

# «САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ УНИВЕРСИТЕТ имени академика С.П. КОРОЛЕВА»



Илюхин В.Н.

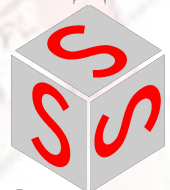
Программирование промышленных логических контроллеров  
«ОВЕН» в системе «CoDeSys»  
Конспект лекций по дисциплине  
«Средства электроавтоматики пневмо- и гидросистем»

# Цель

Обучение эффективной разработке программного обеспечения контроллеров ОВЕН на **CoDeSys**

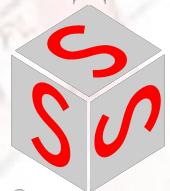
Две основные составляющие:

- Изучение возможностей контроллеров ОВЕН
- Программирование контроллеров ОВЕН



# Содержание

- Программируемые логические контроллеры (ПЛК)
- Контроллеры ОВЕН
- Основные принципы стандарта МЭК 61131-3
- Введение в CoDeSys
- Установка CoDeSys
- Языки и операторы стандарта МЭК 61131-3
- Программные модули (POU)



# Содержание

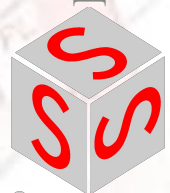
- Что такое библиотека?
- Стандартная библиотека
- Работа с вещественными числами
- Трассировка
- Язык Последовательных Функциональных Диаграмм (SFC)
- Работа с задачами и событиями



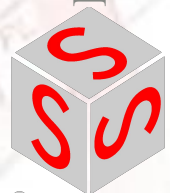
# ЛЕКЦИЯ 1

## ОВЕН ПЛК 100, ПЛК 150 и ПЛК 154

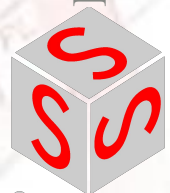




Параметр	ПЛК 100	ПЛК 150	ПЛК 154
<b>Общие характеристики</b>			
Конструктивное исполнение	DIN-рейка, 105 мм		
Степень защиты IP, климатическое исполнение	IP20, -20...+70 °С		
Напряжение питания	<input type="checkbox"/> =24 В <input type="checkbox"/> ~220 В	<input type="checkbox"/> ~220 В	
Потребляемая мощность	6 Вт (без нагрузки)		
Индикация передней панели	светодиодная индикация питания, наличия связи со средой программирования и состояния дискретных входов и выходов		
<b>Ресурсы</b>			
Центральный процессор	32-х разрядный RISC-процессор 200 МГц на базе ядра ARM9		
Объем оперативной памяти	8 МБ		
Объем энергонезависимой памяти хранения программ и архивов	3 МБ		
Размер Retain-памяти	4 кБ (до 16 кБ)		

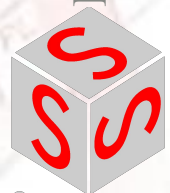


Параметр	ПЛК 100	ПЛК 150, ПЛК 154
<b>Интерфейсы связи</b>		
Интерфейсы	Ethernet 10/100 Mbit   RS-485   RS-232 – 2 шт.   USB-Device	Ethernet 10/100 Mbit   RS-485   RS-232
Скорость обмена по интерфейсам RS-485 и RS-232	настраиваемая, до 115200 bps	
Протоколы	OVEN   Modbus RTU, Modbus ASCII   DCON   Modbus TCP   GateWay (протокол CoDeSys)	
<b>Программирование</b>		
Среда программирования	CoDeSys 2.3.9.6 Rus	
Размер пользовательской программы	ограничен только размерами свободной памяти (около 1 млн. инструкций)	
Интерфейс для программирования и отладки	RS-232, Ethernet, USB Device для ПЛК100	
Подключение при программировании	стандартным кабелем или кабелем, входящим в комплект поставки	



Параметр	ПЛК 100	ПЛК 150	ПЛК 154
<b>Дискретные выходы</b>			
Количество дискретных выходов	<b>варианты исполнения:</b> <input type="checkbox"/> 6 э/м реле (220 В 8 А) <input type="checkbox"/> 12 транзист. ключей	4 э/м реле (220 В 4 А)	
Гальваническая развязка дискретных выходов	1,5 кВ		
<b>Аналоговые выходы</b>			
Количество аналоговых выходов	нет	2	4
Тип выходного сигнала	–	<b>варианты исполнения:</b> <input type="checkbox"/> 0(4)...20 мА <input type="checkbox"/> 0...10 В <input type="checkbox"/> универсальный: 4...20 мА или 0...10 В (переключаемый тип выходного сигнала)	
Встроенный источник питания аналоговых выходов	–	есть, гальванически развязанный (1,5 кВ) от остальной схемы	





Параметр	ПЛК 100	ПЛК 150	ПЛК 154
<b>Дискретные входы</b>			
Количество дискретных входов	8	6	4
Гальваническая развязка дискретных входов	на 1,5 кВ, групповая		
Максимальная рабочая частота дискретного входа	до 10 кГц		
<b>Аналоговые входы</b>			
Количество аналоговых входов	нет	4	
Предел основной приведенной погрешности	—	0,5 %	
Типы поддерживаемых датчиков и входных сигналов	—	<input type="checkbox"/> термосопротивления Pt1000, Ni1000, Pt100, Cu50 (2-х проводная схема) <input type="checkbox"/> ток 0(4)...20 мА, 0...5 мА <input type="checkbox"/> напряжение 0...1 В, 0...10 В <input type="checkbox"/> термопары J, K, L и т.д.	

Подключение датчиков тока и напряжения осуществляется **напрямую** и не требует согласующих резисторов

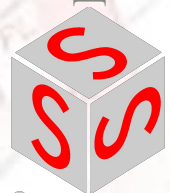
# ЛЕКЦИЯ 2

# CoDeSys

The screenshot displays the CoDeSys software interface with several windows open:

- CoDeSys - example.pro\***: Main application window with menu (Datei, Bearbeiten, Projekt, Einfügen, Extras, Online, Fenster, Hilfe) and toolbar.
- Bausteine**: Project tree showing folders like 'Beispiel Ordner' and sub-items like 'MainTask (PRG)', 'PointerP (PRG)', 'VISU\_FEATURES', 'CFC\_EXAMPLE (PRG)', 'FBD\_EXAMPLE (FUN)', 'IL\_EXAMPLE (FB)', 'LD\_EXAMPLE (PRG)', 'SFC\_EXAMPLE (PRG)', 'SlowTask (PRG)', and 'ST\_EXAMPLE (PRG)'. 'CFC\_EXAMPLE (PRG)' is selected.
- CFC\_EXAMPLE (PRG-CFC)**: Ladder logic window showing a 'PID\_Regler' block with inputs (ACTUAL, SET\_POINT, KP, TN, TV, Y\_OFFSET, Y\_MIN, Y\_MAX, MANUAL, RESET) and outputs (LIMITS\_ACTIVE, OVERFLOW).
- LD\_EXAMPLE (PRG-KOP)**: Ladder logic window showing a network with a timer (T#0ms) and a coil (M = FALSE). Below it, two networks with switch logic (Switch1-Switch5) are visible.
- SFC\_EXAMPLE (PRG-SFC)**: State transition diagram window showing states like 'Init', 'Step8', 'Parallel2', 'Parallel1', and 'Step9' with transitions labeled 'Start\_As', 'Testx', and 'Testy'.
- FBD\_EXAMPLE (FUN-FUP)**: Function block diagram window showing logic with variables (Var1=175, Var2=94) and blocks (MOD, LT, AND, GT).

Курсы по 3S CoDeSys для ОВЕН ПЛК



Smart Software Solutions

We software

# Что такое **CoDeSys**?

- Инструмент программирования
- Инструмент отладки
- Инструмент тестирования
- Инструмент создания визуализаций
- Инструмент документирования проектов

CoDeSys – пакет для создания программного обеспечения для ПЛК в соответствии со стандартом МЭК 61131-3

## Основные принципы стандарта МЭК **61131-3**

- Определяет принципы программирования ПЛК
- Включает хорошо известные и современные языки программирования
- Позволяет разработчику не зависеть от производителя системы программирования
- Повторное использование кода
- Стандарт является международным

# Что определяет стандарт МЭК **61131-3**

- Структуру проекта
- Синтаксис и семантику 5 различных языков программирования: IL, FBD, LD, ST и SFC
- Типы строительных блоков проекта (POU): функции, программы и функциональные блоки
- Правила объявления и типы переменных

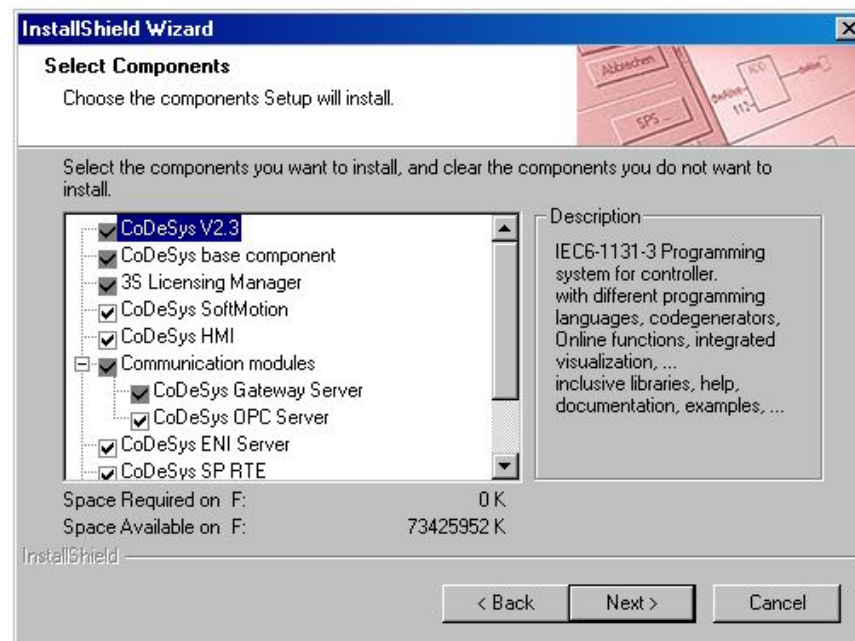
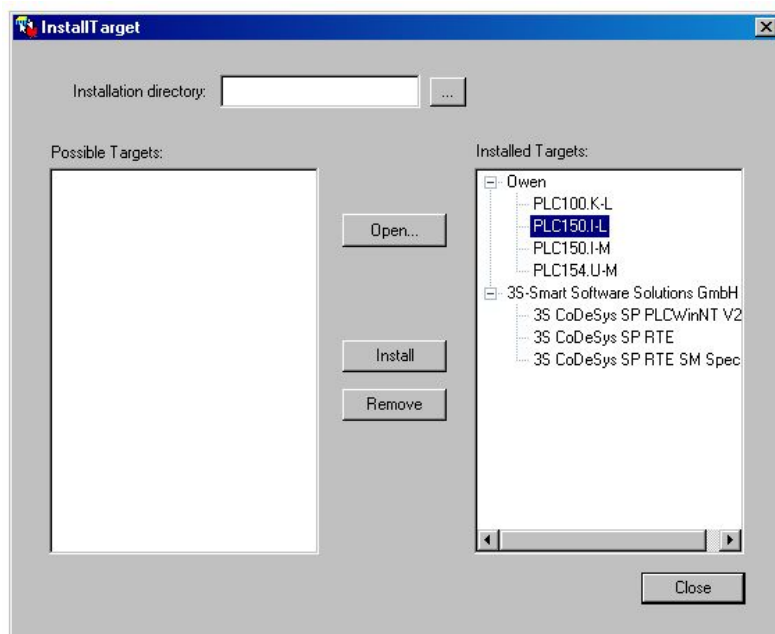


# Введение в **CoDeSys**

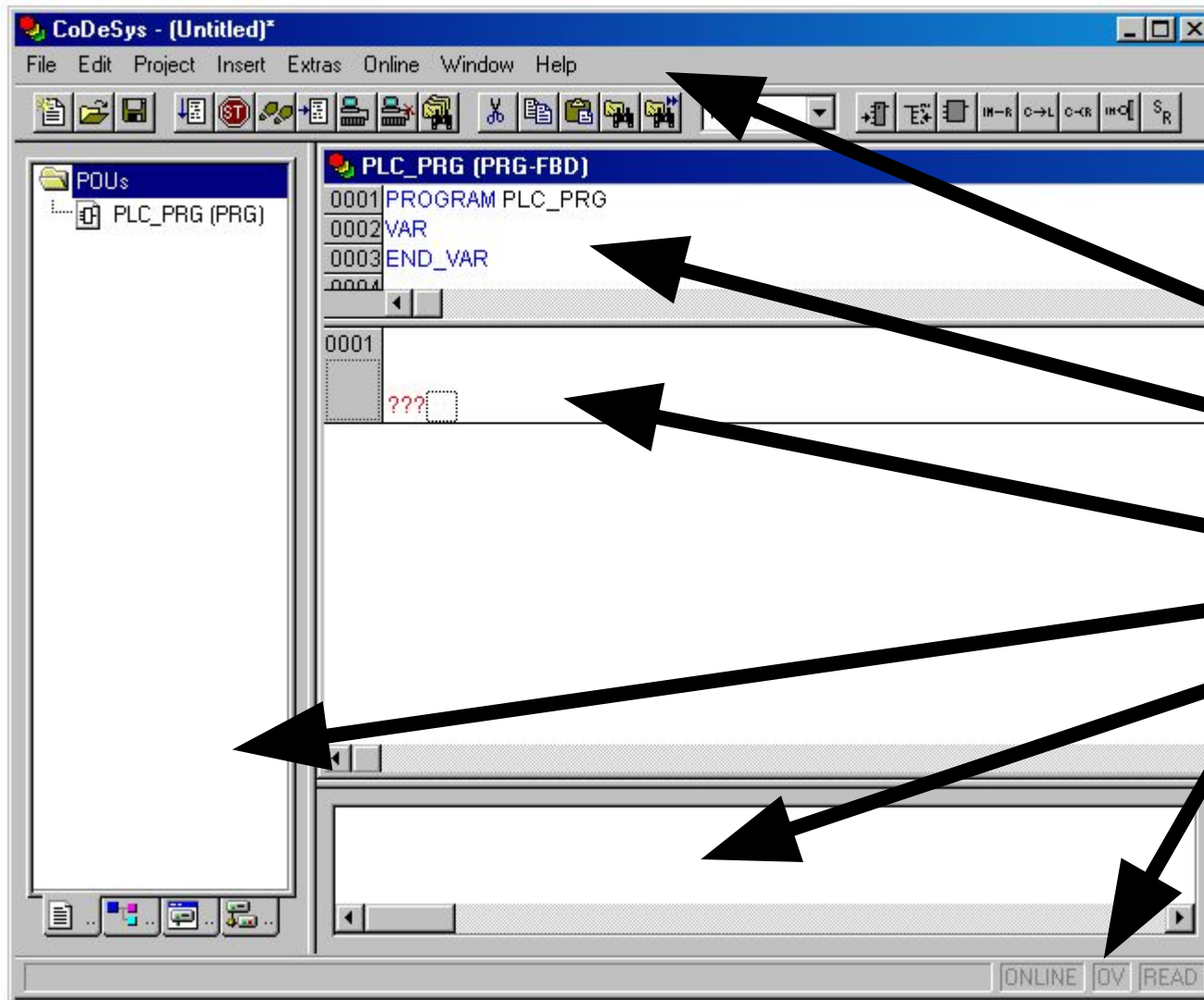
- Состоит из двух частей : системы программирования и системы исполнения.
- Система программирования состоит из:
  - редактора, компилятора и отладчика МЭК проектов;
  - поддерживает все 5 языков программирования МЭК;
  - генерирует машинный код для довольно широкого набора процессоров.
- Система исполнения реализует:
  - управляющий цикл с обновлением входов/выходов;
  - связь с системой программирования;
  - загрузку приложения после включения питания контроллера.

# Инсталляция **CoDeSys**

- Инсталляция CoDeSys с компакт диска или с сайта [www.owen.ru](http://www.owen.ru)
- Инсталляция файлов целевой платформы



# Первый запуск **CoDeSys**



Главное меню и панель инструментов

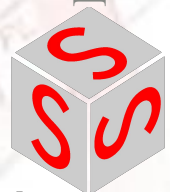
Область определения переменных

Редактор

Менеджер объектов

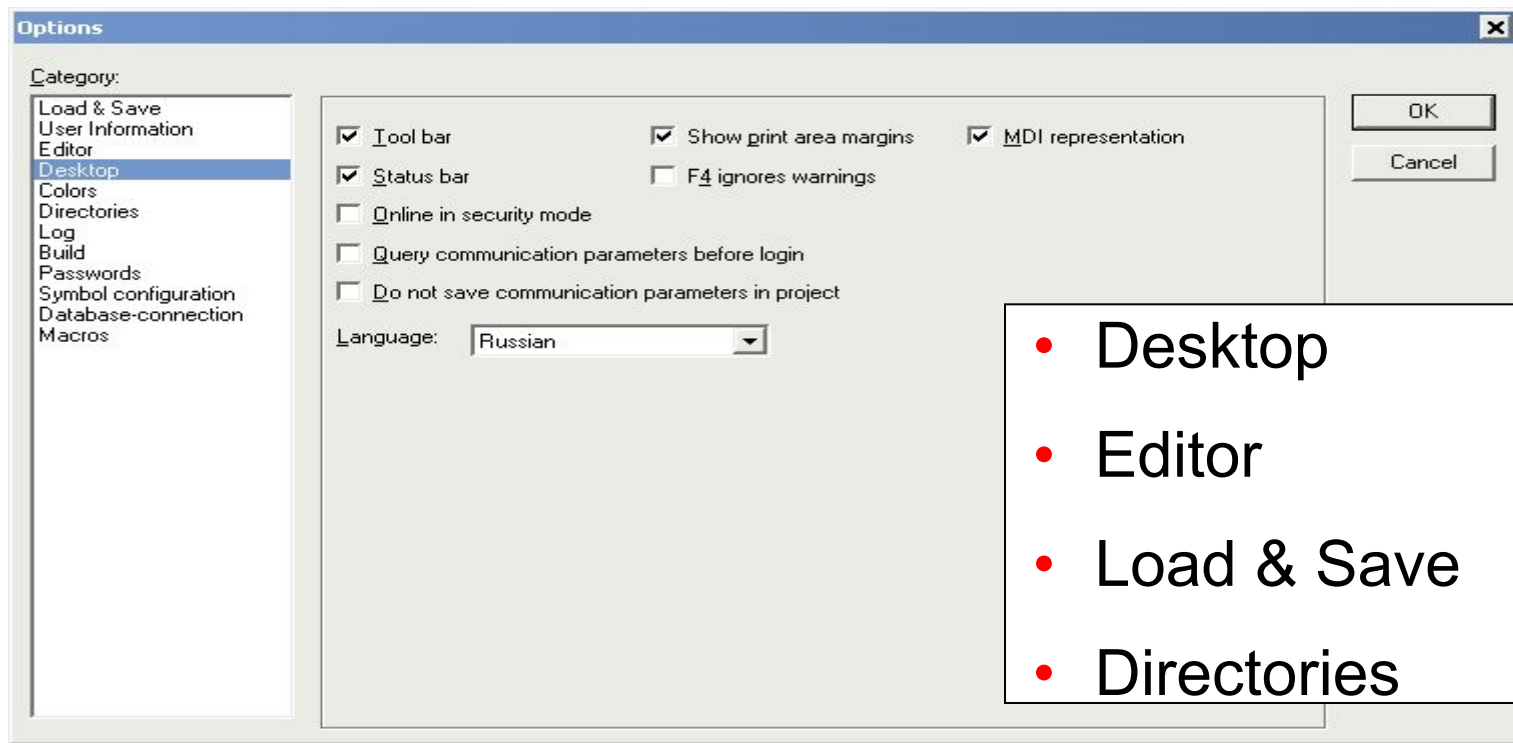
Окно сообщений

Строка статуса





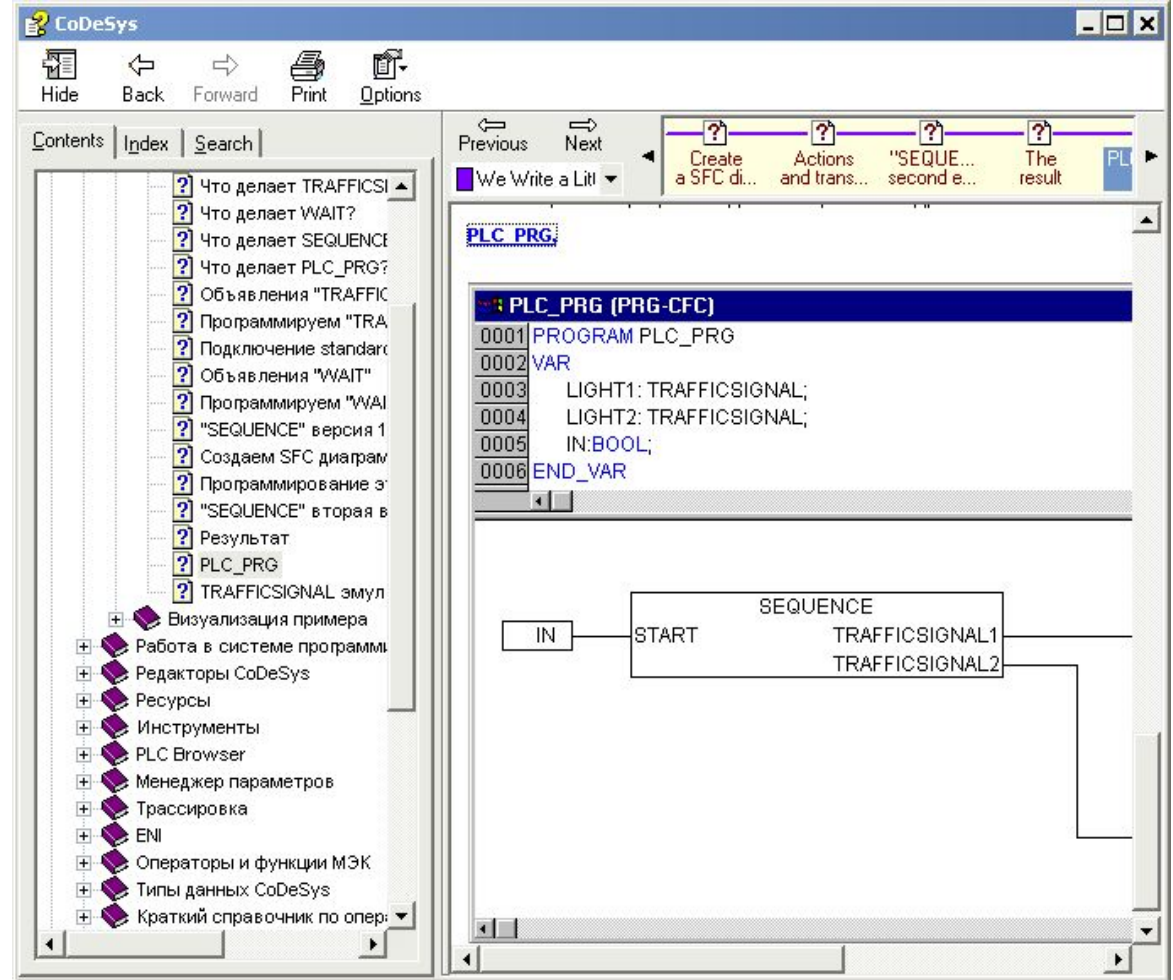
# Наиболее используемые опции **CoDeSys**



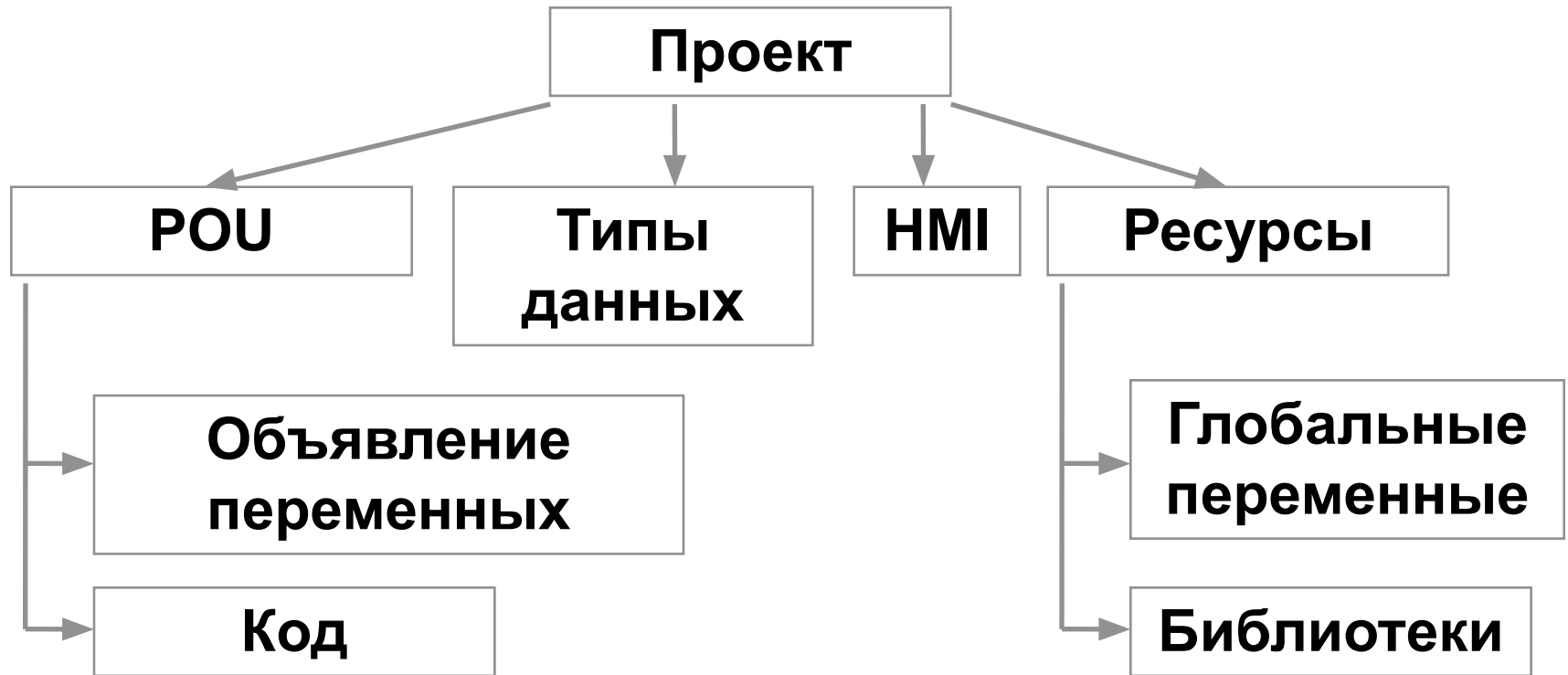
- Desktop
- Editor
- Load & Save
- Directories

# Справочная система

- Содержит ту же информацию, что и документация по CoDeSys
- Индекс по ключевым словам
- Поиск по тексту
- Русифицирована



# Структура проекта



# Структура проекта

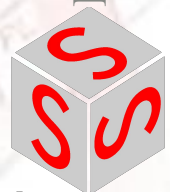
Курсы по 3S CoDeSys для ОВЕН ПЛК

The screenshot displays the CoDeSys software interface for a project named "100kl\_sms.pro - [PLC\_PRG (PRG-ST)]". The interface is divided into several sections:

- Left Panel (Project Explorer):** Shows a tree view of Project Objects (POUs). The items listed are: knopka (FB), mdvy\_cou (PRG), mva (PRG), mvu\_man (PRG), **PLC\_PRG (PRG)** (highlighted), prg1 (PRG), send\_prg (PRG), SMS1\_prg (PRG), and tm1 (FB).
- Code Editor:** Displays the ladder logic code for the selected PLC\_PRG. The code is as follows:

```
0001 PROGRAM PLC_PRG
0002 VAR
0003
0004 END VAR
```
- Annotations:** Four white boxes with black text are overlaid on the code editor, with arrows pointing to the corresponding code lines:
  - POU:** Points to line 0001.
  - Типы данных:** Points to line 0002.
  - HMI:** Points to line 0003.
  - Ресурсы:** Points to line 0004.
- Bottom Panel (Status Bar):** Shows the current line and column (Lin.: 30, Col.: 1) and the status (ONLINE, DV, READ). Below the code editor, there are buttons for "POUs", "Data ...", "Visua...", and "Reso...".
- Bottom Panel (Library Loading):** Shows the loading of standard libraries:

```
Loading library 'F:\Program Files\3S Software\CoDeSys V2.3\Library\STANDARD.LIB'
Loading library 'F:\Program Files\3S Software\CoDeSys V2.3\Library\SYSLIBTIME.LIB'
Loading library 'F:\Program Files\3S Software\CoDeSys V2.3\Library\SYSLIBCALLBACK.LIB'
```



## Что такое проект в **CoDeSys** ?

- ...хранится в одном файле (name.pro)
- ...содержит программные компоненты (POU), визуализации, ресурсы и т.д.
- ... выполнение приложения начинается с POU PLC\_PRG(аналог функции main )
- ... выполняется циклически

# Что такое **POU** ?

POU (Program organisation unit) –это программный модуль

POU **PLC\_PRG** вызывается неявно системой исполнения

Стандарт МЭК 61131-3 определяет 3 типа POU

- Программы <PROGRAM>
- Функциональные блоки <FUNCTION\_BLOCK>
- Функции <FUNCTION>

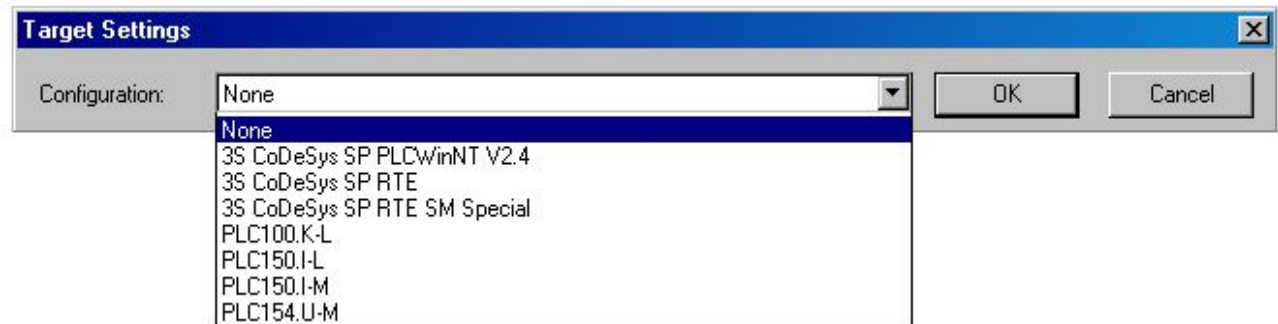
## Главная программа PLC\_PRG:

Для однозадачных систем программа **PLC\_PRG** соответствует **OB1** в системах **S5/7**.

Эта программа вызывается циклически системой исполнения



# Первый проект (Инкремент переменной)



- <File / New>
- Target Settings
- Создание главной программы PLC\_PRG
- Автоматическое объявление
- <Online / Simulation>
- <Online / Login>
- <Online / Start>



# Стандартные типы данных

- В МЭК 61131-3 определены следующие типы данных:

Ключевое слово	Диапазон	Пример
BOOL	0, 1	FALSE, TRUE, 0, 1
SINT, INT, DINT	-128 .. 127, -32768 .. 32767, -2147483648 .. 2147483647	0, 24453 -38099887
USINT, UINT, UDINT	0 .. 255, 0 .. 65535, 0 .. 4294967295	200, 47453 138099887
BYTE, WORD, DWORD	0 .. 255, 0 .. 65535, 0 .. 4294967295	8450 16#2102
REAL, LREAL	$-1.2 \times 10^{-38} \dots 3.4 \times 10^{38}$ $-2.3 \times 10^{-308} \dots 1.7 \times 10^{308}$	1.34996 2.8377E-15
TIME, TOD, DATE, DT	0 ms .. 1193h2m47s295ms 00:00:00 .. 23:59:59 01.01.1970 до. 06.02.2106	T#1d8h12m8s125ms TOD#12:34:17 D#2001-03-15 DT#2001-03-15-12:17:03
STRING	1 .. 255 символов	`Emergency Stop`

# Представление данных в **CoDeSys**

## 3 метода объявления переменных

текстовый, табличный и автоматический





Локальные *(для 1 ФБ)* или Глобальные *(для всех ФБ)*

Сохраняемые и постоянные переменные

# Синтаксис идентификаторов

- Буквы и цифры
  - Должен начинаться с буквы
  - Только одинарные подчеркивания
  - Без пробелов
  - Нельзя использовать зарезервированные слова МЭК и операторы
  - Регистр не различается
- Примеры
    - Otto, otto, ОТТО
    - Valve1
    - a\_long\_name

# Основные команды режима **Online**

- <Online / Simulation Mode >
-  <Online / Login [Alt+F8] / Logout [Ctrl+F8]>
-  <Online / Start [F5]>
-  <Online / Stop [Shift+F8]>
- <Online / Single Cycle>
-  <Online / Breakpoint [F9]>
- <Online / Write Values [Ctrl+F7]>
- <Online / Force Values [F7]>
- <Online / Release Force [Shift+F7]>

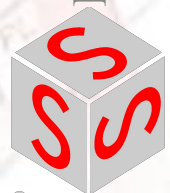
# Запуск приложения в целевой платформе

(в ОВЕН ПЛК)

- Запустить систему исполнения
- Выключить режим эмуляции <Online / Simulation Mode>
- Настроить параметры связи <Online / Communication Parameter...>

# Языки МЭК **61131-3**

- Список инструкций (IL)
- Структурированный текст (ST)
- Язык функциональных блокковых диаграмм (FBD)
- Язык релейных диаграмм (LD)
- Язык последовательных функциональных схем (SFC)



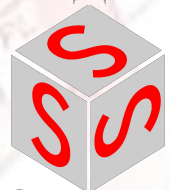
## ЛЕКЦИЯ 3

### Список инструкций (IL)

- Текстовый язык
- Схож с ассемблером
- Все операции производятся через аккумулятор
- Легко читается в случае небольших программ
- Не поддерживает структурного программирования

# Структурный текст (**ST**)

- Текстовый язык
- Язык высокого уровня
- Схож с Паскалем
- Лучший язык для программирования циклов и условий (IF, WHILE, FOR, CASE)





# Язык релейных диаграмм(**LD**)

- Графический язык
- Программа состоит из схем
- Использовался для программирования практически всех классических ПЛК
- Удобен для программирования логических выражений
- Сложно использовать для работы с аналоговыми типами данных
- Переключение между FBD и LD

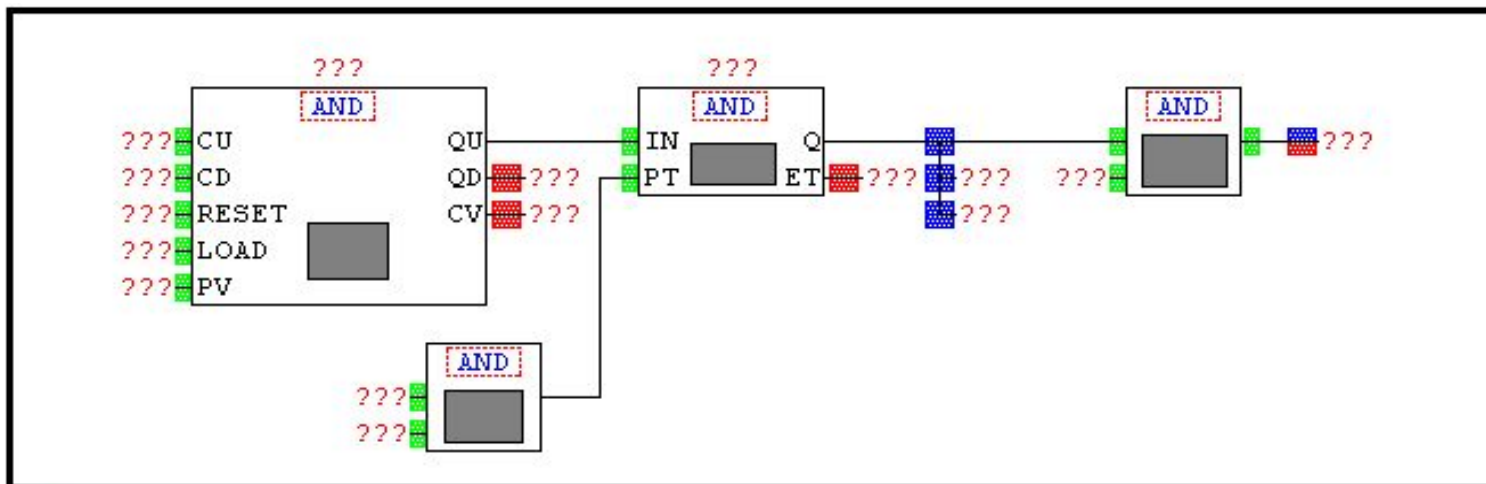
# Язык функциональных блокковых диаграмм (FBD)

- Графический язык
- Программа состоит из нескольких схем
- Легко читается
- Каждая схема состоит из блоков и операндов

# Непрерывные функциональные схемы (CFC)

- Схож с FBD, но...
- Блоки и соединители располагаются свободно
- Разрешаются циклы и свободные соединения

# Язык функциональных блоковых диаграмм (**FBD**)



		[Выход] [Блок] [Присваивание] [Переход] [Возврат] [Инверсия]
		[Добавление входа]
		[Установка/Сброс]
		[Выход]
		[Выход] [Установка/Сброс]
???		< Имя переменной / Имя экземпляра >
		<Имя оператора/функции/функционального блока/программы>

## ЛЕКЦИЯ 4

# Язык последовательных функциональных схем(**SFC**)

- Графический язык
- Используется для структурирования приложений
- Состоит из шагов и переходов
- Действия выполняются внутри шагов
- Не конвертируется в другие языки
- CoDeSys поддерживает два типа SFC
- Подробнее будет рассмотрен завтра !

## Упражнение **2**. Управление освещением в длинном коридоре

- Есть длинный коридор. Для управления освещением в коридоре используется три переключателя:
- Msw- главный переключатель
- Vsw – переключатель в начале коридора.
- Esw – переключатель в конце коридора.

## Упражнение **2**. Управление освещением в длинном коридоре

Подача питания в коридор осуществляется с помощью переключателя Msw.

Необходимо решить задачу включения/выключения света с помощью любого из двух переключателей Bsw и Esw, установленных в разных концах коридора.

Т.е. при входе в коридор с одной стороны необходимо переключить Bsw, чтобы зажечь свет. На выходе с другой стороны коридора необходимо переключить Esw, чтобы свет погас. И наоборот.

# Конфигурирование входов/выходов

- Через ресурс PLC-configuration

- Прямая адресация

Например: `%QX0.7 := (%IX0.3 AND %IX3.7) OR %IX3.0;`

- Присвоение адресам имен

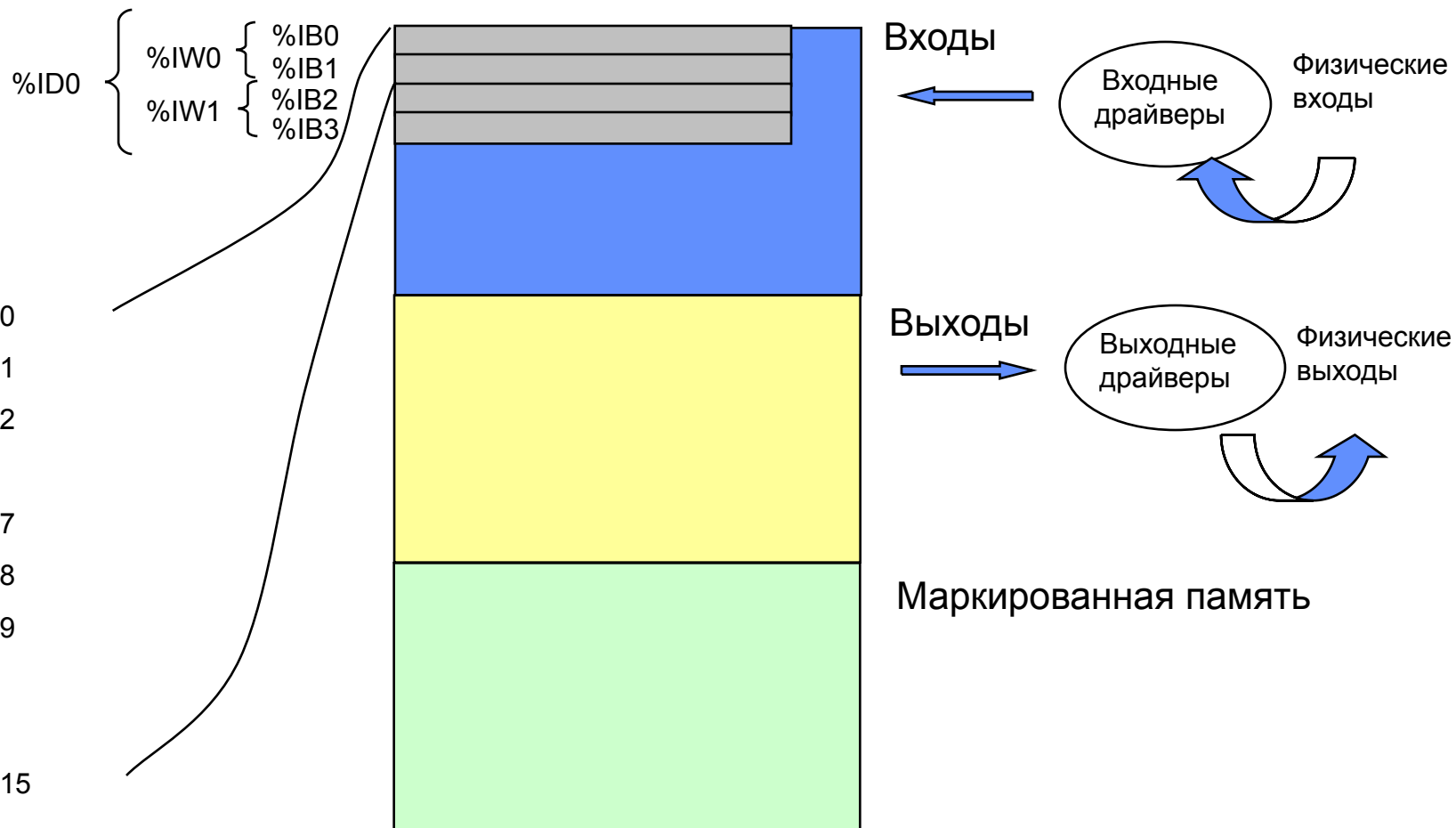
Например: `xInput AT %IX0.7 : BOOL;`

# Синтаксис адресов

- Обозначаются знаком ‘%’
- Тип адреса определяется префиксом
  - I вход
  - Q выход
  - M маркер
- Тип данных
  - Хбит
  - Noneбит
  - Вбайт (8 бит)
  - W слово (16 бит)
  - D двойное слово (32 бит)
- Примеры
  - %IW215
  - %QX1.1
  - %MD48



# Области памяти





# Операторы в **CoDeSys**

CoDeSys поддерживает все операторы  
МЭК 61131-3

- Оператор присваивания
- Битовые операторы
- Сдвиговые операторы
- Операторы сравнения
- Числовые операторы
- Работа с действительными числами
- Логарифмические операторы
- Тригонометрические операторы
- Операторы выбора


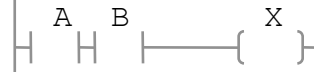

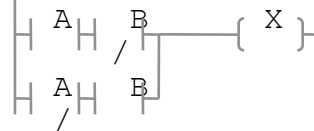
# Операторы присваивания

- Используются для работы со всеми типами данных

Оператор	IL	FBD	LD	ST
<b>LD / ST</b>	LD ST A X	A ————— X		X := A;
<b>LDN / ST</b>	LDN ST A X	A-○ ————— X		X := NOT (A) ;
<b>LD / S</b>	LD S A X	A ————— [S] -X		IF A THEN X := TRUE; END_IF
<b>LD / R</b>	LD R A X	A ————— [R] -X		IF A THEN X := FALSE; END_IF


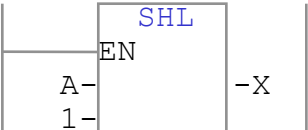

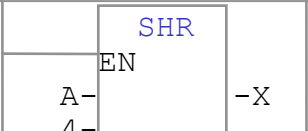

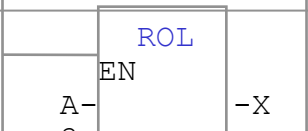

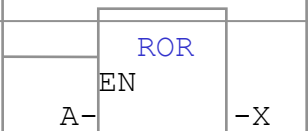
# Битовые операторы

- Используются для работы с двоичными типами данных (BOOL, BYTE, WORD, DWORD)

Оператор	IL	FBD	LD	ST
<b>NOT</b>	LD A STN X	A- NOT -X		X := NOT (A) ;
<b>AND</b>	LD A AND B ST X	A- AND B- -X		X := A AND B ;
<b>OR</b>	LD A OR B ST X	A- OR B- -X		X := A OR B ;
<b>XOR</b>	LD A XOR B ST X	A- XOR B- -X		X := A XOR B ;

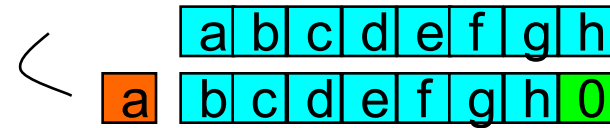
# Сдвиговые операторы (1)

- Используются для работы с двоичными типами данных (BOOL, BYTE, WORD, DWORD)

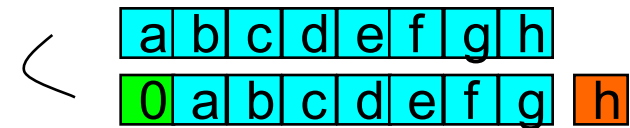
Оператор	IL	FBD	LD	ST
<b>SHL</b>	LD A SHL 1 ST X			X := SHL(A, 1);
<b>SHR</b>	LD A SHR 4 ST X			X := SHR(A, 4);
<b>ROL</b>	LD A ROL 4 ST X			X := ROL(A, 3);
<b>ROR</b>	LD A ROR 1 ST X			X := ROR(A, 1);

## Сдвиговые операторы (2)

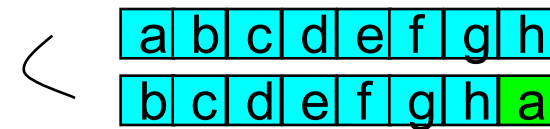
- SHL (сдвиг влево)



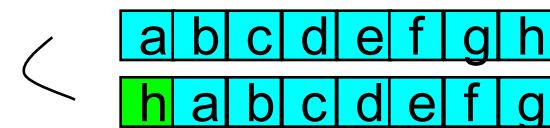
- SHR (сдвиг вправо)



- ROL (цикл. сдвиг влево)



- ROR (цикл. сдвиг вправо)



# Операторы сравнения

- Используются для работы со всеми типами данных

Оператор	IL	FBD	LD	ST	
<b>EQ</b>	LD EQ ST	A B X			X := (A = B);
<b>NE</b>	LD NE ST	A B X		(аналогично)	X := (A <> B);
<b>GE</b>	LD GE ST	A B X		(аналогично)	X := (A >= B);
<b>GT</b>	LD GT ST	A B X		(аналогично)	X := (A > B);
<b>LE</b>	LD LE ST	A B X		(аналогично)	X := (A <= B);
<b>LT</b>	LD LT ST	A B X		(аналогично)	X := (A < B);

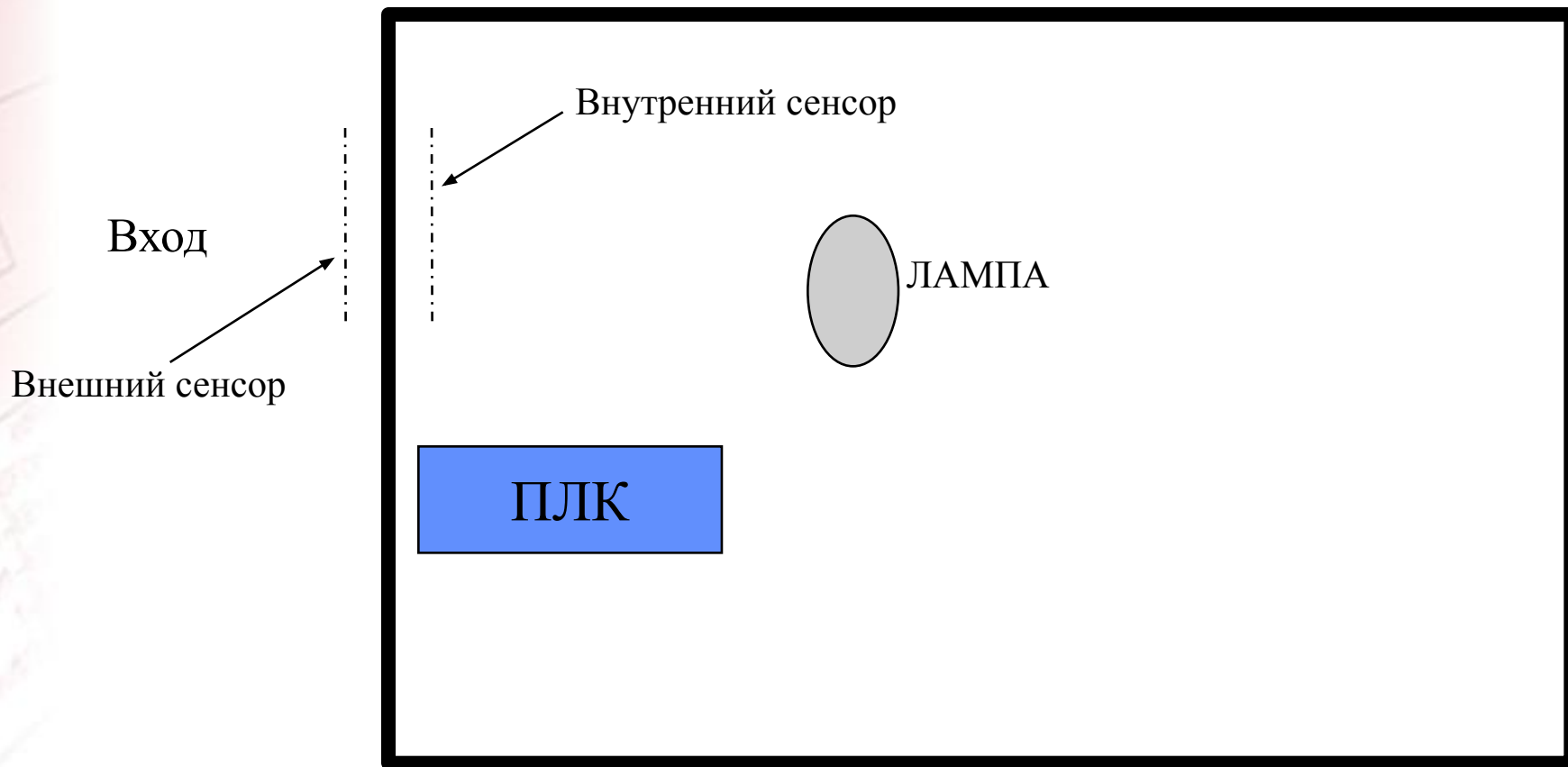
# Арифметические операторы

- Выполняют алгебраические операции над целыми числами и числами с плавающей запятой

Оператор	IL	FBP	LD	ST
<b>ADD</b>	LD A ADD 1 ST X	ADD A- -X 1-	ADD EN A- -X 1-	X := A + 1;
<b>SUB</b>	LD A SUB 4 ST X	SUB A- -X 4-	SUB EN A- -X 3-	X := A - 4;
<b>MUL</b>	LD A MUL B ST X	MUL A- -X B-	MUL EN A- -X B-	X := A * B;
<b>DIV</b>	LD A DIV 8 ST X	DIV A- -X 8-	(аналогично)	X := A / 8;
<b>MOD</b>	LD 12 MOD 8 ST X	MOD 12- -X 8-	(аналогично)	X := 12 MOD 8; ( Result = 4 ) ( не исп. для REAL )



## Упражнение 3. Управление освещением в комнате



Цель - свет должен быть выключен, когда в комнате никого нет!

## Упражнение 3. Управление освещением в комнате

На входе установлены два дискретных датчика: один снаружи комнаты, другой внутри.

Когда срабатывает сначала внешний датчик, затем внутренний, это означает, что человек зашел в комнату.

Когда срабатывает сначала внутренний датчик, затем внешний, это означает, что человек вышел из комнаты.

**Задача1:** Если человек вошел – включить свет, Если человек вышел – выключить свет.

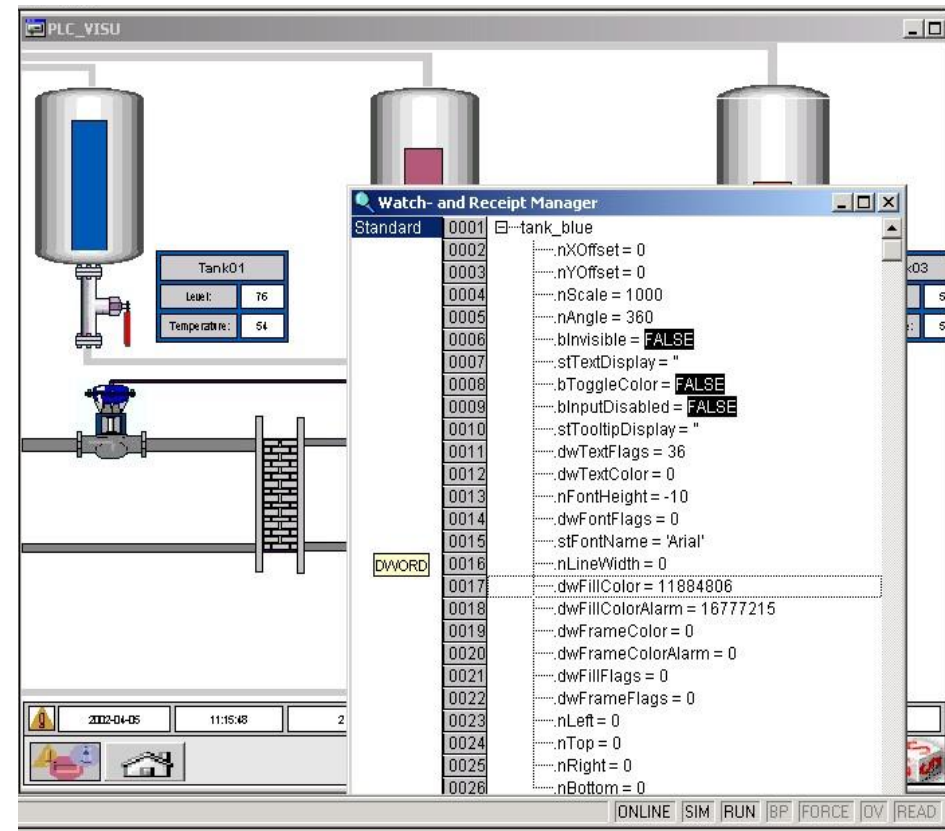
**Задача2:** Необходимо считать количество людей, заходящих и выходящих из комнаты.

Пока в комнате остается хотя бы один человек, свет должен быть включен.

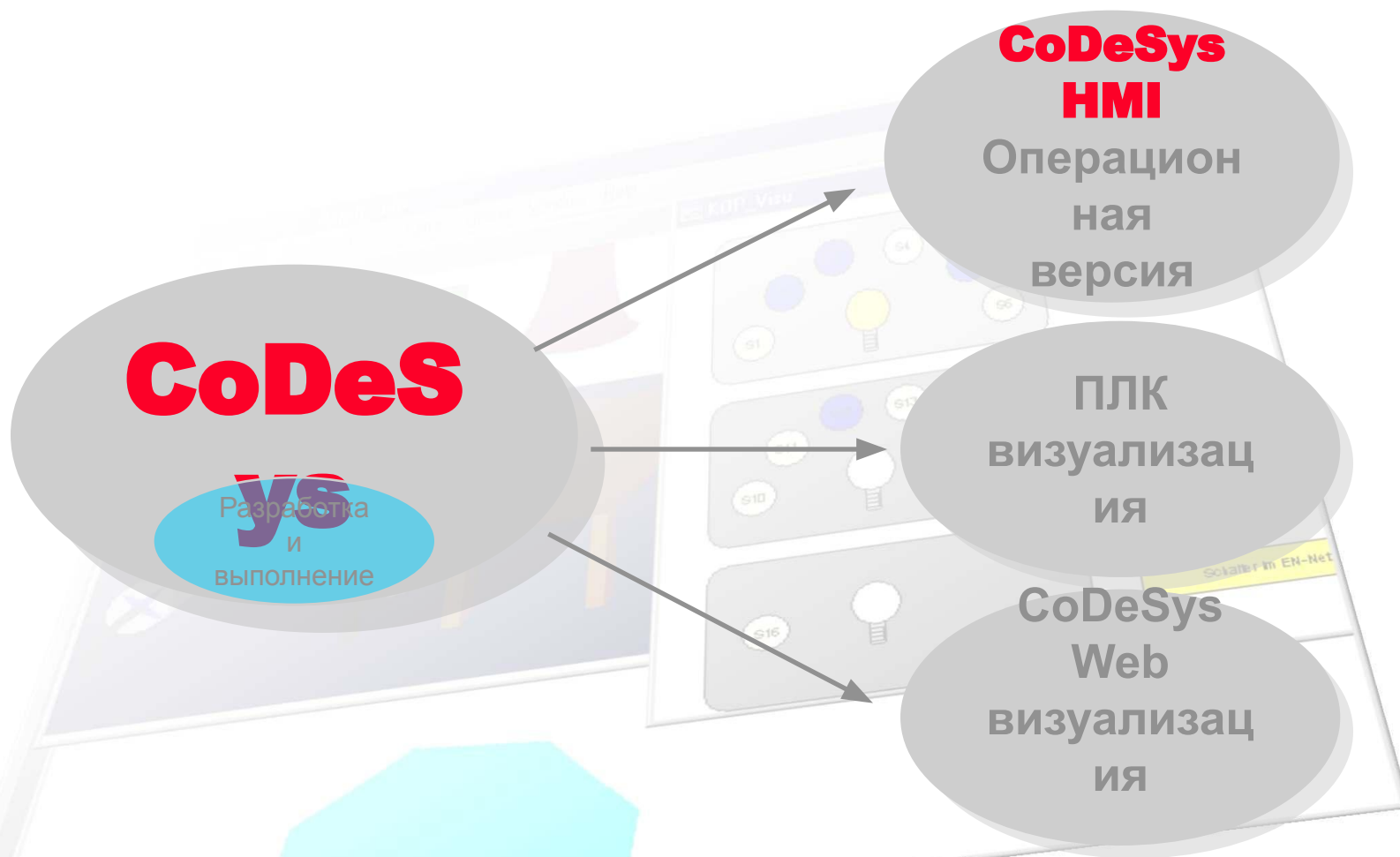
# ЛЕКЦИЯ 5

## Визуализация

- Доступ ко всем данным проекта
- Графическое отображение логических и численных значений
- Ввод логических и численных значений
- Перемещение графических объектов



# Инструменты визуализации



# Типы **POU**

- Функция: **< FUNCTION >**  
Имеет один или более входов, один выход, рекурсии не допустимы
- Функциональный блок: **<FUNCTION\_BLOCK >**  
Имеет произвольное число входов и выходов. Имеет внутреннюю память. Для каждого функционального блока можно объявить несколько экземпляров
- Программа: **< PROGRAM >**  
Подобна функциональному блоку, но имеет один глобальный экземпляр

# Функция

- Не имеет внутренней памяти
- Локальные переменные инициализируются при каждом вызове
- Функция возвращает значение, через свой идентификатор. Функция имеет тип!
- Удобна для реализации комплексных вычислений
- Не рекомендуется использование глобальных переменных в функции

# Функциональный блок

- Все переменные функционального блока сохраняют значения
- При создании экземпляра функционального блока создается новая копия переменных функционального блока. Копия кода функционального блока не создается.
- Рекомендуется для программирования повторно используемого кода, например, счетчиков, таймеров, триггеров и т.д.

# Программа

- Все переменные сохраняют свои значения
- Используется для структурирования приложения





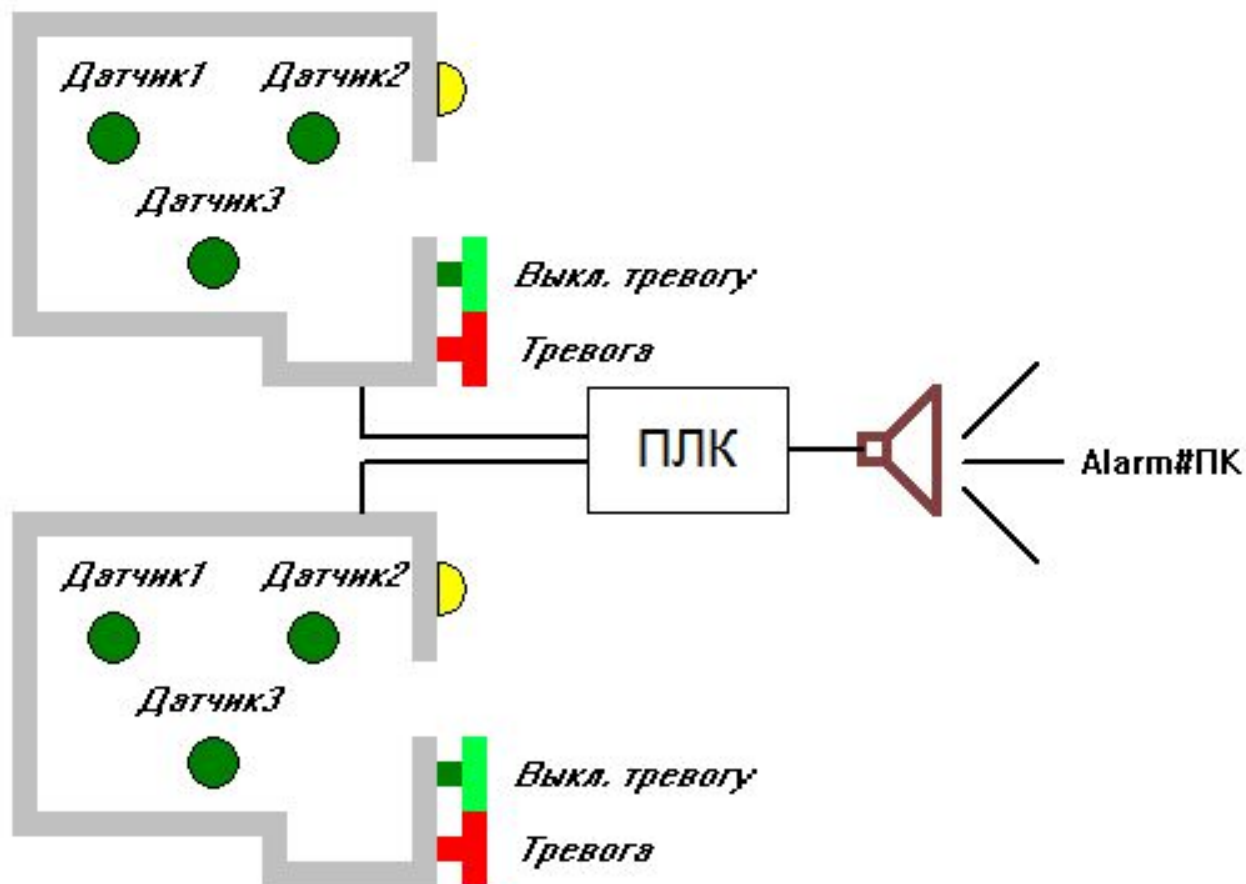
# Вызов POU

	Функция	ФБ	Программа
<b>Пример</b>	Function Fun1:INT 3 входа (INT): A, B, C  Переменная Result (int)	Function_Block FunBlck1 3 входа (INT): A, B, C 2 выхода (INT): D, E Экземпляр: Instance1	Program Prgr1 3 входа (INT): A, B, C 2 выхода (INT): D, E
<b>IL</b>	LD 5 Fun1 3,2 ST Result	CAL Instance1(A:=5, B:=3, C:=2) ... LD Instance1.D ST Result1 LD Instance1.E ST Result2	CAL Prgr1(a := 5, b := 3, c := 2) ... LD Prgr1.D ST Result1 LD Prgr1.E ST Result2
<b>ST</b>	Result:=Fun1(5,3,2);  или Result:=Fun1(A:=5,B:=3,C:=2);	Instance1(A:=5, B:=3, C:=2, D => Result1, E => Result2); или Result1:=Instance1.D; ...	Prgr1(A := 5, B := 3, C := 2, D => Result1, E => Result2); или Result1:= Prgr1.D; ...
<b>LD FBD CFC</b>			

## Упражнение **4.** Работа с программными компонентами **CoDeSys (POU)**

- Функция расчета мощности постоянного тока по напряжению и сопротивлению
- Счетчик положительных фронтов дискретного сигнала
- Вызов функций и функциональных блоков из программы

## Упражнение 5. Система пожарной сигнализации здания



## Упражнение **5.** Система пожарной сигнализации здания

В здании две одинаковые комнаты.

В каждой комнате установлено три пожарных датчика, кнопка ручного включения сигнализации и кнопка ручного отключения сигнализации. Для каждой комнаты предусмотрена сигнальная лампа. Сигнализация пожара является общей для обеих комнат.

Если в комнате срабатывает хотя бы один из датчиков, то загорается сигнальная лампа для соответствующей комнаты. Лампа гаснет, если все датчики в комнате отключены.

Если в комнате срабатывает любые два из трех датчиков, то включается пожарная сигнализация. Сигнализация работает до тех пор, пока ее не отключат соответствующей кнопкой.

Сигнализация может быть включена кнопкой включения вне зависимости от состояния датчиков.

# Сложные типы данных

- Массив

```
abList : ARRAY[0..31] OF BOOL;
```

- Структура

```
TYPE SetType :  
  STRUCT  
    iCount   : INT;  
    rValue   : ARRAY[0..9] OF REAL;  
  END_STRUCT  
END_TYPE
```

- Перечисление

```
TYPE ColorType :  
  ( RED, YELLOW, GREEN, BLUE );  
END_TYPE
```

- Псевдоним

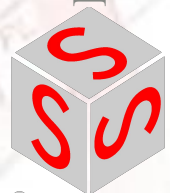
```
TYPE Message : STRING(40); END_TYPE
```

# Предопределенные блоки (Библиотеки)

- Библиотека состоит из объектов, которые могут быть использованы в различных проектах
- Пользователь может создавать и использовать собственные библиотеки.
- Можно создавать библиотеки с защитой.
- Библиотеки могут быть написаны не только на МЭК, но и на других языках программирования
- Библиотека `standard.lib` содержит POU описанные в стандарте МЭК

# Стандартная библиотека

- **Функции работы со строками**
- **Детекторы фронтов**
- **Счетчики**
- **Таймеры**



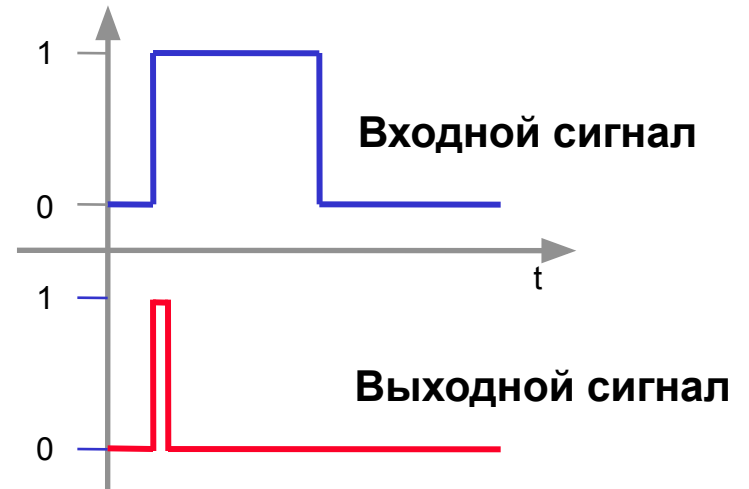
# Функции работы со строками

- **LEN**
- **LEFT**
- **RIGHT**
- **MID**
- **CONCAT**
- **INSERT**
- **DELETE**
- **REPLACE**
- **FIND**

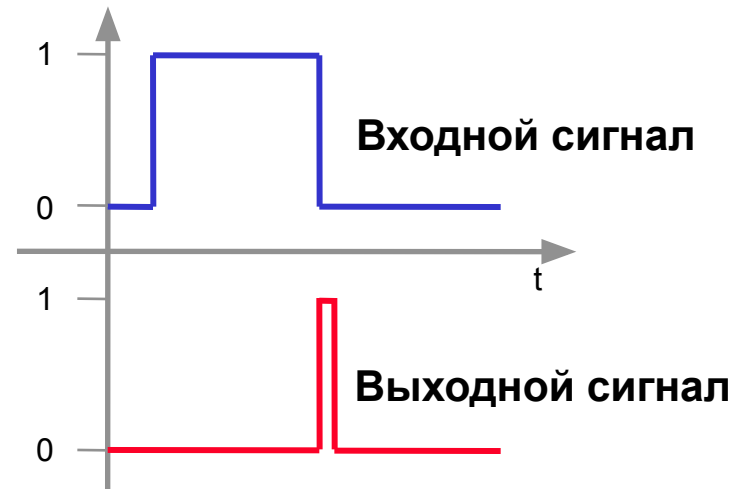


# Детекторы фронтов

- **R\_TRIG**  
определяет  
передний фронт



- **F\_TRIG**  
определяет задний фронт



# Счетчики

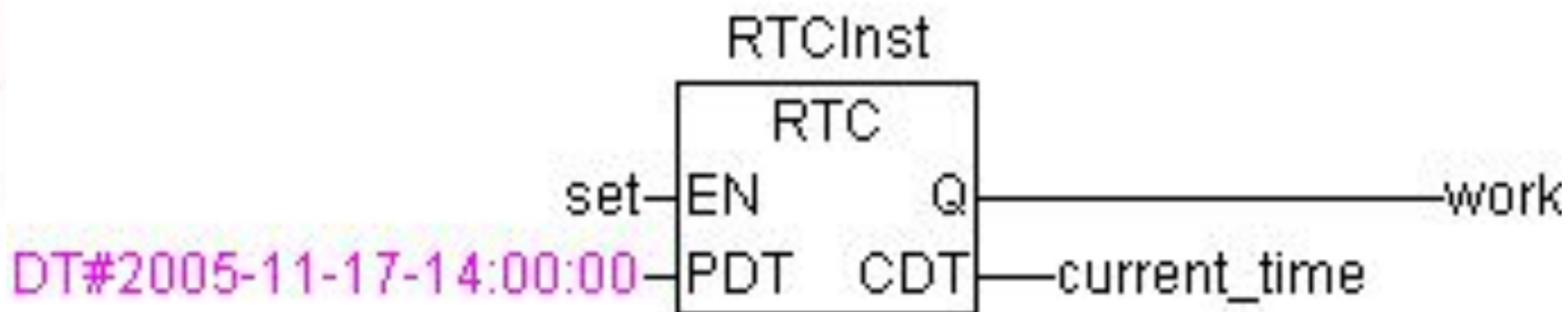
- **STU**  
Инкрементируется по переднему фронту
- **STD**  
Декрементируется по переднему фронту
- **STUD**  
Инкрементируется или  
декрементируется по разным входам



# Временные типы данных МЭК 61131-3

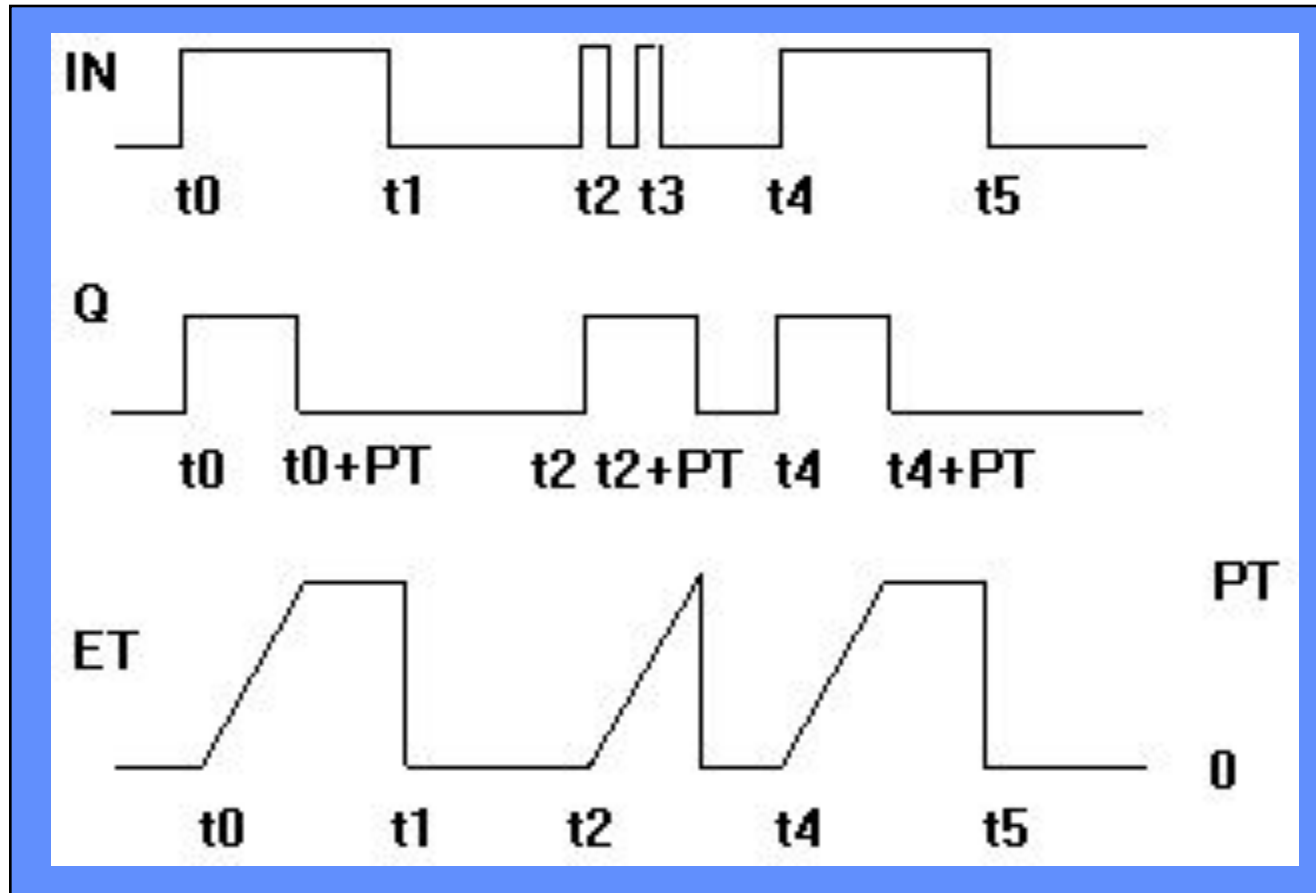
Тип	Описание	Пример
<b>TIME</b>	Используются для выражения интервалов времени	T1:=T#5h45m10s9ms T2:=T#100ms
<b>DATE</b>	Используются для выражения даты	DATE#1996-05-06 d#1972-03-29
<b>TIME_OF_DAY</b>	Используются для выражения времени дня	TIME_OF_DAY#15:36:30.123 tod#00:00:00
<b>DATE_AND_TIME</b>	Используются для выражения даты и времени дня	DATE_AND_TIME#1996-05-06-15:36:30 dt#1972-03-29-00:00:00

# Часы реального времени RTC



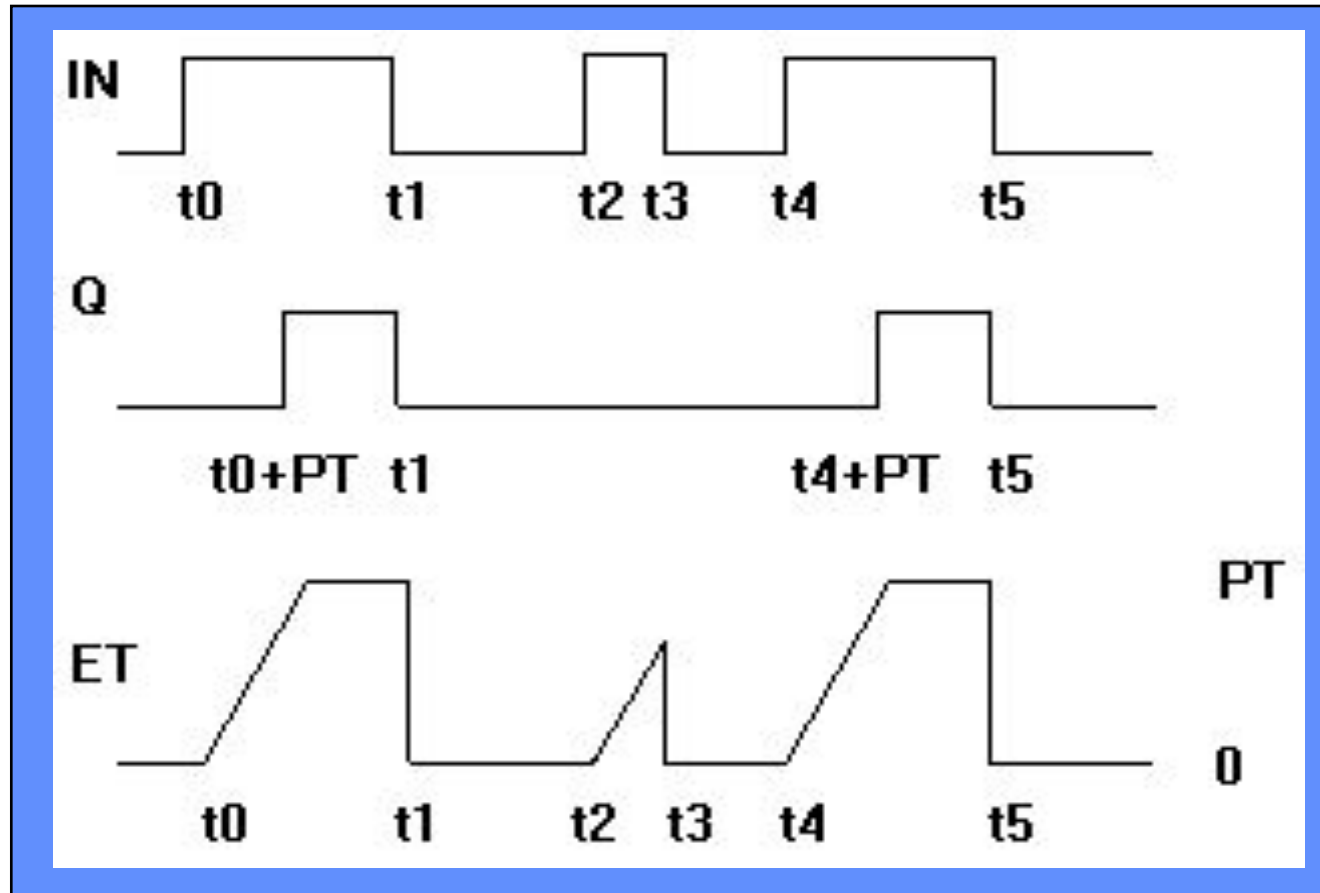
# Таймер **TP**

Генерирует импульс заданной длительности



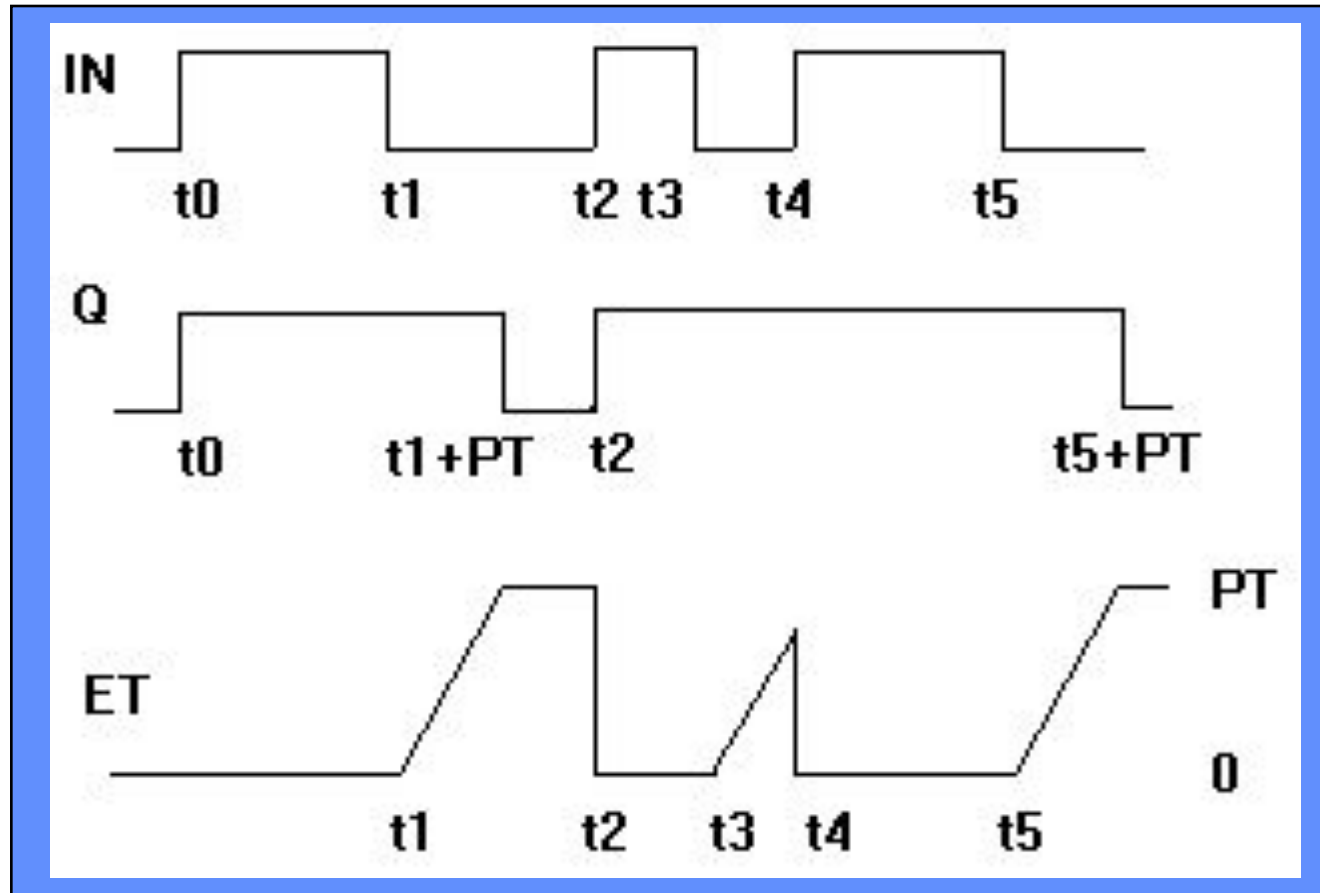
# Таймер **TON**

Включает выход с задержкой по переднему фронту



# Таймер **TOF**

Выключает выход с задержкой по заднему фронту



## Упражнение 6. Работа с элементами стандартной библиотеки

- Реализовать задачу управления светом комнате (упражнение 3) с помощью компонентов стандартной библиотеки. Свет должен выключаться через 5 секунд, после того как последний человек покинет комнату.



# Операторы для работы с числами с плавающей запятой

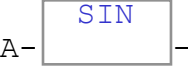
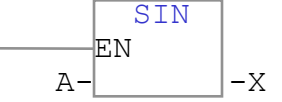

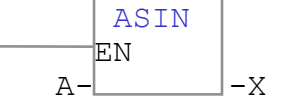

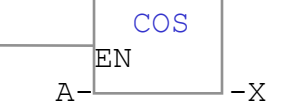

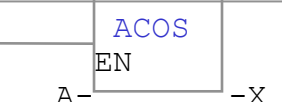

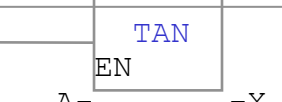

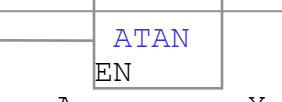
Оператор	IL	FBD	LD	ST
<b>ABS</b>	LD A ABS ST X	A- ABS -X	A- ABS EN -X	X := ABS(A); ( Result = 12 ) ( if A = -12.0 )
<b>TRUNC</b>	LD A TRUNC ST X	A- TRUNC -X	A- TRUNC EN -X	X := TRUNC(A); ( Result = 4 ) ( if A = 4.32 )
<b>EXPT</b>	LD A EXPT 3 ST X	A- EXPT -X 3-	A- EXPT EN -X 3-	X := EXPT(A, 3); ( Result = 8 ) ( if A = 2 )
<b>SQRT</b>	LD A SQRT ST X	A- SQRT -X	A- SQRT EN -X	X := SQRT(A); ( Result = 5 ) ( if A = 25 )

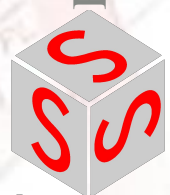
# Логарифмические операторы

- Вычисление логарифмов и экспоненты

Оператор	IL	FBD	LD	ST
<b>EXP</b>	LD EXP ST	A- <div style="border: 1px solid black; padding: 2px; display: inline-block;">EXP</div> -X	<div style="border: 1px solid black; padding: 2px; display: inline-block;">EXP</div> A-EN -X	X := EXP(A); ( Result = 7.389 ) ( if A = 2 )
<b>LN</b>	LD LN ST	A- <div style="border: 1px solid black; padding: 2px; display: inline-block;">LN</div> -X	<div style="border: 1px solid black; padding: 2px; display: inline-block;">LN</div> A-EN -X	X := LN(A); ( Result = 2 ) ( if A = 7.389 )
<b>LOG</b>	LD LOG ST	A- <div style="border: 1px solid black; padding: 2px; display: inline-block;">LOG</div> -X	<div style="border: 1px solid black; padding: 2px; display: inline-block;">LOG</div> A-EN -X	X := LOG(A); ( Result = 3 ) ( if A = 1000 )


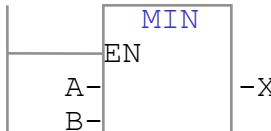
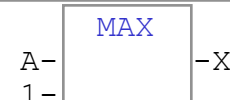
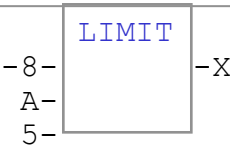
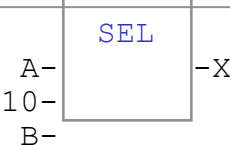
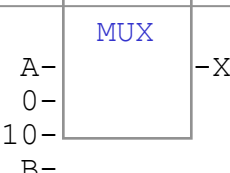
# Тригонометрические операторы

Оператор	IL	FBD	LD	ST
<b>SIN</b>	LD A SIN ST X	A-  -X	 -X	X := SIN (A) ;
<b>ASIN</b>	LD A ASIN ST X	A-  -X	 -X	X := ASIN (A) ;
<b>COS</b>	LD A COS ST X	A-  -X	 -X	X := COS (A) ;
<b>ACOS</b>	LD A ACOS ST X	A-  -X	 -X	X := ACOS (A) ;
<b>TAN</b>	LD A TAN ST X	A-  -X	 -X	X := TAN (A) ;
<b>ATAN</b>	LD A ATAN ST X	A-  -X	 -X	X := ATAN (A) ;



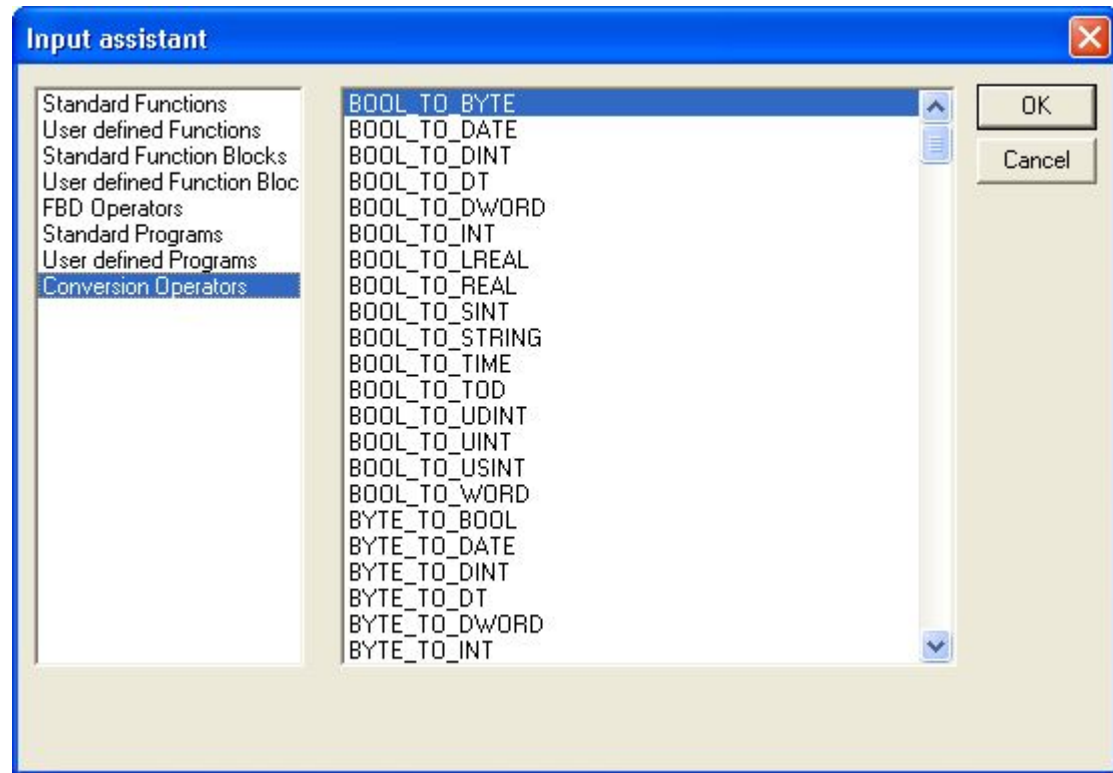
# Операторы выбора

- Предназначены для ограничения и выбора операндов
- Используются с любыми типами данных

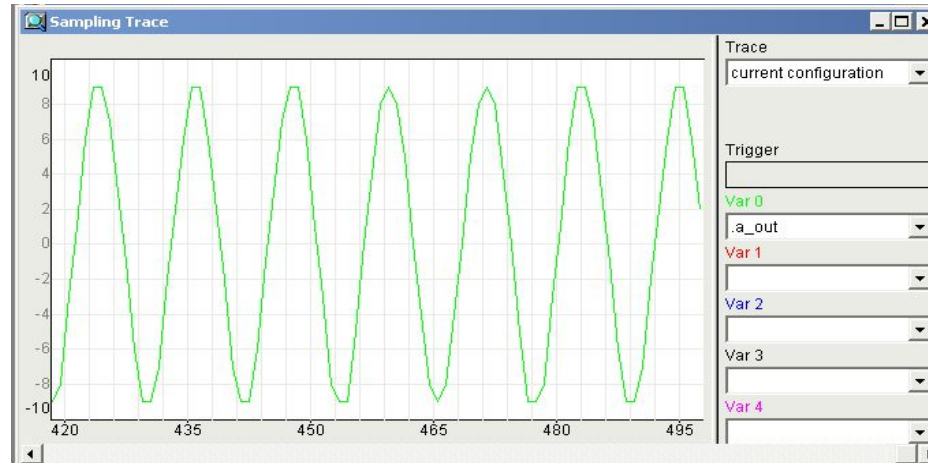
Оператор	IL	FBD	LD	ST
<b>MIN</b>	LD     A MIN    B ST     X	A-  -X B-	 -X	X := MIN(A,B);
<b>MAX</b>	LD     A MAX    1 ST     X	A-  -X 1-	( как выше )	X := MAX(A,1);
<b>LIMIT</b>	LD     -8 LIMIT  A,5 ST     X	-8-  -X A- 5-	( как выше )	X := LIMIT(-8,A,5);  X = -8 if A < -8 X = 5 if A > 5
<b>SEL</b>	LD     A SEL    10,B ST     X	A-  -X 10- B-	( как выше )	X := SEL(A,10,B);  X = 10 if A is FALSE X = B if A is TRUE
<b>MUX</b>	LD     A MUX    0,10,B ST     X	A-  -X 0- 10- B-	( как выше )	X := MUX(A,0,10,B);  X = 0 if A is 0 X = 10 if A is 1 X = B if A is 2

# Операторы преобразования типов данных

- Для каждой пары типов данных используется отдельная функция



## Упражнение 7. Генератор синусоиды



- Операции с вещественными числами
- Преобразование типов
- Первое знакомство с трассировкой

# Язык Последовательных Функциональных диаграмм (SFC)

- Графический язык
- Управление последовательностью выполнения действия
- Состоит из шагов, действий и переходов
- Помогает структурировать приложение
- В CoDeSys есть упрощенная версия SFC

## Упражнение 8. Управление сверлильным станком

- Станок производит сверление отверстий в заготовках по заданной программе: запуск станка, опускание сверла, сверление по одному из выбранных режимов, подъем сверла.
- На станке предусмотрена кнопка запуска, тумблер выбора режима сверления, кнопка останова сверления.
- Контроллер подает три управляющие команды: опускание сверла, подъем сверла, сверление.
- Предусмотрено два режима: либо сверление производится в течение 5 секунд (автоматический режим), либо сверление производится до нажатия оператором кнопки останова сверления. Режим выбирается с помощью тумблера выбора перед запуском станка.



## Упражнение 8. Управление сверлильным станком

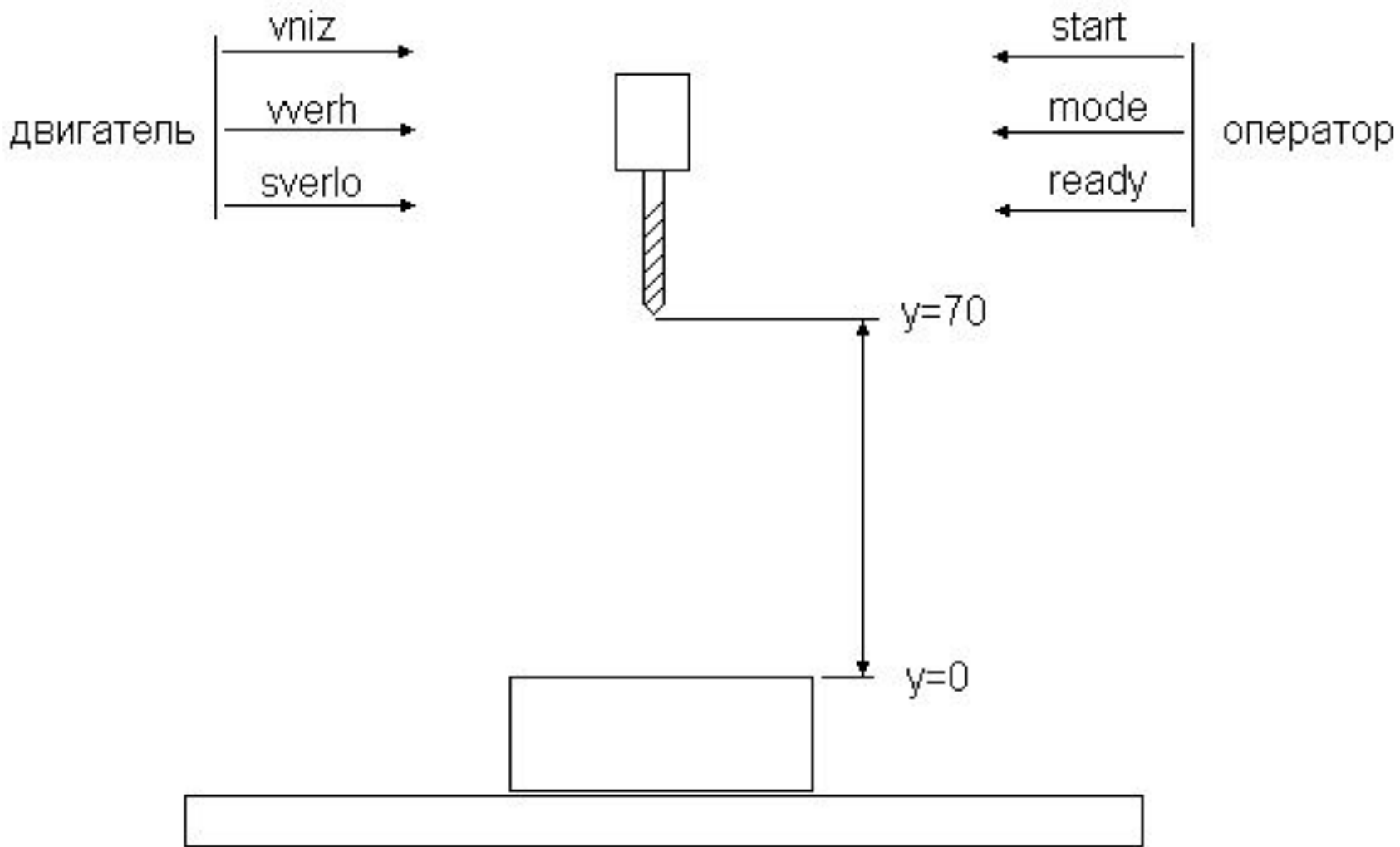
Перед началом работы оператор с помощью тумблера выбора определяет режим сверления.

После нажатия оператором кнопки запуска контроллер начинает управление станком. Подается команда опустить сверло и начинается обратный отсчет координаты. При достижении нижней точки ( $y=0$ ) снимается команда на опускание и подается команда на сверление.

Если выбран первый режим, то команда сверления снимается через 5 секунд. Если выбран второй режим, то команда сверления снимается после нажатия оператором кнопки останова сверления.

Затем контроллер подает команду на подъем сверла и начинает прямой отсчет координаты. После достижения верхнего положения ( $y=70$ ) команда подъема снимается.

# Упражнение 8. Управление сверлильным станком



# Конфигурирование задач

- Задачи выполняются по событию или циклически
- Имеют приоритет
- Вызывают программы
- Есть свободно-выполняемые задачи(аналог idle)

## Упражнение 9. Работа с конфигуратором задач

- Создать циклическую задачу
- Создать задачу, выполняемую по событию
- Создать свободно-выполняемую задачу
- Создать программы – счетчики числа запусков задач
- Проследить за выполнением свободно-выполняемой задачи, изменяя параметры других задач