.

Поскольку скорость выполнения прикладной программы может превышать физические характеристики периферийных модулей (например AI8) – фактические данные измерений будут обновляться не на каждом такте исполнения прикладной программы.

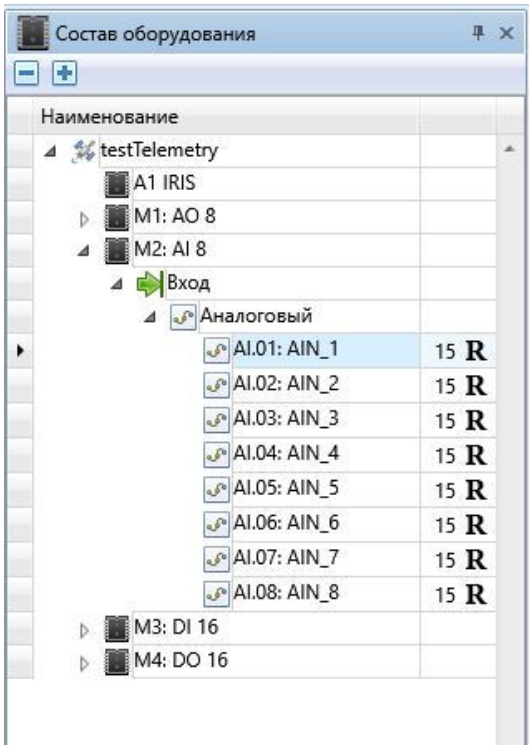


Рисунок 52. Панель настройки периферийных модулей

14.1 Аналоговые входы

Модуль AI8 (**Рис. 53, 54**) основан на 12-разрядном АЦП со встроенным программируемым усилителем.

Подключение модуля рекомендуется выполнять в соответствии со схемой (**Рис. 48**). При этом клеммы «Earth», 1, 16, 17, 32 должны подключаться к контуру заземления шкафа непосредственно в месте установки модуля.

Необходимо подключение всех клемм заземления.

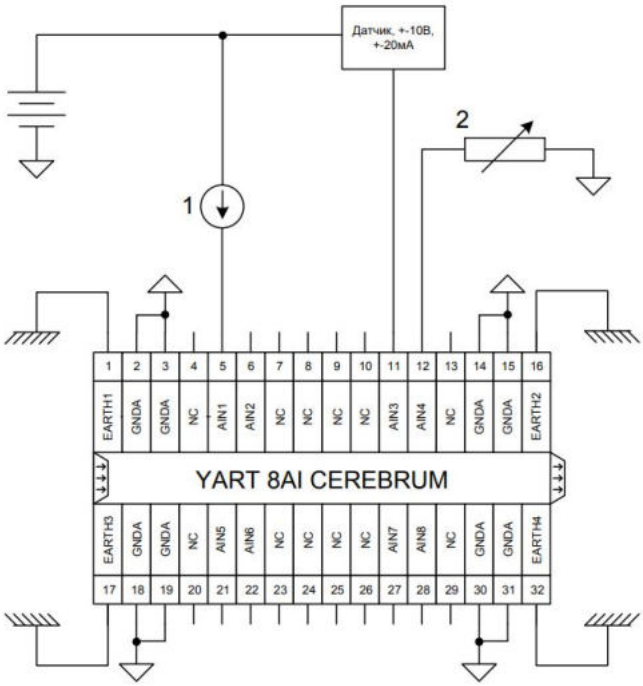


Рисунок 53. Модуль 8AI

На схеме (**Рис. 53**) цифрами обозначены:

1 – Токовый датчик, подключенный по 2-проводной схеме
2 – Измеряемое внешнее сопротивление (датчик температуры, дискретный вход)

Все каналы модуля имеют групповую гальваническую развязку от напряжения питания шины расширения и контроллера.

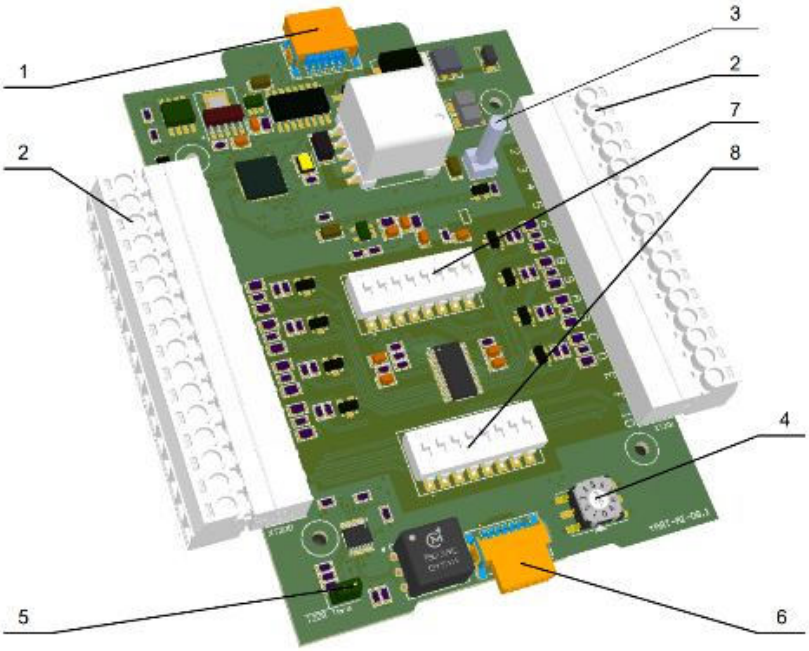


Рисунок 54. Модуль 8AI

1 – разъем для подключения к контроллеру или «ведущему» модулю расширения
2 – группа клемм для подключения внешних соединителей
3 – индикатор состояния работы модуля
4 – переключатель адреса в шине модулей расширения
5 – перемычка включения терминатора линии передачи данных
6 – разъем для подключения последующих модулей расширения
7 – переключатели 1 активации режима измерения сопротивления
8 – переключатели 2 активации режима измерения тока

Кроме того, входной каскад АЦП имеет внешние переключатели для выбора режима работы каналов. Доступно три основных режима (**Таблица 11**):

- измерение напряжения
- измерение тока
- измерение сопротивления

Таблица 11. Режимы работы модуля AI8

Режим работы	Переключатель 1	Переключатель 2
Тестовый режим	Включен	Включен
Измерение сопротивления	Включен	Выключен
Измерение тока	Выключен	Включен
Измерение напряжения	Выключен	Выключен

Каждый канал настраивается индивидуально. В зависимости от настроек программируемого усилителя изменяется входной диапазон измерений, в рамках которого работает 12-разрядный АЦП. Специальные

блоки программного обеспечения ПЛК выполняют дальнейший пересчет полученных кодов АЦП, предоставляя пользователю уже обработанные данные (в физических величинах: В, мА, Ом).

Обратите внимание, что при измерении биполярного напряжения (или тока) 12 разрядов распределяются на весь диапазон измерения, т. е. 11 разрядов АЦП на положительное напряжение и 11 разрядов на отрицательное напряжение.

Также предусмотрены режимы, когда выходом блока AI8 являются необработанные данные АЦП.

Дальнейшие настройки выполняются в YartStudio в соответствии с Таблицей 11

Измерение напряжения

Доступны восемь режимов:

- 0-1.25В / ±0-1.25В
- 0-2.5В / ±0-2.5В
- 0-5.0В / ±0-5.0В

Изменение тока

Все указанные диапазоны приведены с учетом подключаемого нагрузочного резистора 250 Ом.

- 0-20.0mA / ±0-20.0mA
- 0-10.0mA / ±0-10.0mA
- 0-5.0mA / ±0-5.0mA
- 0-2.5mA / ±0-2.5mA

Измерение входного сопротивления / дискретный вход

Измерение сопротивления осуществляется за счет измерения падения напряжения на входе канала при подаче нормированного тока возбуждения.

Схема измерения не является дифференциальной, поэтому подвержена влиянию синфазной помехи, возникающей на длинных кабелях.

При необходимости измерения сопротивления на расстояниях более 10 метров, а также при работе каналов измерения рядом с мощными силовыми установками рекомендуется использовать преобразователи сопротивление/ток.

Доступно три диапазона измерения:

- 0-4k
- 0-18k
- 0-1M

Канал измерения сопротивления способен имитировать дискретный вход. Программа автоматически определяет состояние низкого входного сопротивления, формируя «0» на выходе.

Для использования данных модуля AI8 в прикладной программе (**Рисунок 35**) необходимо использовать индивидуальный функциональный блок Aix или весь модуль сразу.

Добавление входа в программу осуществляется путем перетаскивания из выпадающего списка каналов (**Рис. 55**).

Пункт списка «Аналоговый» добавит все каналы одновременно.

Возможно добавления каждого канала по отдельности, выбирая элементы AI.xx:AIN_x.

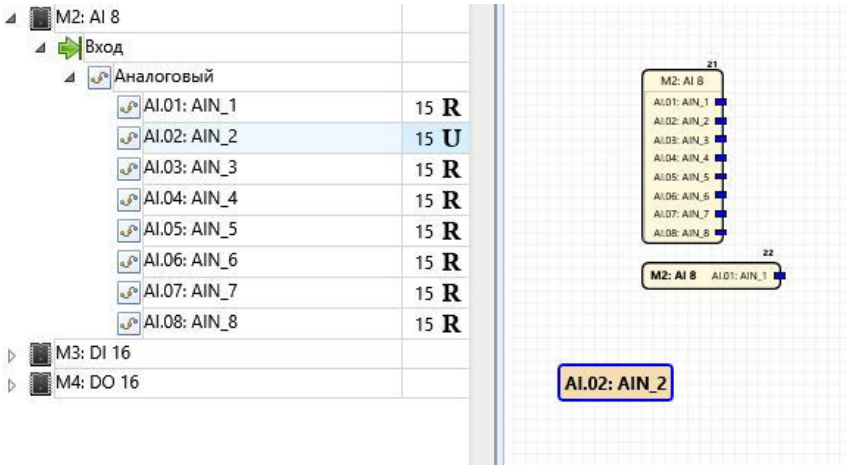


Рисунок 55. Функциональные блоки аналоговых входов

Выходы программных блоков модуля AI формируют выходное значение в формате с плавающей точкой.

Значения выходов соответствуют измеренным физическим величинам в зависимости от выбранного режима работы (R, U, I, DI).

В режиме DI выход имеет тип Bool.

В состав встроенной библиотеки включены уже готовые функциональные блоки для осуществления дальнейших преобразований: датчики температуры, давления и т. д.

14.2 Аналоговые выходы

Модуль 8АО (**Рис. 56**) формирует восемь независимых каналов напряжения и тока. Формирователи тока и напряжения независимые и могут работать одновременно (каждый выходной канал имеет две клеммы: ток и напряжение).

Для работы токовых каналов модулю необходимо внешнее питание (поз. 1 и 2, **Рис 56**). Для устойчивой работы модуля достаточно запитать один вход – второй может оставаться в резерве.

Все каналы модуля имеют групповую гальваническую развязку от напряжения питания шины расширения и контроллера.

Для фильтрации помех предусмотрены клеммы 1, 17, 32, 16 (Earth), которые должны присоединяться к заземляющему контуру системы управления.

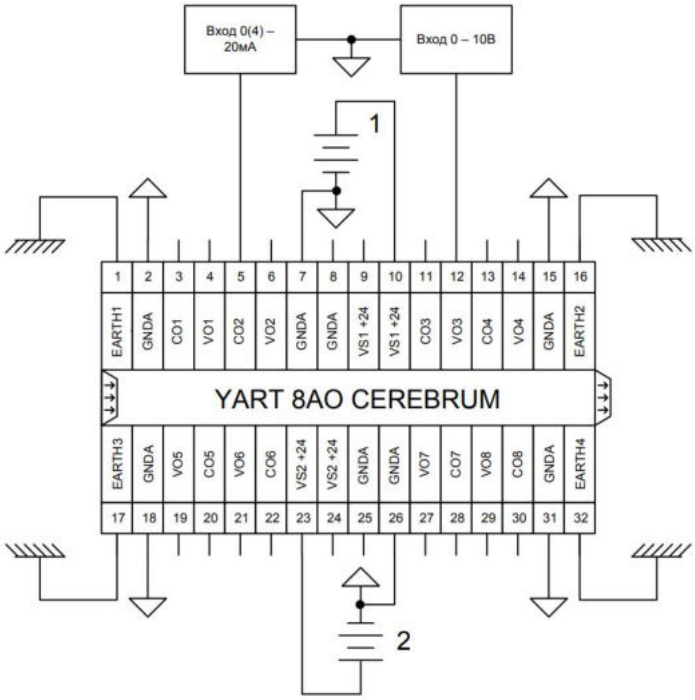


Рисунок 56. Модуль 8АО

Клеммы COx – токовые выходы 0 -20мА

Клеммы Vox – выходы напряжения 0 -10В

Для использования аналогового выхода в программе соответствующий блок необходимо разместить на программном поле, аналогично **п. 9.1, Рис. 31**.

На вход блока (**Рис. 57**) подается число в формате с плавающей точкой в диапазоне 0 -100.0

0.0 соответствует выходному току/напряжению 0мА/0.0В

100 соответствует выходному току/напряжению 20мА/10.0В

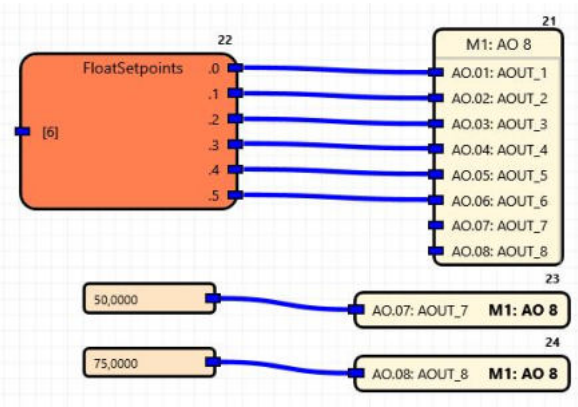


Рисунок 57. Использование модулей аналоговых выходов.

14.3 Дискретные входы

Модуль 16DI содержит 16 входных каналов для дискретных сигналов. Работа входных каскадов модуля осуществляется от внешнего источника питания. При этом все каналы модуля гальванически развязаны от напряжения питания шины расширения.

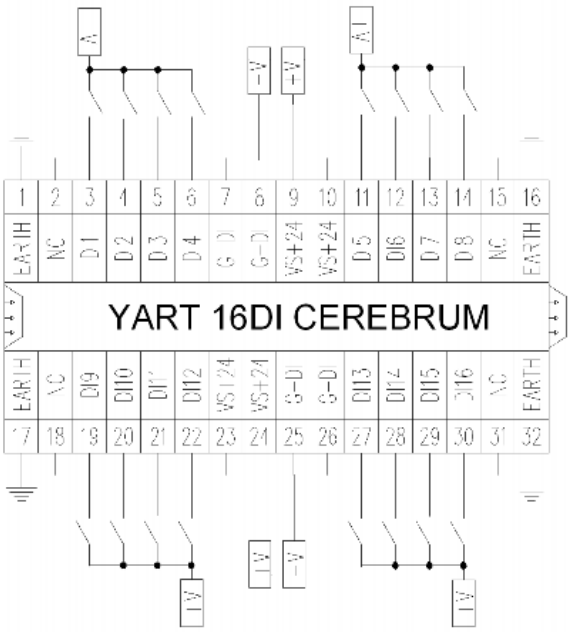


Рисунок 58. Модуль 16DI

Для устойчивой работы модуля достаточно присоединить питание к одной паре клемм.

Клеммы 1, 16, 17, 32 (**Рис. 58**) должны быть присоединены к внутреннему контуру заземления.

Аналогично модулям 8AI 8AO модуль 16DI представлен в виде функционального блока, доступного для размещения на поле программы (**Рис. 59**).

Блоки допускается размещать в любом месте программы.

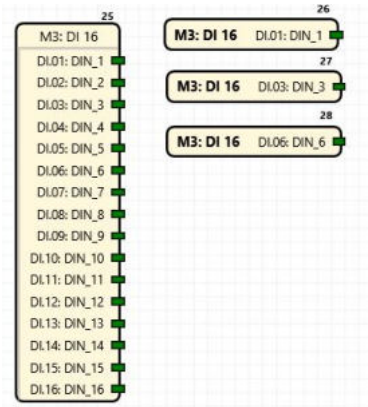


Рисунок 59. Блоки дискретного ввода

14.4 Дискретные выходы

Модуль 16DO содержит 16 выходных каналов с максимальной токовой нагрузкой 1А. Для каждого выходного блока из восьми линий (**Рис. 60**) предусмотрен отдельный вход питания.

При этом все каналы модуля гальванически развязаны от напряжения питания шины расширения.

Выходы модуля имеют встроенную защиту от перегрузки.

Клеммы 1, 16, 17, 32 (**Рис. 61**) должны быть присоединены к внутреннему контуру заземления.

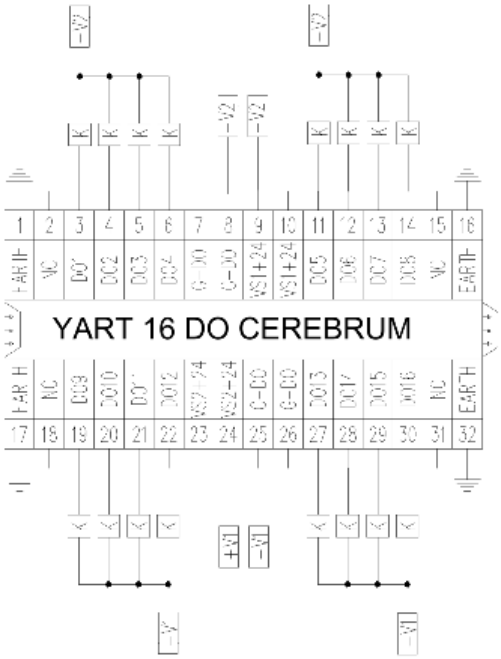


Рисунок 60. Модуль 16DO

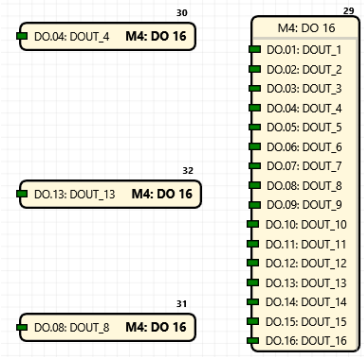


Рисунок 61. Функциональные блоки DO

Значения дискретных выходов задаются при помощи функциональных блоков DO (**Рис. 61**). Данные блоки могут быть размещены в любом месте программы. Компилятор YartStudio автоматически проверяет однократное использование каждого выхода.

При многократной попытке записи в один и тот же канал будет сгенерирована ошибка компиляции.

15. Транспортировка и хранение

Контроллеры транспортируются в заводской упаковке в транспортной таре любым видом транспорта с защитой от дождя и снега. Крепление тары в транспортных средствах должно производиться согласно правилам, действующим на соответствующих видах транспорта.

Условия транспортирования должны соответствовать условиям 5 по ГОСТ 15150-69 при температуре окружающего воздуха от минус 40 до 50 °С с соблюдением мер защиты от ударов и вибраций.

Пребывание в условиях транспортирования – не более 3 месяцев.

Условия хранения в заводской упаковке на складе изготовителя и потребителя должны соответствовать условиям 1 по ГОСТ 15150-69. Наличие в воздухе агрессивных примесей не допускается.

После транспортирования при отрицательных температурах контроллеры перед включением необходимо выдержать в нормальных условиях не менее 24 ч

16. Гарантийные обязательства

Изготовитель гарантирует соответствие контроллера требованиям ТУ при соблюдении условий эксплуатации, транспортирования, хранения и монтажа.

Гарантийный срок– 36 месяцев со дня продажи.

В случае выхода контроллера из строя в течение гарантийного срока при соблюдении пользователем условий эксплуатации, транспортирования, хранения и монтажа предприятие-изготовитель обязуется осуществить его бесплатный ремонт или замену.