МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)
КАФЕДРА САУ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «программируемые логические контроллеры и промышленные сети»

Конфигурирование ПЛК FASTWEL I/O

Студенты гр. 6492 _____ Мурашко А.С.
_____ Огурецкий Д.В.

Преподаватель _____ Филатова Е.С.

Санкт-Петербург
2019

Цель работы — изучение основ конфигурирования и программирования ПЛК FASTWELI/O.

Основные сведения:

Процесс создания конфигурации контроллера FASTWELI/O подразумевает определение состава модулей контроллера, настройку их свойств и создание сигналов.

Первым шагом при конфигурировании контроллера является определение количества и типа модулей, необходимых для выполнения поставленной задачи. Схема соединений контроллера, для которого будет создаваться учебный проект, представлена на рис. 1.

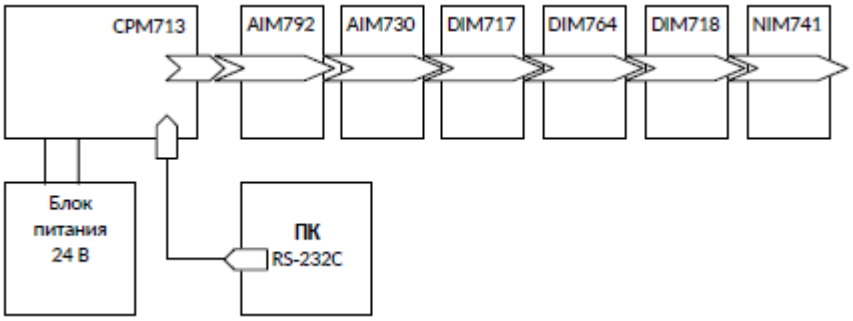


Рис. 1. Конфигурация ПЛК на стенде

Краткое описание модулей, представленных на стенде в лаборатории.

AIM792 1: 8-канальный модуль аналогового ввода

AIM730 2: 2-канальный модуль аналогового вывода

DIM717 3: 8-канальный модуль дискретного ввода

DIM764 4: Универсальный модуль дискретного ввода (8 каналов)

DIM718 5: 8-канальный модуль дискретного вывода

NIM741 6: Модуль универсального асинхронного адаптера последовательного интерфейса RS-485

Создание и сохранение проекта для контроллеров Fastwel I/O серии CPM713.

Запускаем среду разработки CoDeSys:

1)В меню Файл выбираем команду Создать и в появившейся диалоговой панели Настройка целевой платформы выбираем опцию, соответствующую типу используемого контроллера (CPM713), в выпадающем списке Конфигурация. На экран выводится диалоговая панель Новый программный компонент (POU).

При использовании контроллеров серии CPM71x должно соблюдаться следующее соответствие между типом контроллера и выбираемой опцией:

FastwelCPM711CANopen Programmable Controller

для контроллера CPM711;

FastwelCPM712 MODBUS RTU/ASCII Programmable Controller

Для CPM712;

FastwelCPM713 MODBUS TCP Programmable Controller

для CPM713.

2)В меню Файл выбираем команду Сохранить и вводим имя файла проекта CPM713_tutorial_6492 в появившейся диалоговой панели и нажимаем кнопку Сохранить.

3) На вкладке POU и в области редактирования кода программы вводим единственный пустой оператор ";", который позволит немедленно транслировать (построить) проект командой Проект–Компилировать все.

Таким образом, создан проект для платформы, соответствующей типу используемого контроллера, и имеется возможность приступить к созданию конфигурации задач, настройке параметров контроллера, созданию конфигурации модулей ввода-вывода.

Создание и конфигурирование циклической задачи.

Задача является ресурсом контроллера, который предоставляет программам, входящим в приложение, процессорное время.

Создадим циклическую задачу с периодом цикла 10 мс, из которой будет вызываться основная программа PLC_PRG.

Для создания задачи:

1.Добавим задачу в контекстном меню Конфигурации. В древовидном списке Taskconfiguration появляеттся элемент NewTask.

2.В поле Имя вкладки Свойства задачи вводим имя задачи MainTask, а в поле Свойства: Интервал (напр. t#200ms) – требуемый период цикла T#10ms, после чего выбираем команду Добавить вызов программы в появившемся контекстном меню. Имя элемента NewTask в древовидном списке изменяется на MainTask, и под ней появится подчиненный элемент, представляющий вызов программы из данной задачи.

3. Нажимаем кнопку ..., расположенную справа от поля Вызов программы, и выбираем имя вызываемой программы в диалоговой панели Помощника ввода (Inputassistant), сняв флажки WithArguments (с аргументами) и Structured (структурированное представление).

4.При загрузки проекта в контроллер, в контроллере будет циклически, с периодом 10 мс, вызываться программа PLC_PRG, выполняющая единственный ничего не делающий оператор ";".

Создание символических имен для каналов модуля DIM718

Для задания начального состояния каналов модуля DIM718 необходимо создать переменные, чьи значения будут определять состояние каждого канала во время работы приложения, и установить для них начальные значения. Указанные переменные будут связывать разрабатываемую программу с окружением. Окружением программы являются каналы модулей ввода-вывода, входящих в состав контроллера, и коммуникационные объекты внешней сети.

Слева от надписи AT %QX3.8 вводим символическое имя канала relay_out1 в появившееся поле ввода и нажимаем клавишу Enter.

Слева от надписи AT %QX3.9 вводим символическое имя канала relay_out2 в появившееся поле ввода и нажимаем клавишу Enter.

Аналогично relay_out3 для третьего канала DIM718.

Таким образом, созданы три глобальные выходные переменные, ссылающиеся на выходные каналы модуля DIM718.

Создание обработчика системного события OnPowerOn для фиксации начального состояния выходных каналов

Для того, чтобы сохранить указанные логические состояния на выходных каналах модуля, требуется задать начальные значения переменным relay_out1 и relay_out2, ссылающимся на данные каналы.

Установка начальных значений для глобальных выходных переменных, созданных для каналов с символическими именами, может быть осуществлена при помощи пользовательской функции обработки системного события OnPowerOn, которая будет вызываться однократно при включении питания контроллера (или после сброса) перед запуском приложения.

Для создания пользовательской функции обработки системного события OnPowerOn:

Выбираем элемент Системные события в древовидном списке Конфигурация задач. В таблице событий Системные события устанавливаем флажок слева от имени события OnPowerOn, после чего а ячейке OnPowerOn: вызываемый POU, вводим имя функции InitOutputs, которая будет вызываться по событию OnPowerOn.

После кнопка Create POU ... станет активной (доступной для нажатия).

Нажимаем кнопку Создать POUINITOUTPUTS. Имя функции InitOutputs появится в списке программных единиц во вкладке POUs, а кнопка Create POU перестанет быть активной.

Дважды щелкаем левой кнопкой мыши над названием функции

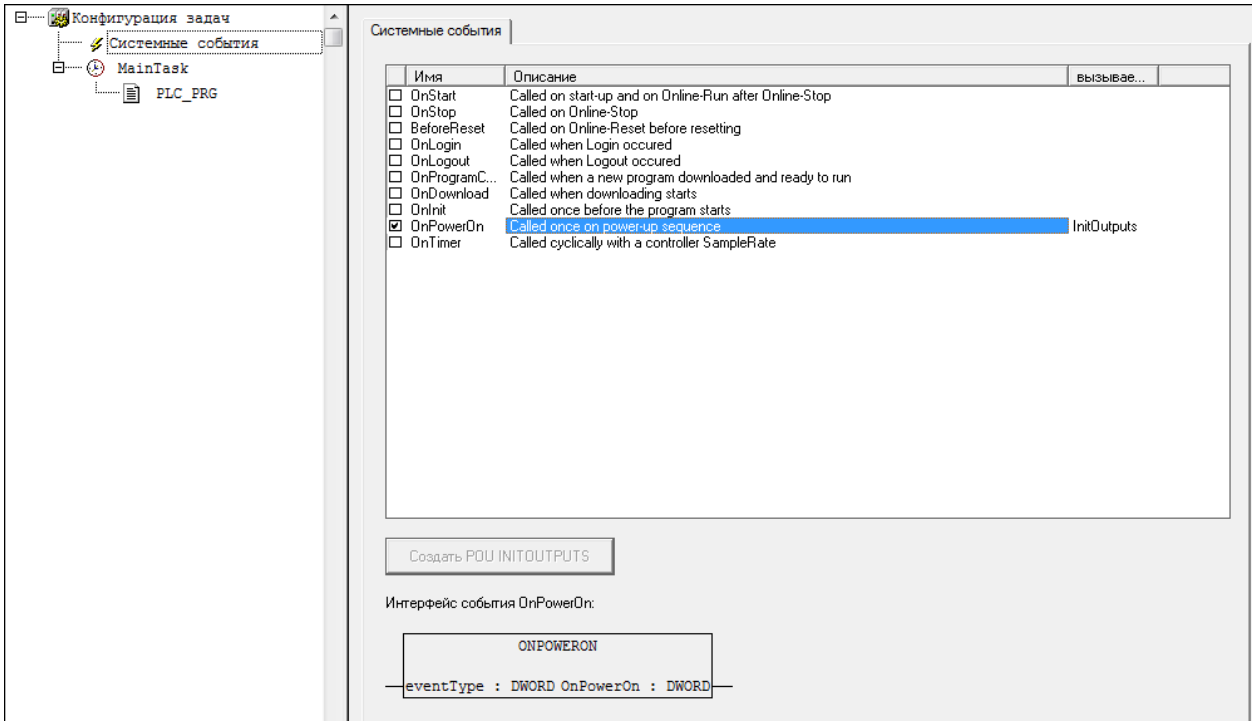
InitOutputsв списке программных единиц. На экран будет выведено окно

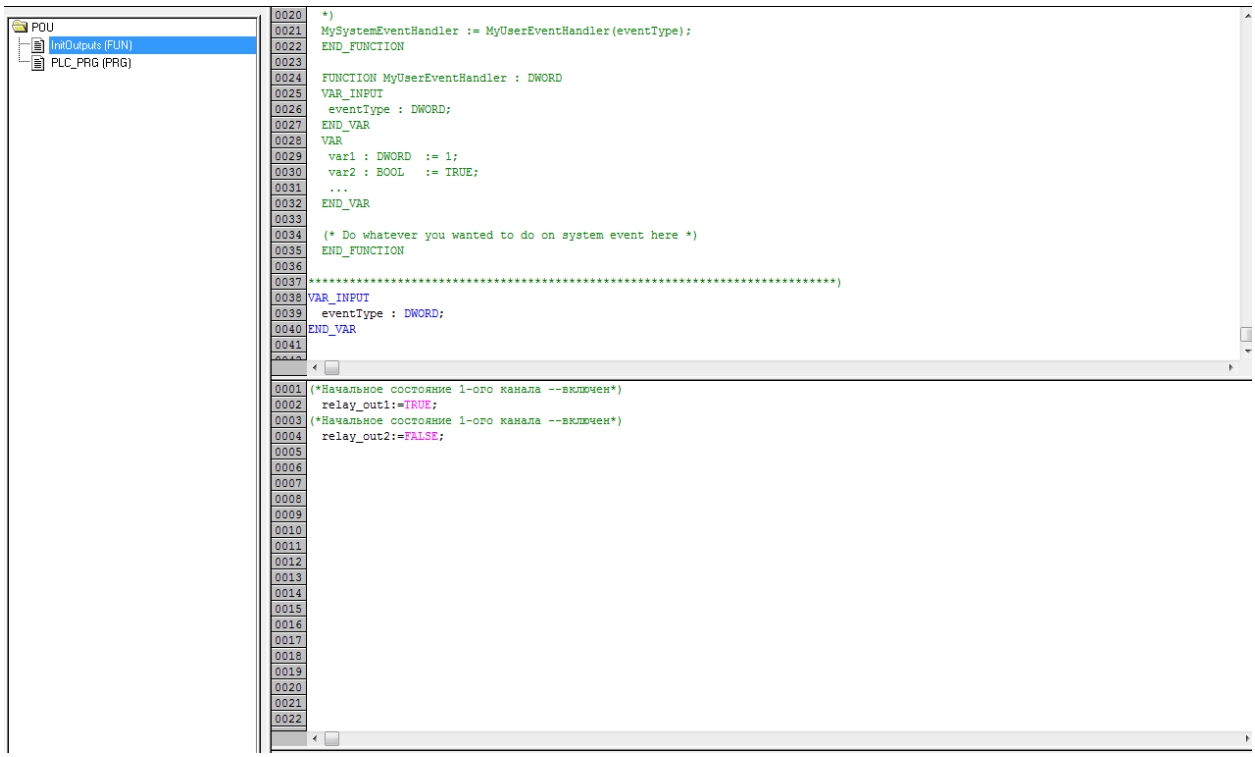
редактора исходного текста ST, содержащее заготовку функции InitOutputs.

В области редактирования кода функции (область реализации) вводим следующие строки:

```
(* Начальное состояние 1-го канала -- включен *)
relay_out1 := TRUE;
(* Начальное состояние 2-го канала -- выключен *)
relay_out2 := FALSE;
```

Компилируем проект.





Создание и конфигурирование ациклической задачи

Создадим ациклическую задачу, которая по нажатию желтой кнопки на стенде будет инвертировать третий бит дискретного вывода модуля DIM718.

1. Создаем программу Blink, на вкладке POU. Она будет содержать

одну команду:

relay_out3:= NOT relay_out3;

2. В конфигурации контроллера создаем символическое имя для 0-го входного канала модуля DIM 717 (InYellowButton) – к нему подключена желтая кнопка. Так мы создатv глобальную переменную, которая будет источником события для ациклической задачи.

3. Добавляем задачу в контекстном меню Конфигурации задач. В поле Имя вкладки Свойства задачи вводим имя задачи BlinkTask, а в поле Свойства: Тип – по событию.

4. Нажимаем кнопку ..., расположенную справа от поля Вызов программы, и выбираем имя вызываемой программы Blink в диалоговой панели Помощника ввода (Inputassistant), сняв флажки WithArguments (с аргументами) и Structured (структурированное представление).

Проверяем работоспособность приложения.

Вводим в область переменных VAR для программы PLC_PRG, ассоциированной с ациклической задачей New_Task следующие строки:

```
adress1: POINTER TO F_TASK_INFO;
answer :F_TASK_INFO;
```

Переменная adress1 является указателем на переменную answer, в которую

функция F_IecTasks_getInfo запишет диагностическую информацию о задаче.

В тело программы пишем следующие команды:

```
adress1:=ADR(answer); (* Получение адреса переменной answer*)
F_IecTasks_getInfo(adress1,65535); (*Вызов функции F_IecTasks_getInfo*)
answer:=adress1^; (*Чтение значения ячейки с адресом adress1*)
```

Использование ресурса VAR_CONFIG.

Функциональные блоки могут включать в себя декларации входных и выходных переменных с недоопределенными ссылками на образ процесса в форме AT %I* или AT %Q*. Указанные ссылки должны быть доопределены в проекте в ресурсе VAR_CONFIG при использовании подобных блоков в приложении.

1. Создаем функциональный блок на вкладке POU с именем inout. В область локальных переменных VAR вводим:

in AT%I* :bool:=TRUE;

out AT%Q* :bool;

2. Переходим в программу PLC_PRG и в область локальных переменных

VAR добавляем:

fb1 :inout;

fb2 :inout;

Таким образом мы создали два экземпляра функционального блока inout, которые мы можем использовать в программе. Но для их использования, необходимо доопределить конкретные адреса для переменных блока. Это делается в ресурсе VAR CONFIG.

3. Переходим на вкладку Ресурсы/Глобальные переменные/Variable_Configuration, и в окне редактирования конфигурационных переменных вводим:

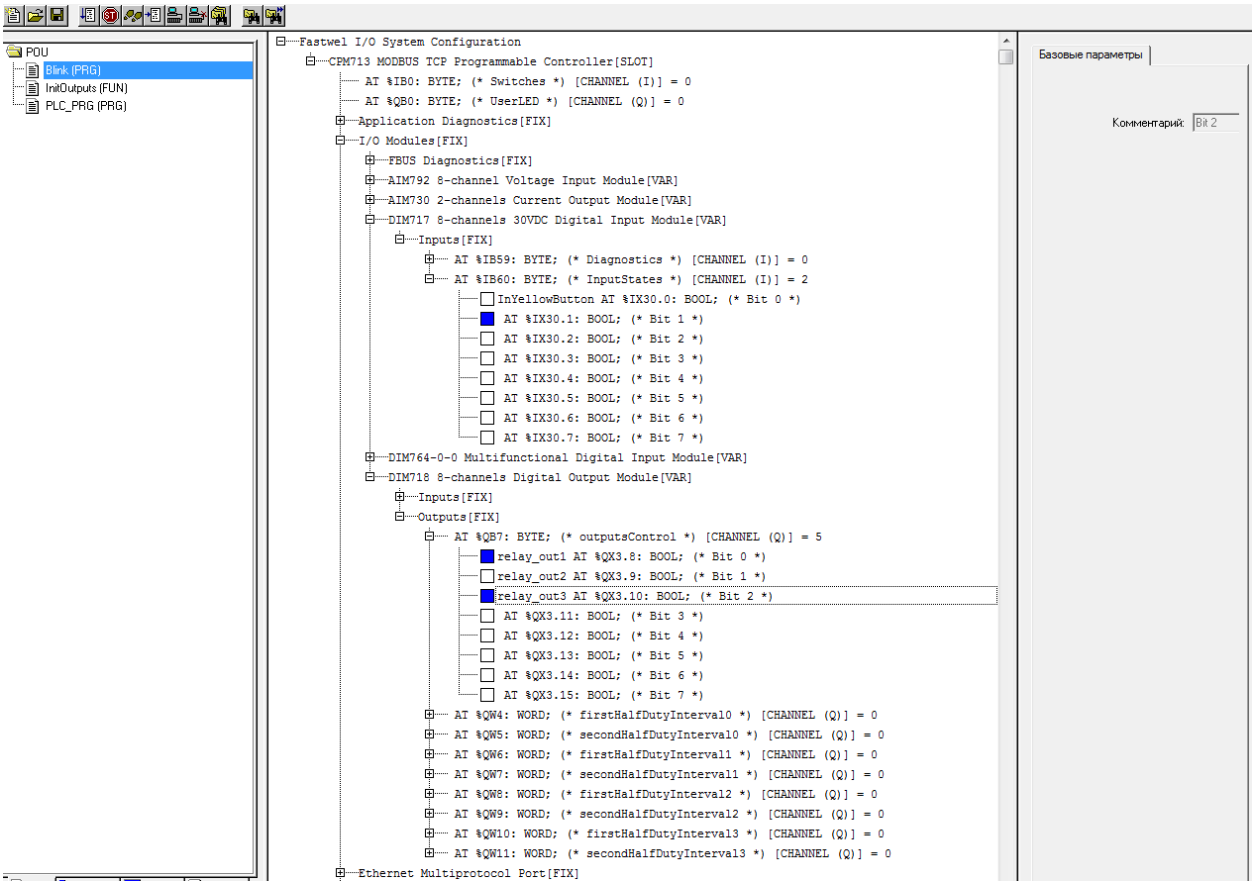
PLC_PRG.fb1.in AT%I____:bool;

PLC_PRG.fb1.outAT%Q_____ :bool;

PLC_PRG.fb2.in AT%I_____ :bool;

PLC_PRG.fb2.out AT%Q_____ :bool;

Таким образом мы задали разные адреса для переменных каждого функционального блока.



Вывод: научились настраивать конфигурацию модулей ПЛК, создавать символические имена для каналов модулей и использовать глобальные переменные. А также научились создавать и конфигурировать циклические и ациклические задачи. Ациклическая задача всегда приоритетней циклической.