

# Программируемые логические контроллеры PRO-Logic

Краткое руководство по настройке и  
программированию



## Оглавление

1. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ .....	3
2. СТРУКТУРА И КОМПОНЕНТЫ ПРОГРАММЫ И КОНТРОЛЛЕРА.....	3
2.1. ПРОГРАММНЫЕ БЛОКИ .....	3
2.2. КОМПОНЕНТЫ ПРОГРАММЫ И КОНТРОЛЛЕРА .....	4
2.3. КАРТА РЕГИСТРОВ MODBUS.....	6
3. СОЗДАНИЕ ПРОЕКТА И ПРОГРАММИРОВАНИЕ.....	6
4. ПРОВЕРКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ В СИМУЛЯТОРЕ .....	12
5. ПОДКЛЮЧЕНИЕ КОНТРОЛЛЕРА К ПК. НАСТРОЙКА.....	13
6. ЗАГРУЗКА И ВЫГРУЗКА ПРОЕКТА .....	15
7. ПУСК И ОСТАНОВ ПРОГРАММЫ .....	15
8. ОНЛАЙН-МОНИТОР .....	15
9. ПОМОЩЬ ПО НАСТРОЙКЕ И ПРОГРАММИРОВАНИЮ .....	16

## 1. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

Для настройки и программирования контроллеров PRO-Logic требуется скачать и установить бесплатное программное обеспечение **PRO-Logic master**. Программа бесплатная и доступна для скачивания на сайте <https://ekfgroup.com/> на странице продукта в разделе «Документация и ПО».



## 2. СТРУКТУРА И КОМПОНЕНТЫ ПРОГРАММЫ И КОНТРОЛЛЕРА

### 2.1. ПРОГРАММНЫЕ БЛОКИ

Проект состоит из программных блоков (главные программы, подпрограммы, программы прерывания). Суммарное максимальное количество программных блоков – 31.

#### Главная программа

Главная программа (main program) - это программный блок, который выполняется при переводе переключателя на лицевой панели прибора в состояние RUN.



#### Подпрограмма

Подпрограмма (subprogram) – это программный блок, который вызывается другим программным блоком с помощью специальных команд. Подпрограмма может иметь свои собственные входные и выходные параметры (до 8 входных и 3 выходных параметров).

#### Программа прерывания

Программа прерывания (interrupt program) – это программный блок, который выполняется по специальному условию. Когда в системе происходит событие (условие) прерывания, выполнение основных программ и подпрограмм прерывается, выполняется соответствующая программа прерывания, и система возвращается к нормальному выполнению программы.

## 2.2. КОМПОНЕНТЫ ПРОГРАММЫ И КОНТРОЛЛЕРА

Для хранения, обработки и обмена информацией ПЛК использует различные типы компонентов: X, Y, T, C, M, SM, LM, S, AI, AQ, TV, CV, V, LV, SV, P. Это переменные, в которые можно записать информацию определенного типа данных.

### Типы данных

Тип данных	Формат	Объем	Диапазон значений
BOOL	bit	1 bit component	1(ON). 0(OFF)
INT	integer with sign	16 bits,1 register component	- 32768~32767
DINT	long integer with sign	32 bits,2 register components	-2147483648~2147483647
REAL	floating point	32bits,2 register components	-3.402823e+38~3.402823e+38
CHAR	character string	1 character occupy one byte	

### Соответствие компонентов и типов данных

Тип данных	Компоненты															
	X	Y	T	C	M	SM	LM	S	AI	AQ	TV	CV	V	LV	SV	P
BOOL																
INT	constant								AI	AQ	TV	CV	V	LV	SV	P
DINT	constant										TV	CV	V	LV	SV	
REAL	constant												V	LV		
CHAR	constant												V	LV		

### Константы

Тип константы	Пример	Диапазон значений
16 bits integer with sign	1234. -7890	-32768~32767
32 bits integer with sign	12345678. -9876543	-2147483648~2147483647
16 bits constant in hexadecimal	0x2EF8. 0x9A12	0x0~0xFFFF
32 bits constant in hexadecimal	0xA76DCFE9	0x0~0xFFFFFFFF
floating point constant in single precision	3.1415926. -0.02341	-3.402823e+38~3.402823e+38

### Битовые компоненты

Компонент	Имя	Диапазон	Доступ	Описание
X	External input relay	X0~X1023	read	Соответствуют состоянию дискретных входов ПЛК
Y	External output relay	Y0~Y1023	read/write	Соответствуют состоянию дискретных выходов ПЛК
M	Auxiliary relay	M0~M12287	read/write	Вспомогательные переменные
T	Timer	T0~T1023	read/write	Переменные, состояние которых зависит от выполнения соответствующих команд-таймеров

Компонент	Имя	Диапазон	Доступ	Описание
C	Counter	C0~C255	read/write	Переменные, состояние которых зависит от выполнения соответствующих команд-счетчиков
SM	System status bit	SM0~SM215	all be read/some be wrote	Системные переменные
S	Step relay	S0~S2047	read/write	Переменные для шагового управления программой
LM	Local relay	LM0~LM31	read/write	Внутренние переменные для подпрограмм

### Регистровые компоненты

Компонент	Имя	Диапазон	Доступ	Описание
AI	Analog input register	AI0~AI255	read	Соответствуют состоянию аналоговых входов ПЛК
AQ	Analog output register	AQ0~AQ255	read/write	Соответствуют состоянию аналоговых выходов ПЛК
V	Internal data register	V0~V14847	read/write	Вспомогательные переменные
TV	Current value of timer	TV0~TV1023	read/write	Текущее время таймеров
CV	Current value of counter	CV0~CV255	read/write	Текущее время счетчиков
SV	System register	SV0~SV154	all be read/some be wrote	Системные регистры
LV	Local register	LV0~LV31	read/write	Внутренние переменные для подпрограмм
P	Indexed addressing point	P0~P29	read/write	Переменные для индексирования

### Хранение и использование данных 32 бит

Тип данных DINT.REAL имеет длину 32 бита, но один регистр занимает длину 16 бит, поэтому для хранения 32-битных данных необходимы 2 непрерывных адресных регистра. При хранении 32-битных данных в начале идет младшее слово, затем старшее слово. Например, 32-битные

целочисленные данные 0xA76DCFE9 хранятся в регистрах V0V1, тогда 0xCFE9 хранится в V0, 0xA76D и хранится в V1.

В зависимости от того, какой тип данных используется, необходимо использовать соответствующий тип команды (инструкции). Команды, начинающиеся на «D.» (например, D.MOV) – это 32-битные команды. Команды не имеющие в начале «.D» - это 16-битные команды.

### 2.3. КАРТА РЕГИСТРОВ MODBUS

#### Битовые компоненты

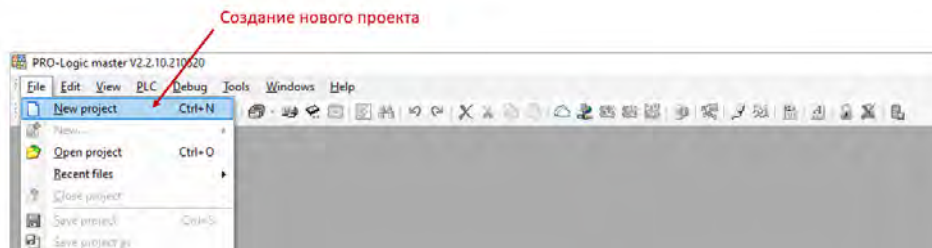
Компонент	Диапазон компонентов	Функция	Команды	Адрес	
				HEX	DEC
X	X0~X1023	R	1,2,5,15	0x0000~0x03FF	0~1023
Y	Y0~Y1023	R/W	1,2,5,15	0x0600~0x09FF	1536~2559
M	M0~M12287	R/W	1,2,5,15	0x0C00~0x3BFF	3072~15359
T	T0~T1023	R/W	1,2,5,15	0x3C00~0x3FFF	15360~16383
C	C0~C255	R/W	1,2,5,15	0x4000~0x40FF	16384~16639
SM	SM0~SM215	R/W	1,2,5,15	0x4200~0x42D7	16896~17111
S	S0~S2047	R/W	1,2,5,15	0x7000~0x77FF	28672~30719

#### Регистровые компоненты

Компонент	Диапазон компонентов	Функция	Команды	Адрес	
				HEX	DEC
CR	CR0~CR255	R/W	3,4,6,16	0x00~0xFF	0~255
AI	AI0~AI255	R	3,4,6,16	0x0000~0x00FF	0~255
AQ	AQ0~AQ255	R/W	3,4,6,16	0x0100~0x01FF	256~511
V	V0~V14847	R/W	3,4,6,16	0x0200~0x3BFF	512~15359
TV	TV0~TV1023	R/W	3,4,6,16	0x3C00~0x3FFF	15360~16383
CV	CV0~CV255	R/W	3,4,6,16	0x4000~0x40FF	16384~16639
SV	SV0~SV900	R/W	3,4,6,16	0x4400~0x4784	17408~18308

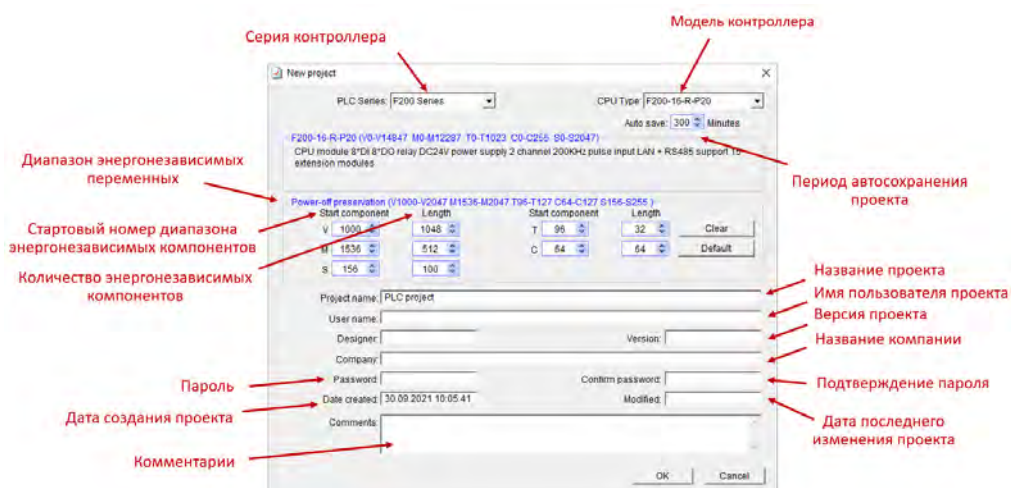
### 3. СОЗДАНИЕ ПРОЕКТА И ПРОГРАММИРОВАНИЕ

Запустите PRO-Logic master, создайте новый проект.

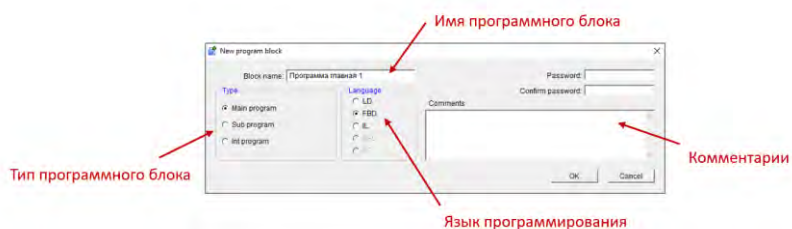


Выберите серию и модель контроллера, период автосохранения проекта и диапазон энергонезависимой памяти контроллера. Укажите имя проекта. При необходимости можно указать автора проекта, компанию, дату создания, пароль для защиты проекта и комментарий.

Нажмите ОК.



После создания проекта автоматически появится окно для создания первого программного блока.



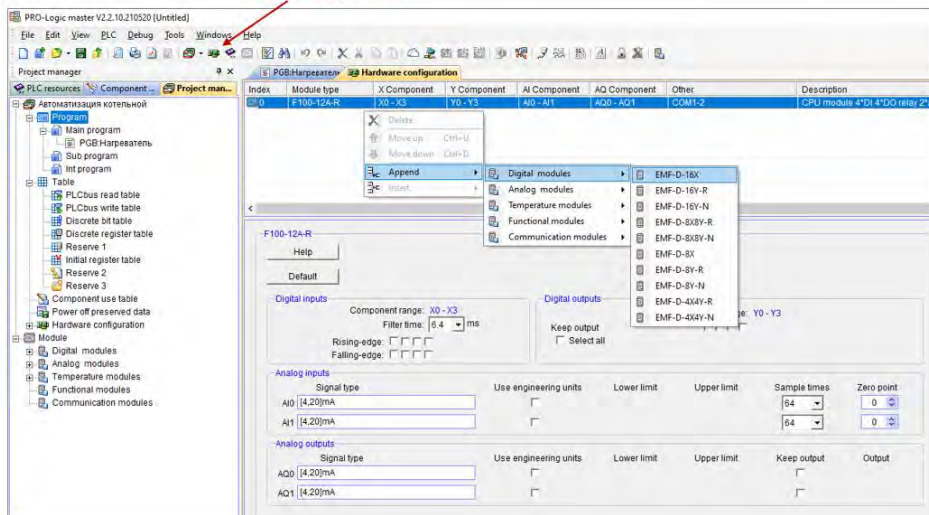
Впишите имя блока и выберите его тип (главная программа/подпрограмма/программа прерывания) и язык программирования (LD, FBD, IL). Рекомендуем начинать с создания главной программы (main program) и использовать язык программирования FBD (это наиболее распространенный и простой язык программирования контроллеров). При необходимости можно указать пароль для защиты блока и комментарий. Нажмите ОК.

### Конфигурация оборудования

Для дополнительной настройки оборудования зайдите в раздел конфигурации оборудования («Hardware configuration»).



Конфигурация  
оборудования



В этом разделе можно задавать настройки контроллера, указать подключаемые модули расширения, настроить свойства дискретных и аналоговых входов/выходов и т.д.

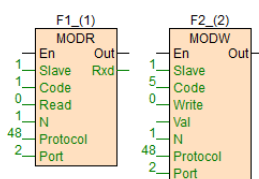
### ВНИМАНИЕ!

Если в проекте будут использоваться удаленные модули ввода/вывода PRO-Logic REMF, подключаемые к ПЛК по интерфейсу RS-485, указывать их в окне «Hardware configuration» не требуется.

Контроллер



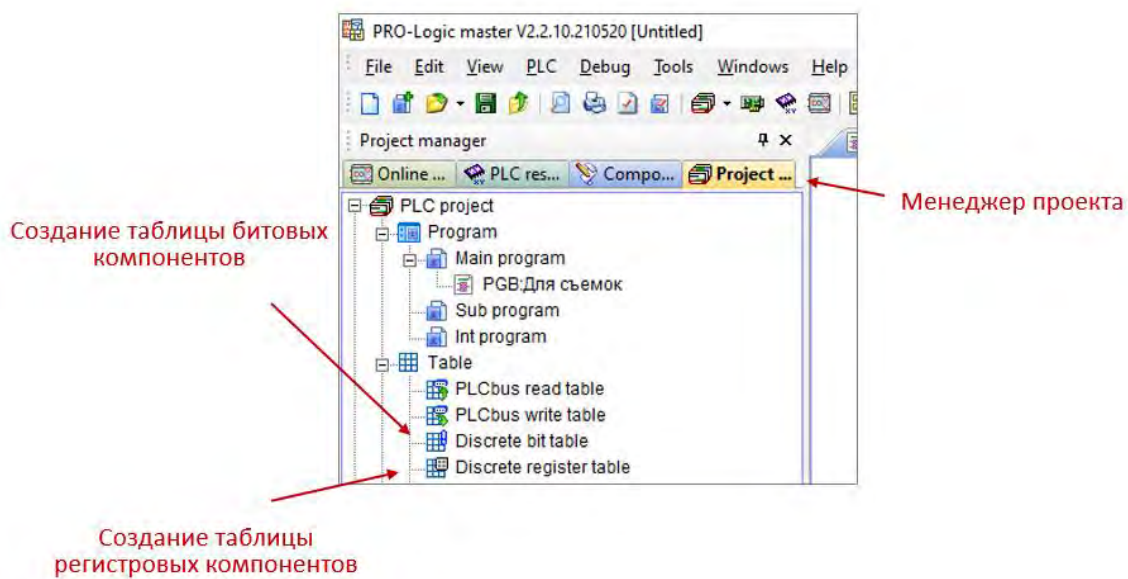
Для обмена данными между ПЛК PRO-Logic и модулями REMF следует использовать инструкции MODR (чтение) и MODW (запись) при написании программы для контроллера.



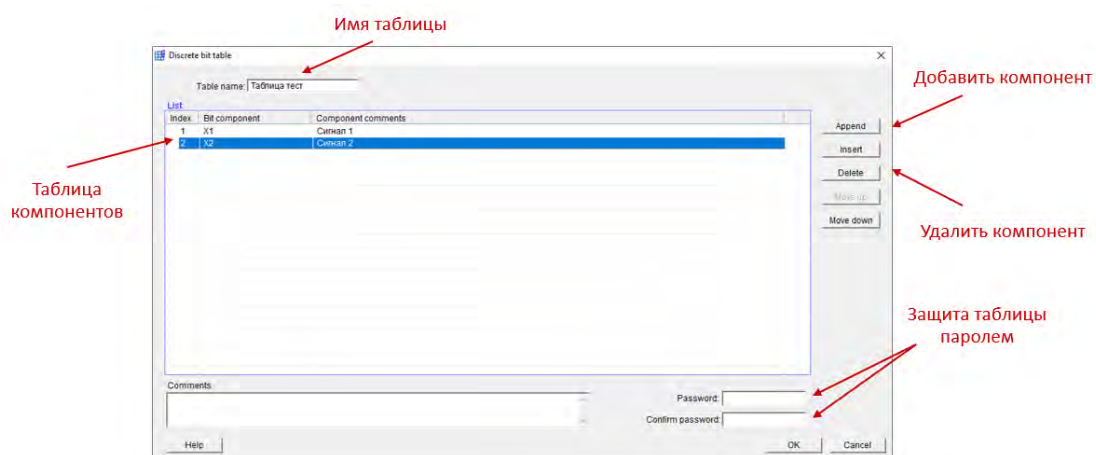


## Таблицы компонентов

Для добавления комментариев компонентов (битовых и регистровых) зайдите в менеджер проекта и откройте соответствующие разделы.

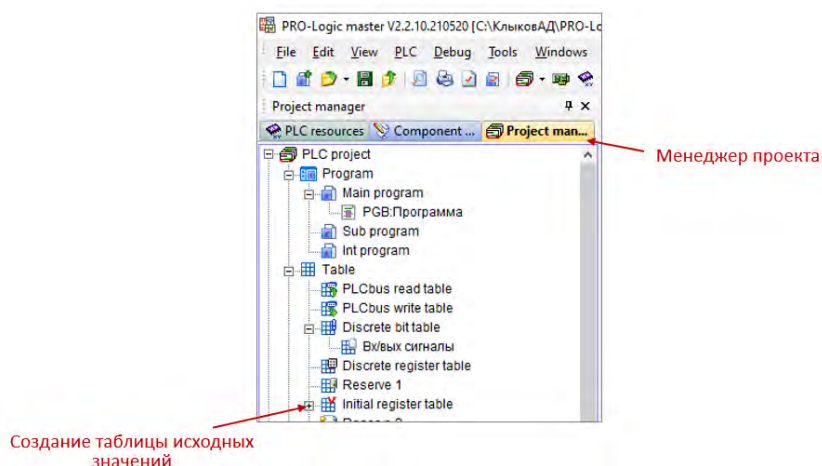


В открывшемся окне можно добавлять, удалять, задавать комментарии для компонентов.

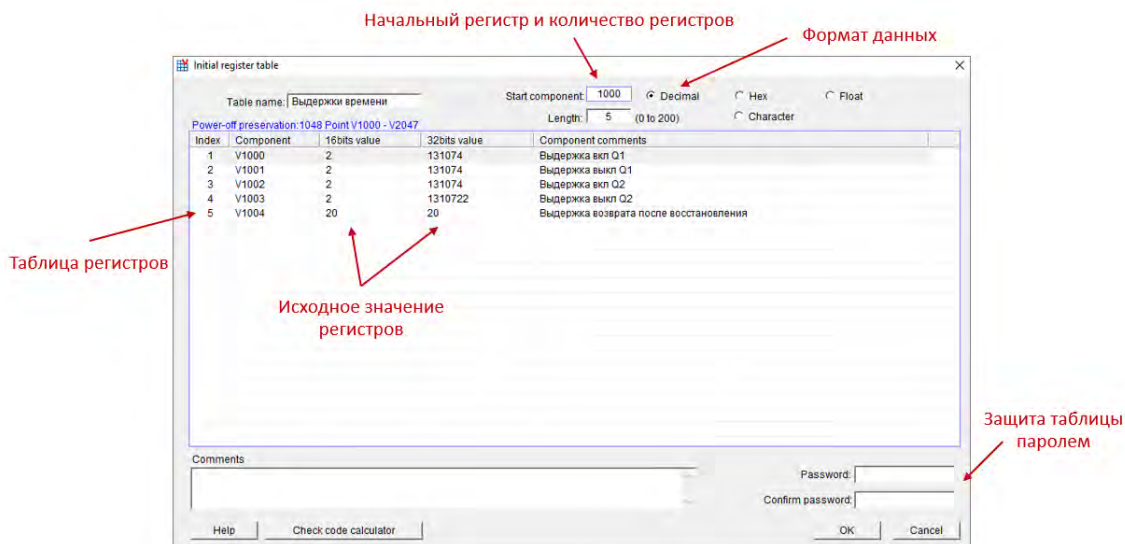


## Исходные значения компонентов

Для задания исходных значений регистровых компонентов V зайдите в менеджер проекта и откройте соответствующий раздел.

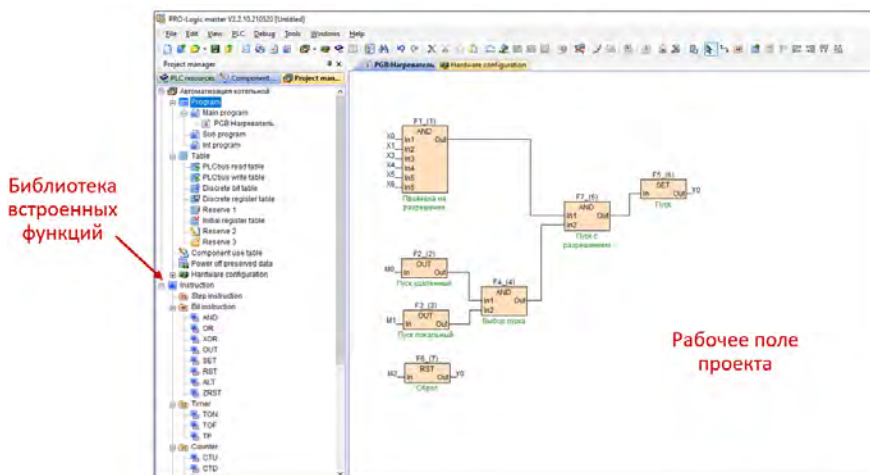


Откроется таблица исходных значений компонентов. Выберите формат данных, начальный регистр и количество компонентов, которые нужно отобразить. После этого можно вписать исходное значение для каждого компонента.



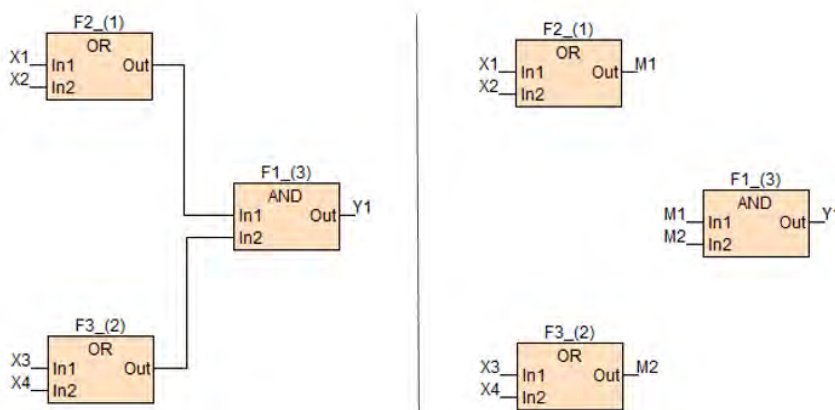
## Написание и сохранение программы

Напишите программу на выбранном языке программирования, используя библиотеку встроенных функций. Для ознакомления с функцией нажмите на нее левой клавишей мыши и нажмите F1 для открытия руководства.



Обратите внимание, что при программировании на языке FBD создавать связи входов и выходов функциональных блоков можно 2 способами:

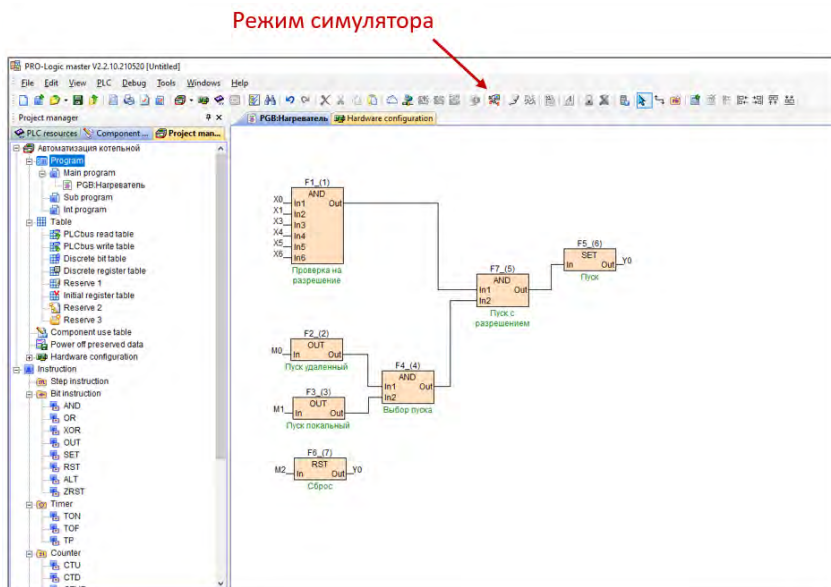
- Соединять их линиями
- Задавать входные и выходные компоненты



После написания программы сохраните проект, нажав Ctrl+S и выбрав путь сохранения.

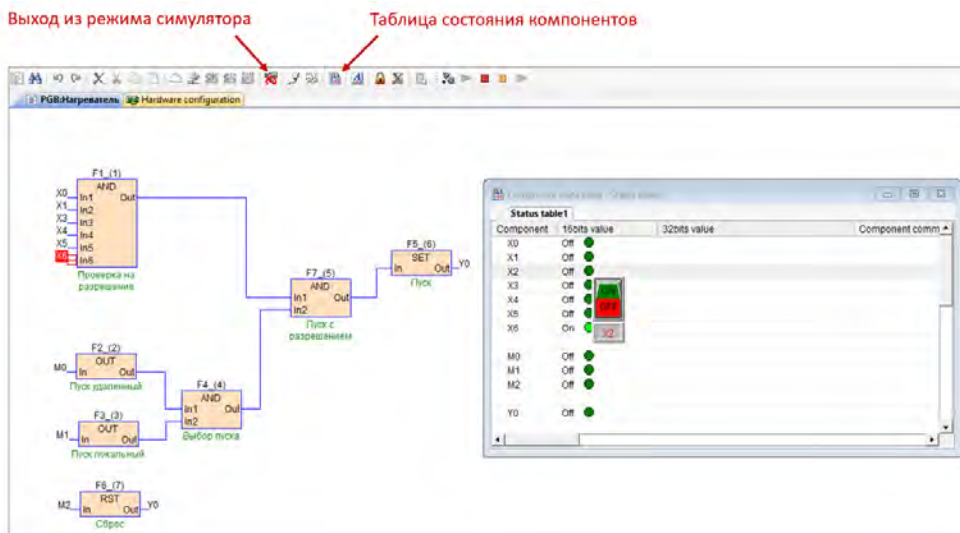
#### 4. ПРОВЕРКА И ТЕСТИРОВАНИЕ ПРОГРАММЫ В СИМУЛЯТОРЕ

После написания проекта (перед загрузкой его в ПЛК) программу следует протестировать. Для этого в PRO-Logic master предусмотрен встроенный симулятор. Для запуска режима симулятора нажмите клавишу «Run simulator».



Для подачи входных сигналов дважды щелкните на соответствующий компонент и выберите нужное значение. Программа отработает по заданной вами логике в зависимости от состояния компонентов.

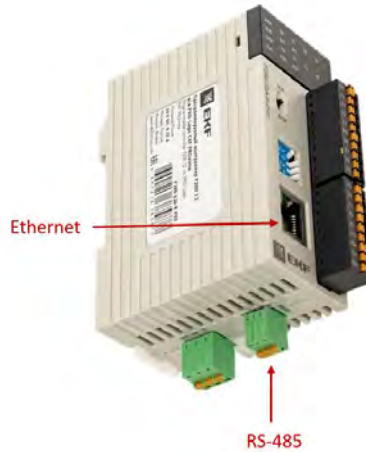
Для табличного отображения сигналов и состояний элементов в режиме симулятора откройте таблицу состояния компонентов («Component state table»). В таблице аналогичным образом можно моделировать необходимые сигналы и следить за выполнением команд и состоянием выходов.



После успешного тестирования программы выйдите из режима симулятора нажав клавишу «Stop simulator».

## 5. ПОДКЛЮЧЕНИЕ КОНТРОЛЛЕРА К ПК. НАСТРОЙКА

Подключите контроллер к ПК через интерфейс RS-485 или Ethernet.



Откройте вкладку «PLC online» и выберите соответствующий способ подключения (COM для подключения через RS-485, TCP/IP для подключения через Ethernet). Выберите номер COM-порта, автоматически определившегося при подключении прибора к ПК.

Для автоматического поиска устройства нажмите «Find», запустится автопоиск модуля.

Если известны сетевые настройки (скорость обмена, формат данных, диапазон адресов) задайте их и нажмите «Online» для ускоренного поиска устройства.

### Сетевые настройки по умолчанию:

**Протоколы:** Modbus RTU, Modbus ASCII (по умолчанию: Modbus RTU)

**Адрес в сети:** 1-256 (по умолчанию: 1)

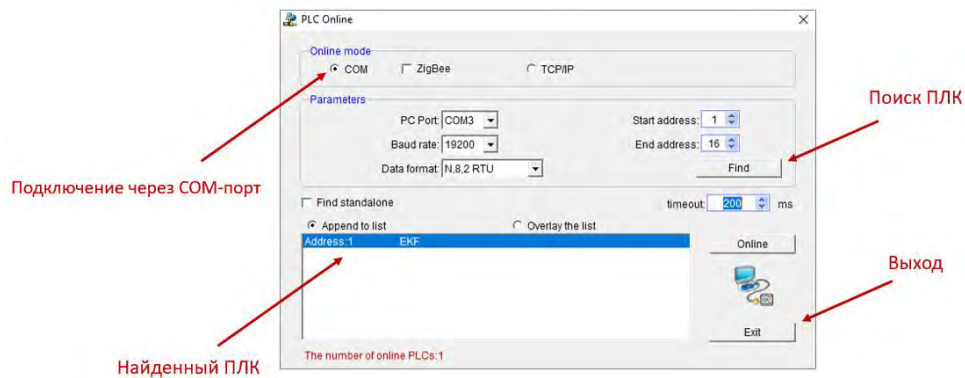
**Скорость:** 2400, 4800, 9600, 19200, 38400, 57600, 115200 (по умолчанию: 19200 бит/с)

**Формат данных:** N,8,2; E,8,1; O,8,1; N,7,2; E,7,1; O,7,1; N,8,1 (по умолчанию: N,8,2)

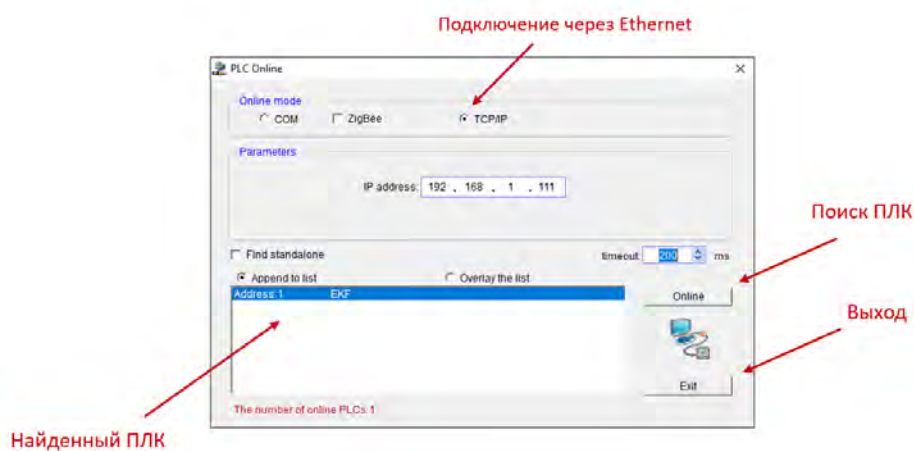
После определения сетевых параметров и нахождения прибора он появится в соответствующем окне.

Для поиска нескольких устройств поставьте отметку «Find standalone».

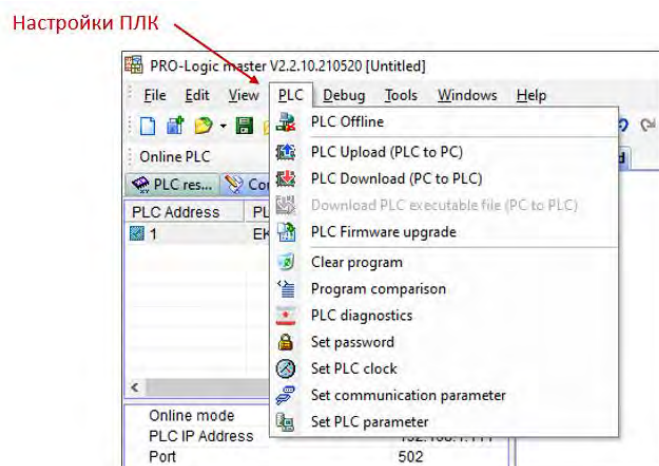
По завершении поиска нажмите кнопку «Exit» для выхода из режима поиска прибора.



При подключении к контроллеру через Ethernet впишите IP-адрес ПЛК (по умолчанию 192.168.1.111). При этом ПК, к которому подключается контроллер, должен находиться с ним в одной сети, т.е. иметь соответствующий IP-адрес (например, 192.168.1.1). Далее нажмите «Online» для поиска контроллера. После нахождения прибора он появится в соответствующем окне. Далее нажмите «Exit» для выхода из режима поиска прибора.



Для настройки сетевых параметров (RS-485, Ethernet), часов реального времени и других параметров устройства необходимо зайти в раздел «PLC».





## 6. ЗАГРУЗКА И ВЫГРУЗКА ПРОЕКТА

При успешном соединении ПК с контроллером на панели инструментов появится возможность загрузить готовый проект или выгрузить уже имеющийся проект в контроллере. Для загрузки проекта в контроллер нажмите клавишу «PLC Download» на панели инструментов. Для выгрузки проекта из контроллера нажмите клавишу «PLC Upload» на панели инструментов.



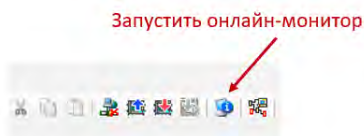
## 7. ПУСК И ОСТАНОВ ПРОГРАММЫ

Для запуска загруженной программы на контроллере подайте на него питание и переведите переключатель на лицевой панели прибора в состояние «RUN». Для остановки программы необходимо перевести переключатель в состояние «STOP».

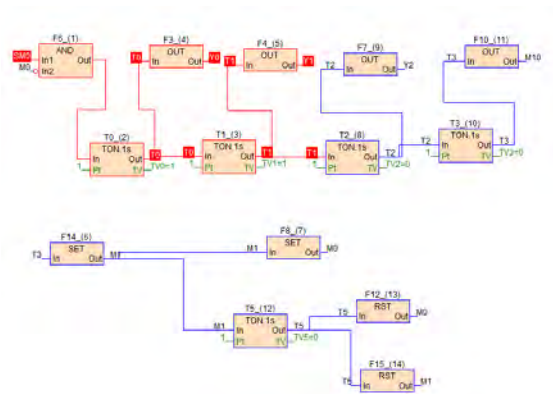


## 8. ОНЛАЙН-МОНИТОР

Есть возможность наблюдать за выполнением программы на контроллере в режиме реального времени. Для этого в PRO-Logic master предусмотрен онлайн-монитор. Для его запуска необходимо подключиться к ПЛК одним из ранее описанных способов, загрузить проект в ПЛК и нажать на клавишу «Start monitor» на панели инструментов.

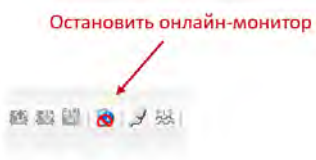


После перевода положения переключателя на лицевой панели прибора в состояние «RUN» на экране ПК будет отображаться выполнение программы.





Для выхода из режима онлайн-монитора нажмите клавишу «Stop monitor» на панели инструментов.



## 9. ПОМОЩЬ ПО НАСТРОЙКЕ И ПРОГРАММИРОВАНИЮ

Для более подробного обучения по программированию контроллеров PRO-Logic используйте подробное руководство, нажав F1 во время работы программного обеспечения PRO-Logic master.

**General declare of the instruction**

1. En enable input. En is the enable input item of the instruction. Only En have electricity (ON), the instruction executed, otherwise not executed.
2. Enio Enable output. Enio is the Enable output item of the instruction, indicate the instruction is executing. When En have electricity (ON) and instruction executed properly then Enio output have electricity (ON), when En have not electricity (OFF) or instruction executed error (a parameter not proper of the instruction) then Enio output have not electricity (OFF). The application instruction in LD/FBD language the great mass of the instruction have Enio Enable output item. All R, instructions have not Enio output item, it will be instead of the ENIO instruction in R language.
3. In LD language: the AND, OR, NOR instructions, will be instead of logic link.
4. 32 bit instruction at 16 bit instruction name "D": indicates use 2 continuous register. Such as ADD, 16 bit addition is ADD, 32 bit addition is D-ADD.
5. 8 bit instruction at 16 bit instruction name plus "LB": indicate only use the low byte of the register. Such as COMM, 16 bit instruction is COMM, 8 bit instruction is COMM.LB.
6. When the parameter items of many instruction which autoOccup several continuous register, pay special attention to them when programming, avoid reusing the register to program execution incorrect.

Note: except C148-C179 are 32 bit register (total 32 entries) PLC other registers (M, AQ, V, SV, LV, TV, CV, FV) all are 16 bit register, one 16 bit register have 2 byte compose, one 32 bit register have 2 continuous 16 bit registers compose.

**Compare switch**

Compare switch used in LD program language dedicated, divide into 16 bit compare instruction, 32 bit compare instruction, floating point compare instruction, low byte compare instruction, high byte compare instruction.

Compare mode have equal to (=), unequal to ( $\neq$ ), greater than (>), greater than or equal to ( $\geq$ ), less than (<), less than or equal to ( $\leq$ ) bit type.

Program example: **Compare** instruction list as follows:

Instruction name	8 bit model	32 bit model	Instruction function	Support language		
				LD	FBD	R
=	LB= HB=	D=	Equal to compare switch. Have 16 bit/32 bit low byte/high byte model.	✓		
$\neq$	LB $\neq$ HB $\neq$	D $\neq$	Unequal to compare switch. Have 16 bit/32 bit low byte/high byte model.	✓		
>	LB> HB>	D>	Greater than compare switch. Have 16 bit/32 bit low byte/high byte model.	✓		
$\geq$	LB $\geq$ HB $\geq$	D $\geq$	Greater than or equal to compare switch. Have 16 bit/32 bit low byte/high byte model.	✓		
<	LB< HB<	D<	Less than compare switch. Have 16 bit/32 bit low byte/high byte model.	✓		
$\leq$	LB $\leq$ HB $\leq$	D $\leq$	Less than or equal to compare switch. Have 16 bit/32 bit low byte/high byte model.	✓		
F=			Floating-point number equal to compare switch.	✓		
F $\neq$			Floating-point number unequal to compare switch.	✓		
F>			Floating-point number greater than or equal to compare switch.	✓		
F $\geq$			Floating-point number greater than or equal to compare switch.	✓		
F<			Floating-point number less than or equal to compare switch.	✓		
F $\leq$			Floating-point number less than or equal to compare switch.	✓		

**Step instruction**

Step instruction list as follows:

Instruction name	8 bit model	32 bit model	Instruction function	Support language		
				LD	FBD	R
ST			Step start.	✓		

В руководстве имеется вся информация, необходимая для работы PRO-Logic master.

Успешных проектов!