## А.С. Гирник, А.Л. Федянин

# АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ КОНВЕЙЕРОМ



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ Федеральное государственное автономное образовательное учреждение высшего образования «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

А.С. Гирник, А.Л. Федянин

## АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ КОНВЕЙЕРОМ

Методические указания к выполнению лабораторной работы

Издательство Томского политехнического университета 2024

#### Гирник А.С.

Г51 Автоматизированная система управления конвейером : методические указания к лабораторным работам / А.С. Гирник, А.Л. Федянин ; Томский политехнический университет. – Томск : Издво Томского политехнического университета, 2024. – 99 с.

В методических указаниях изложена последовательность проектирования автоматизированной системы управления конвейером в составе модульного контроллера, человеко-машинного интерфейса на борту сенсорной панели и SCADA системы.

Предназначено для студентов и других сторонних слушателей, обучающихся по направлению 13.03.02 «Электроэнергетика и электротехника».

© ФГАОУ ВО НИ ТПУ, 2024
© Гирник А.С., 2024
© Оформление. Издательство Томского политехнического университета, 2024

СОДЕРЖАНИЕ	3
ВВЕДЕНИЕ	4
1. ОПИСАНИЕ УЧЕБНОГО ОСНАЩЕНИЯ	5
<ul> <li>2. ЛАБОРАТОРНЫЙ ПРАКТИКУМ</li></ul>	1 1 6
2.3. Создание программы на языках программирования СFC и FBD в среде CODESYS для контроллера XC-CPU201	e 9
<ul> <li>2.1. Создание программы на языке программпрования 51° в среде</li> <li>СОDESYS для контроллера НМІ панели визуализации XV102</li></ul>	9
<ul> <li>HMI панели визуализации XV102</li></ul>	4
2.7. Настройка ОРС сервера	2 1 3
3. УСТАНОВКА ПОГРАММНОГО ОБЕСПЕЧЕНИЯ	9 9
3.2. Установка Trace Mode 6.09	1 1 2
СПИСОК ЛИТЕРАТУРЫ	5 9

#### ВВЕДЕНИЕ

В настоящее время современные технологии позволяют создавать системы управления на базе таких микропроцессорных устройств, с помощью которых можно автоматизировать практически любой технологический процесс [1]. Ранее алгоритм управления реализовывался внутри больших релейно-контактных шкафов. Сегодня она может размещаться внутри компактной электронной памяти программируемой аппаратуры. Такие средства автоматизации не только повышают эффективность производства, но также освобождают человека от выполнения работы по контролю за состоянием технологического процесса и формированию управляющих воздействий на исполнительные органы рабочих механизмов. Применение программируемых логических контроллеров и реле в системах управления производственных объектов способствует автоматизации технологических процессов. В настоящее время данные электронные аппараты относятся к числу обладателей искусственного интеллекта.

Современные автоматизированные системы управления разделяются на три уровня:

1. Верхний уровень – управление технологическим процессом с помощью виртуальной панели оператора, которая представляет собой SCADA-программу, установленную на персональном компьютере.

2. Средний уровень – программно-электрическая часть, реализуемая с помощью программируемых логических контроллеров или реле, имеющих информационные цифровые или аналоговые входы и выходы.

3. Нижний (полевой) уровень – физические модели, исполнительные органы, управляемые элементами среднего уровня.

В совокупности все три уровня автоматизации представляют собой интеллектуальную систему управления технологическими процессами. В данном учебном пособии рассматривается реализация нижнего и среднего уровней на базе программного обеспечения и оборудования производства Eaton (Moeller). Также уделено внимание языкам программирования стандарта МЭК, на которых создаются программные алгоритмы управления такими объектами, как ленточный конвейер, автоматическая дверь, сообщающиеся сосуды и нефтяная задвижка. Эти объекты могут рассматриваться как отдельные элементы автоматики.

В учебном пособии рассматриваются программные продукты, имеющиеся в распоряжении Инженерной школы энергетики НИ ТПУ, EasySoft и CodeSys.

## 1. ОПИСАНИЕ УЧЕБНОГО ОСНАЩЕНИЯ

Все лабораторные работы, приведённые в данном учебном пособии, проводятся на специализированном оборудовании, которым оснащён учебный центр ТПУ «Технические средства автоматизации» (рис. 1):

- пять универсальных учебных стендов с контроллерами;
- лабораторная установка ленточного конвейера;
- лабораторная установка автоматической двери;

• лабораторная установка тепло-станции (сообщающихся сосудов);

• лабораторная установка нефтяной задвижки.



Рис. 1. Учебный центр «Технические средства автоматизации»

К каждому универсальному стенду (рис. 2) специальными кабелями подключена своя отдельная лабораторная установка (физическая модель).



Рис. 2. Структура универсального учебного стенда: 1 – кнопочный пост; 2 – модульный ПЛК XC-201 с набором модулей; 3, 4 – интерфейсные панели разъёмов; 5 – ПЛК ЕС4Р-222-MRAD1; 6 – модуль расширения EASY411-DC-ME; 7 – программируемое реле EASY820-DC-RC; 8 – модуль Ethernet EASY209-SE; 9 – реле безопасности Easy-Safety ES4P-221-DRXD1; 10 – блок питания EASY600 POW; 11 – блок питания EASY400 POW; 12 – HMI-панель XV-102-E6-57TVRC-10; 13 – концентратор EU5C-SWD-CAN; 14 – концентратор EASY800 with SmartWire-DT EASY806-DC-SWD; 15 – розетка подключения к CANopen HMI-панели; 16 – розетка подключения к CANopen концентратора EU5C-SWD-CAN; 17 – кнопочный пост; 18 – автомат NZMN2-ME90; 19 – SWD-модуль NZM-XSWD-704 соединения для NZM; 20 – пусковая сборка MSC-DEA-12-M7; 21 – панель сетевых коммуникаций; 22 – персональный компьютер; 23 – стол; 24 – стойки; 25 – система подвески оборудования



Рис. 3. Подключение всего учебного оборудования в общую сеть Ethernet





На рис. 4, 5 показаны интерфейсные панели разъёмов для электрического соединения:

1) кнопок управления с входами аппаратов (реле, контроллеры);

2) сигнальных ламп с выходами аппаратов;

3) датчиков физических моделей с входами аппаратов;

4) исполнительных элементов физических моделей (контакторы пуска двигателя, соленоиды и т. д.) с выходами аппаратов.

Первая панель разъёмов (рис. 4) разделена на 3 основных секции:

1. Синяя секция. В ней расположены электрические разъёмы, которые соединены со входами программируемых реле и контроллеров. Причём для каждого аппарата предусмотрена пара дублированных разъёмов. То есть на каждом из двух разъёмов контакты с одинаковыми номерами соединены параллельно друг другу. Это необходимо для того, чтобы можно было подключать ко входам контроллера как кабель для передачи сигналов от датчиков лабораторной установки, так и кабель для передачи сигналов управления от кнопок на универсальном стенде.

2. Серая секция. В ней расположены электрические разъёмы, которые соединены с выходами программируемых аппаратов. Здесь точно так же предусмотрено парное дублирование разъёмов, для того чтобы можно было подавать управляющие сигналы от контроллеров как на лабораторную установку, так и на сигнальные лампы универсального стенда.

3. *Чёрная секция*. В ней расположены электрические разъёмы, через которые можно подключаться к датчикам лабораторной установки, а также к её исполнительным элементам. О

Вторая панель разъёмов (рис. 5) предназначена для подключения к модульному контроллеру XC-201 и его отдельным модулям. Эта панель также условно разделена на секции: синие (для входных сигналов), серые (для выходных сигналов) и черные (для ручного управления от кнопок на универсальном стенде).

На рис. 6–8 показаны электрические схемы соединения первой панели разъёмов со входами и выходами таких программируемых электронных аппаратов, как:

• программируемый логический контроллер EC4P-222-MRAD1;

- программируемое реле EASY820-DC-RC;
- реле безопасности ES4P-221-DRXD1.



XIOC-8AI-I2

XIOC-8DO

XIOC-4AO-U1

XC-CPU201

Рис. 5. Вторая интерфейсная панель разъёмов



Рис. 6. Подключение контроллера EC4P-222-MRAD1 к первой панели разъёмов



Рис. 7. Подключение реле EASY820-DC-RC к первой панели разъёмов



Рис.8. Подключение реле ES4P-221-DRXD1 к первой панели разъёмов

На рис. 9, 10 показаны электрические схемы соединения первой панели разъёмов с согласующими отдельными разъёмами (расположенными на задней стороне универсального стенда) для подключения к лабораторной установке.



Рис. 10. Выходы лабораторной установки



На рис. 11, 12 показаны электрические схемы соединения второй панели разъёмов со входами и выходами модулей контроллера XC-CPU201.

Рис. 11. Подключение выходов модулей контроллера XC-CPU201 ко второй панели разъёмов





На рис. 13 показаны электрические схемы подключения кнопок управления и сигнальных ламп ко второй панели разъёмов.



Рис. 13. Подключение элементов ручного управления ко второй панели разъёмов

На рис. 14—16 показаны электрические соединения внутри универсальных кабелей, с помощью которых можно подключать входы и выходы контроллеров к лабораторной установке через первую и вторую панели разъёмов.



Рис. 14. Кабели ручного управления



Рис. 15. Кабель подключения лабораторной установки к входам контроллера



Рис. 16. Кабель подключения лабораторной установки к выходам контроллера

На рис. 17 показаны варианты подключения любого из контроллеров к лабораторной установке: напрямую (A) и через реле безопасности (B). На рис. 18 показана электрическая схема разветвителя, который предназначен





Рис. 17. Варианты подключения лабораторной установки к выходам разных контроллеров



Рис. 18. Разветвитель сигналов от конвейера к модульному контроллеру

### 2. ЛАБОРАТОРНЫЙ ПРАКТИКУМ

Цель: научиться создавать комплекс программ для автоматизированной системы управления конвейером на разных языках программирования в средах разработки CODESYS, TRACEMODE и GALILEO.

#### Порядок выполнения работы

#### 2.1. Изучение лабораторной установки «Конвейер»

Физическая модель конвейера представляет собой установку с транспортной лентой, приводимой в движение ведущим барабаном, который, в свою очередь, вращается от приводного двигателя. Над лентой расположена оптическая система для контроля габаритов транспортируемых грузов и их высоты. В конце транспортного пути расположены две корзины, предназначенные для приёма как высоких, так и низких грузов, которые сбрасываются поворотной площадкой. Выбор корзины для сброса груза осуществляется системой управления. Под каждой корзиной имеются датчики веса грузов (рис. 19).



Рис. 19. Абстрактная модель конвейера: 1 – транспортируемый груз; 2 – электродвигатель асинхронный; 3 – инфракрасный луч; 4 – лента; 5 – источник света; 6 – оптический датчик; 7 – направляющая поворотная площадка для спуска грузов в корзины; 8 – датчики перемещения для измерения веса грузов; 9 – приёмные корзины для сбора грузов; 10 – силовой шкаф с электрооборудованием



Внешний вид установки конвейера приведён на рис. 20.

Рис. 20. Лабораторная установка конвейера

Система управления конвейером выполняет следующие задачи:

1. Контроль позиционирования грузов по ширине ленты. Реализуется с помощью инфракрасных лучей, направленных вдоль ленты.

2. Контроль заданной высоты транспортируемых грузов за счёт использования двух световых лучей на двух разных высотах поперёк направления движения ленты.

3. Контроль наполнения корзин грузами с помощью двух датчи-ков перемещения.

4. Контроль скорости ленты с помощью оптического датчика на ведущем барабане. Датчик подсчитывает число свето-импульсов через вращающееся зеркало, измеряя также и длительность между импульсами.

В табл. 1 и 2 приведены способы подключения датчиков и исполнительных элементов к входам и выходам программируемых аппаратов универсального стенда в соответствии с электрической схемой (рис. 21).

## Таблица 1

		Поступление на входы				
Назначение	Контакт на разъёме DB-15	XC-CPU201	XIOC-8AI-U1	EASY820	ES4P	EC4P (EASY411)
Левый габарит	1	—	U0+	15	<i>I5</i>	<i>I5</i>
Правый габарит	2	_	U1+	16	<i>I6</i>	<i>I6</i>
Высокий груз	3	-	U2+	<i>I</i> 7*	I7*	I7*
Низкий груз	4	-	U3+	<i>I8</i> *	<i>I8</i> *	<i>I8</i> *
Вес левой корзины	5	_	U4+	<i>I11*</i>	<i>I11*</i>	<i>I11*</i>
Вес правой корзины	6	_	U5+	<i>I12*</i>	<i>I12</i> *	<i>I12</i> *
Ведущий вал	7	-	U6+	<i>I9</i>	19	(IA1)
Пуск двигателя	9	I0.0	_	<i>I1</i>	<i>I1</i>	<i>I1</i>
Останов двигателя	10	I0.1	-	<i>I2</i>	<i>I2</i>	<i>I2</i>
Скорость ниже	11	I0.2	-	I3	I3	I3
Скорость выше	12	I0.3	_	<i>I4</i>	<i>I</i> 4	<i>I4</i>
Общий провод	15	0	0	0	0	0

# Варианты подключения датчиков к различным аппаратам универсального стенда

### Таблица 2

	Управление с выходов						
Назначение	Контакт на разъёме DB-25	XC-CPU201	XIOC-8DO	XIOC-4A0-UI	EASY820	ES4P	EC4P (EASY411)
Прямой пуск	1	Q0.0	0		Q1	QS1	Q1
Сброс груза влево	2	Q0.1	1		Q2	QS2	<i>Q</i> 2
Сброс груза вправо	3	Q0.2	2		<i>Q3</i>	QS3	<i>Q3</i>
Двоичное задание частоты 001	4	Q0.3	3		<i>Q4</i>	QS4	<i>Q4</i>
Двоичное задание частоты 010	5	Q0.4	4		Q5	_	Q5
Двоичное задание частоты 100	6	Q0.5	5		Q6	_	Q6
Включение ПЧ ди- станционно	7	_	6		_	_	(S1)
_	8	_					—
Аналоговое задание частоты	9	_		V0+	QA1		(QA1)
Аналоговое ПИД- управление	10	_		Vl+			(QA2)
Общий провод	25	0	0	0	0	0	0

Подключение исполнительных элементов к различным аппаратам





#### 2.2. Общая концепция системы управления в составе ОРС сервера

Общий принцип работы подобной системы управления состоит в следующем:

- Передача управления сигналов (пакетов данных) стороны SCADA осуществляются co системы на персональном компьютере в сторону ОРС сервера для записи значения управляющего сигнала в символьный файл. В нашем случае ОРС сервер в виде специальной программы находится в составе самого компьютера. В отдельных случаях данные сигналы управления можно передавать дистанционно в OPC сервер по протоколу TCP/IP Ethernet через специальный кабель.
- Значение управляющего сигнала из символьного файла передаётся в программируемый логический контроллер для запуска в нём определённого программного алгоритма. В нашем случае контроллер подсоединён в общую информационную сеть Ethernet с компьютером.
- Электрический сигнал управления передаётся со стороны контроллера в сторону исполнительных органов лабораторной установки конвейера.
- Передача сигналов с датчиков лабораторной установки конвейера осуществляется в обратном порядке в SCADA систему.
- Дополнительным звеном управления в системе является HMI панель, которая соединена с контроллером специальным кабелем для двусторонней передачи данных по сетевому протоколу CAN Open. В данном случае обмен данных проиводится только между контроллером и самой панелью. ОРС сервер напрямую здесь не участвует. Но принятые контроллером сигналы управления от HMI панели могут поступать далее в ОРС сервер уже по протоколу TCP/IP Ethernet, попадая в символьный файл и далее в SCADA систему.

Общая концепция работы данной системы управления в виде структурной схемы приведена на рис. 22.



Рис. 22. Общая концепция системы управления

На рис. 23 показан внешний вид универсального учебного стенда, на борту которого находится такое необходимое для системы управления оборудование, как программируемый логический контроллер XC-CPU201, HMI панель серии XV102, кнопки управления, персональный компьютер, панели разъёмов.



Рис. 23. Внешний вид учебного стенда

Всё программируемое оборудование данного стенда подключено в общую информационую сеть Ethernet по рис. 3. Это в первую очередь необходимо для загрузки программных алгоритмов во внутренню память оборудования со стороны персонального компьютера.

#### 2.3. Создание программы на языках программирования CFC и FBD в среде CODESYS для контроллера XC-CPU201

- 1. Запустите программную среду CODESYS 2.3.9.
- 2. Создайте новый проект: «Файл/Создать» (File/New).

3. Настройте целевую платформу (Target Settings). Откройте вкладку «Целевая платформа» (Target Settings) и на странице диалогового окна «Конфигурация» (Configuration) выберите тип контроллера **XC-CPU201-EC512K-8DI-6DO-XV V2.3.9 SP8**. Во вкладке основных настроек (General) должна стоять галочка напротив «Download symbol file». Во вкладке сетевой функциональности (Network functionality) поставьте галочку на против «Support network variables» и в открывшемся поле ввода пропишите «CAN» (рис. 24). Подтвердите ввод нажатием кнопки OK.

Target Settings	×
Configuration: XC-CPU201-EC512K-8D1-6D03 Target Platform Memory Layout General Net Support parameter manager Index ranges Index ranges for parameters: Index ranges for variables: 16#2000-16#5fff Index range for mappings: Subindex range: 1-100 Example of syntax of ranges: 16#2000-16#2010;16#2500-16#2600	<ul> <li>✓ V2.3.9 SP8</li> <li>✓ Visualization</li> <li>✓ Support network variables</li> <li>Names of supported network interfaces:</li> <li>CAN</li> <li>Example of a name list: CAN;UDP;DP;DEVNET max. 7 characters/name !</li> </ul>
	Default OK Cancel

Рис. 24. Установка поддержки сетевой передачи данных через CAN Open протокол

4. Создайте главную программу. После настройки целевой платформы в автоматически появившемся диалоговом окне New POU выберите тип компонента «Программа» (Program), язык CFC, задайте имя программы PLC\_PRG.

5. *Настройке конфигурацию контроллера*. Зайдите во вкладку ресурсов «Resources» проекта и откройте конфигурацию контроллера «PLC Configuration» (рис. 25). Далее нажмите правой клавишей мыши по верхнему уровню меню (Configuration XC-CPU201...) и добавьте «CanMaster». Нажмите внутри опции «Non Display[SLOT]» по «EMPTY-SLOT» правой клавишей мыши и выберите элемент «XIOC-8AI-U1» - это модуль расширения для приёма сигналов 10В, но уже преобразованных из 24В от датчиков конвейера.



Рис. 25. Конфигурация контроллера

Общая структура главной программы на языке CFC показана на рис. 26, а её переменные – на рис. 27. Чтобы создать данную программу, необходимо сначала создать дополнительные функциональные блоки, а затем соединить их между собой.

Предварительно разберём принцип работы данной программы.

Запуск двигателя производится с помощью блока start\_motor путём подачи от кнопки пуска на вход start сигнала TRUE, при условии, что на контрольные входы slowdown, left\_dimension, right\_dimension и full\_weight поступают сигналы FALSE.

Разберём остальные блоки, подающие сигналы на контрольные входы start\_motor. Блок lowdown сравнивает скорость вращения ведущего барабана с минимальной уставкой. На вход этого блока поступает сигнал от блока velocity\_measure о величине скорости этого барабана. Последний блок вычисляет скорость вращения барабана по интервалу между импульсами от датчика velo\_sensor. Таким образом, контролируется снижение скорости из-за возможной неисправности двигателя.

Сигнал о наполнении любой из корзин поступает от переменных weight\_high и weight\_low через промежуточные блоки инерционной задержки weight\_high\_inertial и weight\_low\_inertial на входы аварийного контроля в блок start\_motor.

Приведение в движение поворотной площадки осуществляется с помощью блока stage\_check совместно с блоками movie\_stage и movie\_stage2. При этом подаются сигналы в переменные movie\_high и movie\_low, далее с физических выходов %QX0.1 и %QX0.2, связанных с этими переменными, поступают сигналы на привод поворотной площадки сброса грузов.

Сигнал запуска двигателя поступает от переменной work\_motor, связанной с физическим выходом %QX0.0.

Из переменных Q4, Q5, Q6, связанных с физическими выходами %QX0.3, %QX0.4, %QX0.5, подаётся сигнал на специальные входы преобразователя частоты для задания скорости вращения двигателя.



Рис. 26. Корневая программа PLC\_PRG

6. Объявите переменные корневой программы PLC\_PRG. Пропишите переменные в соответствии с рис. 27. Сделайте привязку переменных к физическим входам и выходам контроллера с помощью оператора «АТ».



Вообще, чтобы определить, как правильно прописывать в такой ситуации физические входы и выходы, необходимо в разделе «Ресурсы» (Resources) открыть вкладку «Конфигурация контроллера» (PLC Configuration) и в диалоговом окне выбрать опцию Configuration XC-CPU201, а в ней – АТ %IB0: BYTE (Local Inputs) и АТ%QB0: BYTE (Local Outputs). Объявленные переменные имеют следующие значения:

• start\_button AT %IX0.0: BOOL – переменная, принимающая сигнал от кнопки пуска двигателя;

• stop\_button AT %IX0.1: BOOL – переменная, принимающая сигнал от кнопки остановки двигателя;

• rotation\_up AT %IX0.2: BOOL – переменная, принимающая сигнал от кнопки увеличения частоты вращения двигателя;

• rotation\_down AT %IX0.3: BOOL – переменная, принимающая сигнал от кнопки уменьшения частоты вращения двигателя;

• work\_motor AT %QX0.0: BOOL – переменная, подающая сигнал для пуска двигателя;

• movie\_high AT %QX0.1: BOOL – переменная, подающая сигнал для поворота площадки в сторону корзины для высоких грузов;

• movie\_low AT %QX0.2: BOOL – переменная, подающая сигнал для поворота площадки в сторону корзины для низких грузов;

• Q4 AT %QX0.3: BOOL – переменная, подающая сигнал первого регистра для задания частоты вращения двигателя через ПЧ;

• Q5 AT %QX0.4: BOOL – переменная, подающая сигнал второго регистра для задания частоты вращения двигателя через ПЧ;

• Q6 AT %QX0.5: BOOL – переменная, подающая сигнал третьего регистра для задания частоты вращения двигателя через ПЧ;

• left\_sensor AT %IW6: WORD – переменная, принимающая сигнал перекрытия инфракрасного луча левого габарита;

• right\_sensor AT %IW8: WORD – переменная, принимающая сигнал перекрытия инфракрасного луча правого габарита;

• high\_sensor AT %IW10: WORD – переменная, принимающая сигнал от датчика высоких грузов;

• low\_sensor AT %IW12: WORD – переменная, принимающая сигнал от датчика низких грузов;

• weight\_high AT %IW14: WORD – переменная, принимающая сигнал от датчика веса под корзиной для высоких грузов;

• weight\_low AT %IW16: WORD – переменная, принимающая сигнал от датчика веса под корзиной для низких грузов;

• velo\_sensor AT %IW18: WORD – переменная, принимающая импульсы от фотодатчика для ведущего барабана;

• left\_dimension: BOOL – переменная выхода груза за левый габарит ленты конвейера;

• right\_dimention: BOOL – переменная выхода груза за правый габарит ленты конвейера;

• velocity\_mesure: velocity – локальное в PLC\_PRG имя функционального блока velocity; • slowdown\_check: slowdown – локальное в PLC\_PRG имя функционального блока slowdown;

• movie\_stage: stage – локальное в PLC\_PRG имя функционального блока stage; movie\_stage2 – то же;

• stage\_check: stage\_prove – локальное в PLC\_PRG имя функционального блока stage prove;

• start\_motor: motor – локальное в PLC\_PRG имя функционального блока motor;

• weight\_height\_inertial: inertial – локальное в PLC\_PRG имя функционального блока inertial;

• weight\_low\_inertial: inertial – локальное в PLC\_PRG имя функционального блока internal;

• velocity\_inertial: inertial – локальное в PLC\_PRG имя функционального блока internal;

• M\_M: Manage\_motor – локальное в PLC\_PRG имя функционального блока Manage\_motor;

• scada\_start\_button – переменная приёма сигнала от Scada системы для запуска двигателя;

• scada\_stop\_button – переменная приёма сигнала от Scada системы для останова двигателя;

• scada\_rotation\_up – переменная приёма сигнала от Scada системы для увеличения скорости вращения двигателя;

• scada\_rotation\_down – переменная приёма сигнала от Scada системы для уменьшения скорости вращения двигателя.

7. Объявите глобальные переменные. Пропишите их в разделе Global Variables, как показано на рис. 28. Эти переменные будут использоваться в остальных функциональных блоках и имеют следующее назначение:

• minimum\_velocity – минимальная скорость ленты;

• sensor\_way – геометрический путь по длине окружности между зеркалами на барабане конвейера;

• interval\_init – время удержания поворотной площадки в наклонённом положении, чтобы после перекрытия инфракрасного луча грузом он успел упасть в корзину.


Рис. 28. Глобальные переменные

8. Создайте дополнительные функциональные блоки.

• Создайте функциональный блок запуска двигателя. В первой системной вкладке, где список POU, щёлкните правой клавишей мыши и в появившемся контекстном меню выберите команду «Добавить объект» (Add Object). В автоматически открывшемся диалоговом окне выберите тип объекта Function Block, задайте ему имя motor и язык ST (рис. 29).

🍤 File	Edit	Project	Insert	Extras	Online	Window	Hel	р
1		JE 🗊 🖉	<u>ا ال</u>	╘╘	<b>3</b>	è 🔒 🐅	<b>S</b> ,	100
		Add Obje Rename ( Edit Obje Copy Obj Delete Ob Convert ( Object Pr Project di Add Actio	ct Dbject ct operties. atabase					100 0001 0002 0003 0004 0005 0006 0007 0008 0009 0010 0011 0012 0013 0014
	st.	New Fold Expand N Collapse View Inst Show Cal Save as te Exclude fi	er lode Node ance I Tree :mplate rom buil	 d ualization	s 🚛 Res	sources		0015 0016 0017 0018 0019 0020 0021 0022 0023 0024 Cod 0 Er < [
Adds a ne	ew obie	ect to the lis	t on the le	eft side.				

Рис. 29. Добавление нового функционального блока

Объявите в блоке motor следующие входные переменные:

- start типа BOOL для запуска двигателя;
- stop типа BOOL для остановки двигателя;

– slowdown типа BOOL – переменная замедления ленты конвейера;

 left\_dimension типа BOOL – переменная наличия выхода транспортируемого груза за левую границу (перекрытие светового луча левого габарита);

 right\_dimension типа BOOL – переменная наличия выхода транспортируемого груза за правую границу (перекрытие светового луча правого габарита); – full\_weight типа BOOL – переменная полного заполнения грузами любой из приёмных корзин.

Объявите в блоке motor выходную переменную work типа BOOL – переменная, которая в состоянии TRUE подаёт сигнал на запуск двигателя.

В рабочей области редактора блока motor пропишите программный код на языке программирования ST (рис. 30).



Рис. 30. Окно функционального блока motor

Данный программный код осуществляет проверку условий безаварийной транспортировки: если переменные slowdown, left\_dimension, right\_dimension, full\_weight равны FALSE, то переменной work paspeшено принимать значение TRUE. Также данный код проверяет условия запуска и останова двигателя: если переменная start равна TRUE (нажата кнопка пуска конвейера), то переменная work принимает значение TRUE; если переменная stop равна TRUE (нажата кнопка стопа конвейера), то переменная work принимает значение FALSE.

• Создайте функциональный блок контроля замедления ленты. В первой системной вкладке, где список POU, щёлкните правой клавишей мыши и в появившемся контекстном меню выберите команду «Добавить объект» (Add Object). В автоматически открывшемся диалоговом окне выберите тип объекта «Функциональный блок» (Function Block), задайте ему имя slowdown и язык ST.

Объявите в блоке slowdown входную переменную velocity типа REAL – это переменная скорости барабана, приводящего в движение ленту.

Объявите в блоке slowdown выходную переменную alarm – переменная типа BOOL, которая сигнализирует о замедлении ленты конвейера.

Программный код функционального блока slowdown показан на рис. 31.

🎭 slowdown (FB-ST)	- • •
0001 FUNCTION_BLOCK slowdown	
0002 VAR_INPUT	
0004 END VAR	
0005 VAR_OUTPUT	
0006 alarm: BOOL := FALSE;	
0007 END_VAR	
0008 VAR	
0009 END_VAR	
<	>
0001 IF velocity < minimum_velocity	
0002 THEN alarm:=TRUE; ELSE alarm:=FALSE;	
0003END_F;	
0005	
0006	
<	>

Рис. 31. Окно функционального блока slowdown

• Создайте блок измерения скорости вращения барабана. В первой системной вкладке, где список POU, щёлкните правой клавишей мыши и в появившемся контекстном меню выберите команду «Добавить объект» (Add Object). В автоматически открывшемся диалоговом окне выберите тип объекта «Функциональный блок» (Function Block), задайте ему имя velocity и язык ST.

Объявите в блоке velocity входную переменную sensor типа BOOL – переменная наличия импульса от фотодатчика, принимающего свет от зеркала, которое вращается вместе с барабаном.

Объявите в блоке velocity выходную переменную velocity типа REAL – переменная, выдающая значение окружной скорости вращения барабана в см/с.

Объявите в блоке velocity следующие локальные переменные:

– block – переменная типа BOOL, необходимая для фиксации момента замера времени появления импульса от фотодатчика;

 time1 – переменная типа ТІМЕ, принимающая значение момента времени появления «предыдущего» импульса от фотодатчика;

– time2 – переменная типа ТІМЕ, принимающая значение момента времени появления «следующего» импульса от фотодатчика;  interval\_time – переменная типа TIME, принимающая значение интервала между «предыдущим» и «следующим» импульсами от фотодатчиков;

interval\_dword – переменная типа DWORD (двойное слово),
 т. е. это то же время interval\_time, только в формате двойного слова;

– interval\_real – переменная типа REAL, это то же время interval\_dword, только измеряется в миллисекундах.

Вставьте в рабочую область блока velocity код (рис. 32).



Рис. 32. Окно функционального блока velocity

Принцип работы кода состоит в следующем. При подаче на вход блока velocity в переменную sensor значения TRUE, так как block=TRUE, переменная time1 принимает значение начального момента времени, затем переменной блокировки block присваивается значение FALSE. Теперь даже при повторном чтении данного кода вычислительной машиной (ПК), уже при block=FALSE, первое IF-условие уже не

будет выполняться, а значит, переменная time1 далее будет сохранять своё ранее принятое значение. Переменная interval\_time будет равна 0 на начальном этапе прочтения компьютером кода. При снятии сигнала sensor, т. е. при переводе его в состояние FALSE, переменная блокировки block принимает значение TRUE, тем самым производится подготовка к следующему запуску первого IF-условия для последующей стадии расчёта временного интервала. Пока sensor находится в FALSE, значение переменной time2 увеличивается в режиме реального времени. Как только sensor станет TRUE, сработает первый оператор IF и в нём произойдёт расчёт разности времени между конечным значением time2 и начальной временной точкой time1, которая до этого момента сохранялась постоянной. Таким образом, происходит вычисление времени между «предыдущим» импульсом sensor и «последующим». После этого time1 принимает новое «стартовое» время, и когда блокиратор block станет FALSE, time1 останется постоянным до следующего такого же импульса sensor. После каждого расчёта разности времени между импульсами sensor производится вычисление скорости движения ленты конвейера путём деления расстояния, пройденного лентой от одного момента подачи sensor до следующего, на время между этими подачами сигнала sensor. Причём это время и есть та разница interval\_time:=time2time1. Подача импульсов sensor напрямую связана с отражением света от зеркал на барабанах конвейера и попаданием его на фотодатчики.

• Создайте функциональный блок сигнала сброса груза. В первой системной вкладке, где список POU, щёлкните правой клавишей мыши и в появившемся контекстном меню выберите команду «Добавить объект» (Add Object). В автоматически открывшемся диалоговом окне выберите тип объекта «Функциональный блок» (Function Block), задайте ему имя stage и язык ST.

Объявите в блоке stage входную переменную в редакторе объявлений sensor типа BOOL – переменная наличия импульса, когда перекрывается инфракрасный луч контроля высоты груза.

Объявите в блоке stage выходную переменную movie типа BOOL – переменную, с которой подаётся сигнал на поворот площадки для сброса груза в корзину.

Объявите в блоке stage следующие локальные переменные:

- time1 переменная типа ТІМЕ;
- time2 переменная типа TIME;
- interval\_time переменная типа TIME;

– interval\_dword – переменная типа DWORD (двойное слово),

т. е. это то же время interval\_time, только в формате двойного слова;

interval\_real – переменная типа REAL, то же время interval\_dword, только измеряется в миллисекундах.
 Вставьте в рабочую область блока stage код (рис. 33).

stage (FB-ST)	
0003 sensor BOOL := FALSE:	
0004 END VAR	
0005 VAR OUTPUT	
0006 movie: BOOL := FALSE;	
0007 END_VAR	
0008 VAR	
0009 time1: TIME :=T#0s;	
0010 time2: TIME;	
0011 interval_time: TIME;	
0012 interval_dword: DWORD;	
0013 interval_real: REAL;	
0014 END_VAR	
0015 VAR CONSTANT	
0016 END_VAR	
0017	
	+
0001 IF sensor = TRUE	
0002 THEN	
0003 movie:=TRUE;	
0004 END_IF;	
0005	
0006 IF sensor = FALSE	
0008 time2:=1ME();	
0009 Interval_time:=time2-time1;	
0010 Interval_dword.=TIME_TO_DWORD(Interval_time), 0011 interval_sol:=DWORD_TO_REAL(interval_time),	
0011 Interval_real/1000) > interval_init)	
0013 mevie:=EALSE:	
0015 END IF	
0016 ELSE	
0017time1:=TIME():	
0018 END_IF;	

Рис. 33. Окно функционального блока stage

• Создайте функциональный блок координации поворотной площадки. В первой системной вкладке, где список POU, щёлкните правой клавишей мыши и в появившемся контекстном меню выберите команду «Добавить объект» (Add Object). В автоматически открывшемся диалоговом окне выберите тип объекта «Функциональный блок» (Function Block), задайте ему имя stage\_prove и язык ST.

Объявите в блоке stage\_prove следующие входные переменные:

 high\_input – переменная типа BOOL, принимающая сигнал при перекрытии светового луча высоким грузом; – low\_input – переменная типа BOOL, принимающая сигнал при перекрытии светового луча низким грузом.

Объявите в блоке stage\_prove следующие выходные переменные:

– move\_high – переменная типа BOOL, подающая сигнал поворота площадки в сторону корзины для высоких грузов;

– move\_low – переменная типа BOOL, подающая сигнал поворота площадки в сторону корзины для низких грузов.

Вставьте в рабочую область блока stage\_prove код (рис. 34).

🎭 stage_prove (FB-ST)	
0001 FUNCTION_BLOCK stage_prove	
0002VAR_INPUT	
0003 high_input: BOOL;	
0004 low_input: BOOL;	
0005END_VAR	
0006 VAR_OUTPUT	
0007 movie_high: BOOL;	
0008 movie_low: BOOL;	
0009 END_VAR	
UTTIEND_VAR	
	Þ
O001 IF high_input = TRUE AND low_input = TRUE	
O001 IF high_input = TRUE AND low_input = TRUE     O002 THEN	
<pre> 0001 IF high_input = TRUE AND low_input = TRUE 0002 THEN 0003 movie_high:=TRUE; </pre>	
<pre> 0001 IF high_input = TRUE AND low_input = TRUE 0002 THEN 0003 movie_high:=TRUE; 0004 movie_low:=FALSE;</pre>	
<pre></pre>	
<pre>0001 IF high_input = TRUE AND low_input = TRUE 0002 THEN 0003 movie_high:=TRUE; 0004 movie_low:=FALSE; 0005 ELSE 0006 IF high_input = FALSE THEN movie_high:=FALSE; END_IF; 0007 IF high_input = TRUE THEN movie_low:=FALSE; END_IF; 0008 IF low_input = FALSE THEN movie_low:=FALSE; END_IF; 0009 IF low_input = TRUE THEN movie_low:=TRUE; END_IF; 0010 END_IF;</pre>	

Рис. 34. Окно функционального блока stage\_prove

Смысл блока stage\_prove – предотвратить ложную подачу сигналов на поворот площадки сброса грузов одновременно в одну и в другую стороны, когда высокий груз перекрывает два световых луча, контролирующих, соответственно, высокие и низкие грузы.

• Создайте функциональный блок инерционной задержки сигнала. В первой системной вкладке, где список POU, щёлкните правой клавишей мыши и в появившемся контекстном меню выберите команду «Добавить объект» (Add Object). В автоматически открывшемся диалоговом окне выберите тип объекта «Функциональный блок» (Function Block), задайте ему имя inertial и язык FBD.

Необходимость в этом блоке обусловлена вероятностью ложных импульсов с датчиков веса, что может привести к остановке конвейера за счёт ложного поступления сигнала на вход full\_weight блока motor. Также этот блок нужен для предотвращения ложной остановки двигателя через блок slowdown, пока лента набирает скорость при пуске.

Программный код функционального блока inertial показан на рис. 35.

🍤 inertial (FB-FBD)	
0001 FUNCTION_BLOCK inertial	
0002 VAR_INPUT	
0003 input: BOOL;	
0004 delay: TIME;	
0005END_VAR	
0000 VAR_001P01	
0009 VAR	
0010 timer: TON:	
0011 END_VAR	
<	>
0001	
timer	
TON	
input-IN C Q output	
delay-PT C ET-	
	· · · ·

Рис. 35. Окно функционального блока inertial

• Создайте функциональный блок задания скорости вращения двигателя. В первой системной вкладке, где список POU, щёлкните правой клавишей мыши и в появившемся контекстном меню выберите команду «Добавить объект» (Add Object). В автоматически открывшемся диалоговом окне выберите тип объекта «Функциональный блок» (Function Block), задайте ему имя Manage\_motor и язык ST. Этот блок необходим в случае использования частотного преобразователя, когда включены контакторы К2 и К3 (см. рис. 21).

Задайте переменные и создайте программный код, как показано на рис. 35.

У данного блока есть две входные переменные – rotation\_up и rotation\_down – для увеличения или уменьшения внутренней переменной CV счётчика CTUD\_1. Переменная CV передаёт своё значение на внутреннюю переменную pv функционального блока Manage\_motor. С помощью условных операторов производится проверка значения переменной pv, на основании которого формируется двоичный 3-разрядный код. Этот код поразрядно подаётся на выходные переменные Q4, Q5, Q6 функционального блока Manage\_motor. Последние значения двоичного кода в виде трёх отдельных сигналов подаются через электрические выходы контроллера на входы DI3, DI4, DI5 преобразователя частоты (см. рис. 21).

S Manage_motor (FB-ST)	
0001 FUNCTION BLOCK Manage motor	
0002 VAR_INPUT	
0003 rotation_up:BOOL;	
0004 rotation_down:BOOL;	
0005END_VAR	
0006 VAR_OUTPUT	
0007 Q4:BOOL;	
0008 Q5:BOOL;	
0009 Q6:BOOL;	
0010 END_VAR	
0011 VAR	
0012 CTUD_1: CTUD;	
0013 pv: INT;	
0014 Reset: BOOL := FALSE;	
0015END_VAR	
<	>
0001 IF pv < 7 THEN rotation_up:=rotation_up; ELSE rotation_up:=	FALSE; END_IF
0002	
0003CTUD_1(	
0004 CU:= rotation_up,	
0005 CD:= rotation_down,	
0006 RESET:=Reset,	
0007 LOAD:= ,	
0008 PV:= 7,	
0009 QU=>,	
0010 QD=>,	
0011 CV=> pv);	
	(*000*)
0013 IF pV=0 THEN Q4:=FALSE; Q5:=FALSE; Q6:=FALSE; END_IF	(*000*)
0014IF pv=1 THEN Q4.=TRUE, Q5.=FALSE, Q0.=FALSE, END_IF	(*010*)
0015 PV=2 THEN Q4:=FALSE, Q5:=TRUE, Q6:=FALSE, END_F	(*010*)
001701F pv=3 THEN 04:-EALSE: 05:-EALSE: 08:-TRUE: END_IF	(*100*)
0018 PV-4 THEN Q4FALSE, Q3FALSE, Q0TRUE, END_F	(*101*)
001011 pv=5 THEN 04:=FALSE: 05:=TRUE: 06:=TRUE: END_IE	(*110*)
0020 IF nv=7 THEN Q4:=TRUE: 05:=TRUE: 06:=TRUE: END IF	(*111*)
ween pretimentar. Hitter, au-Hitter, au-Hitter, end_in	( ) ) J

Рис. 36. Окно функционального блока Manage\_motor

9. Объявите переменные передачи данных по CAN Open. Перейдите на вкладку «Ресурсы» (Resources), и далее в подраздел «Глобальные переменные» (Global\_Variables). Кликните правой клавишей мыши по разделу (папке) глобальных переменных добавьте новый объект с настройками как на рис. 37.

Properties	?	×
Global Variable List		
Name of the global variable list: CAN_XC201		
Link to file		
Filename: Browse	(	
Import before compile     C Export before compile	Add net	work
Connection 1 (CAN)		
Network type CAN  Settings	Remo	ve
Pack variables		
List identifier (COB-ID):		
Transmit checksum		
C Acknowledgement		
Read Request on bootup		
Vrite Answer bootup requests		
Cyclic transmission Interval: T#50ms		
✓ Transmit on change Minimum gap: T#20ms		
Transmit on event Variable:		
ОК	Ca	ncel

Рис. 37. Добавление нового объекта в глобальные переменные

Задайте переменные передачи данных по протоколу CAN Open от НМІ панели в контроллер, как показано на рис. 38.



Рис. 38. Переменные передачи САN Ореп данных

10. Проверьте работоспособность программы в режиме эмуляции. Для запуска проекта в режиме эмуляции установите во вкладке «Онлайн» (Online) галочку против «Эмуляция» (Simulation). Откомпилируйте проект: «Проект/Компилировать всё» (Project/Rebuild All). Установите соединение с контроллером: «Онлайн/Соединение» (Online/Login). Запустите проект: «Онлайн/Запуск» (Online/Run). Поэкспериментируйте с заданием различных переменных.

11. Сформируйте символьный файл с тегами OPC переменных. Для этого откройте опции проекта в Project/Options. Далее зайдите в символьную конфигурацию и в окне установки атрибутов выберите все те переменные программы PLC\_PRG. Эти переменные отвечают за следующее: приём сигналов с электрических кнопок, приём сигналов с нопок и задатчиков от SCADA системы, вывод управляющих сигналов с контактных разъёмов контроллера, посылку сигналов в SCADA систему (рис. 39).



Рис. 39. Конфигурация символьного файла

12. Откомпилируйте проект «Проект/Компилировать всё» (Project/Rebuild All). В результате в папке проекта должен появиться символьный файл с расширением «SYM». В составе тегов символьного файла будут присутствовать следующие записи:

PLC\_PRG.left\_dimension:BOOL:4:541:1:b:16#00000040 PLC\_PRG.Q4:BOOL:2:3:0:b:16#0000020 PLC\_PRG.Q5:BOOL:2:4:0:b:16#0000020 PLC\_PRG.Q6:BOOL:2:5:0:b:16#0000020 PLC\_PRG.right\_dimention:BOOL:4:542:1:b:16#0000040 PLC\_PRG.scada\_rotation\_down:BOOL:4:546:1:b:16#0000040 PLC\_PRG.scada\_rotation\_up:BOOL:4:545:1:b:16#0000040 PLC\_PRG.scada\_start\_button:BOOL:4:543:1:b:16#0000040 PLC\_PRG.scada\_stop\_button:BOOL:4:544:1:b:16#0000040 PLC\_PRG.work\_motor:BOOL:2:0:0:b:16#0000020

13. Загрузите программу во внутреннюю память контроллера XC-CPU201. Во вкладке «Онлайн» (Online) откройте диалог «Параметры соединения» (Communication parameters) и нажмите кнопку «Создать» (New) для настройки нового соединения с типом TCP/IP. Далее присвойте ему осмысленное имя. Задайте IP-адрес в соответствии с картой IP адресов (рис. 3), в данном случае 192.168.119.23. Установите соединение с контроллером: «Онлайн/Соединение» (Online/Login). Далее подтвердите загрузку (download) кода проекта. Также во вкладке «Онлайн» (Online) выберите опцию «Создать загрузочный проект» (Create boot project).

## 2.4. Создание программы на языке программирования ST в среде CODESYS для контроллера НМІ панели визуализации XV102

1. Запустите программную среду CODESYS 2.3.9.

2. Создайте новый проект: «Файл/Создать» (File/New). ВНИ-МАНИЕ! путь к папке с файлом проекта и имя самого файла должны быть названы латинскими буквами.

3. Настройте целевую платформу (Target Settings). Откройте вкладку «Целевая платформа» (Target Settings) и на странице диалогового окна «Конфигурация» (Configuration) выберите тип контроллера **XV-1xx-V2.3.9 SP8**. Во вкладке сетевой функциональности (Network functionality) поставьте галочку на против «Support network variables» и в открывшемся поле ввода пропишите «CAN» (рис. 40). Подтвердите ввод нажатием кнопки OK.

Target Settings	×
Configuration: XV-1xx-V2.3.9 SP8 Target Platform Memory Layout General Network functionality Visualization Support parameter manager V Support network variables Names of supported network interfaces: CAN Example of a name list: CAN:UDP;DP;DEVNET max. 7 characters/name !	
Default OK Cance	

Рис. 40. Установка поддержки сетевой передачи данных через CAN Open протокол

4. Создайте главную программу. После настройки целевой платформы в автоматически появившемся диалоговом окне New POU выберите тип компонента «Программа» (Program), язык ST, задайте имя программы PLC\_PRG.

5. Настройке конфигурацию контроллера в НМІ. Зайдите во вкладку ресурсов «Resources» проекта и откройте конфигурацию контроллера «PLC Configuration» (рис. 41). Далее нажмите правой клавишей мыши по верхнему уровню меню (Configuration) и добавьте «CanMaster».

Resources	Configuration	Base parameters CAN parameters	
CAN_XV102		baud rate:	20000 💌
Global_Variable		Com. Cycle Period (µsec):	0
Variable Confic		Sync. Window Lenght (µsec):	0
ibrary 3S_CanDrv.li		Sync. COB-ID:	128 activate: 🔽
Emiliary 35_CANoper		Node-Id:	1
umi ibrary 35_CANOPE Ibrary CanUser.lib 2			Automatic startup
⊞… 🛄 library CanUser_Ma ⊕… 🚞 library SysLibFile.lib:			☑ Support DSP301,V <u>4</u> .01 and DSP306
		Heartbeat Master [ms]:	0
			_
Ibrary SysLibSocke			CAN Master Library is not stopped when on BP.
⊞···			
👘 Library Manager			
ELC Browser			
Sampling Trace			
	v .		
₩ Watch- and Recipe	< >		
	Loading library 'C:\Program Files (x86	6)\Common Files\CAA-Targets\E	aton Automation\V2.3.9 SP8\Lib_XV-1xx\3S_CA
	<		

Рис. 41. Конфигурация контроллера в НМІ

6. Объявите переменные передачи данных по CAN Open. Перейдите на вкладку «Pecypcы» (Resources), и далее в подраздел «Глобальные переменные» (Global\_Variables). Кликните правой клавишей мыши по разделу (папке) глобальных переменных добавьте новый объект с настройками как на рис. 42.

Properties	?	×
Global Variable List		
Name of the global variable list: CAN_XV102		
	-	$\sim$
Filename:	Add net	twork
Import before compile     C Export before compile		_/
Connection 1 (CAN)		
Network type	Remo netwo	ove
Pack variables		
List identifier (COB-ID):		
Transmit checksum		
C Acknowledgement		
Read Request on bootup		
Vrite Answer bootup requests		
Cyclic transmission Interval: T#50ms		
✓ Transmit on change Minimum gap: T#20ms		
Transmit on event Variable:		
ОК	Ca	incel

Рис. 42. Добавление нового объекта в глобальные переменные

Задайте переменные передачи данных по протоколу CAN Open от контроллера в HMI панель, так и от HMI панели в контроллер, как показано на рис. 43. ВНИМАНИЕ! Наименование CAN переменных и тип данных должны совпадать в проекте ПЛК, ПЛК панели и проекте визуализации.



Рис. 43. Переменные передачи САN Ореп данных

Главная программа во внутреннем контроллере HMI панели на языке ST показана на рис. 44.



Рис. 44. Главная программа во внутреннем контроллере НМІ панели

7. На панели оператора включите RS-сервер Start/Communication/RS-Server.

8. Загрузите программу во внутреннюю память контроллера НМІ панели XV102. Во вкладке «Онлайн» (Online) откройте диалог «Параметры соединения» (Communication parameters) и нажмите кнопку «Создать» (New) для настройки нового соединения с типом TCP/IP. Задайте IP-адрес 192.168.119.24. Установите соединение с контроллером: «Онлайн/Соединение» (Online/Login). Подтвердите загрузку (download) кода проекта. Во вкладке «Онлайн» (Online) выберите опцию «Создать загрузочный проект» (Create boot project).

## 2.5. Создание графического интерфейса управления в среде GALILEO для НМІ панели визуализации XV102

1. Откройте Galileo 8 и создайте новый проект Project/New.

2. Выберите тип панели Panel Type. В автоматически открывшемся окне выбрать Panel selection и после этого выбрать наименование панели оператора (рис. 45). Также после создания проекта, тип панели можно выбрать в Config/Panel Type.



Рис. 45. Выбор панели оператора

3. Выберите коммуникацию Config/Select Communication. В открывшемся окне нажмите кнопку Add, затем выберите Codesys Xsoft-CoDesys-2/MXpro (рис. 46).

Config Extras Build Window	Select PLC	×
Panel Type	Firm / Model   Info   PLC Data   Communication	1
Define <u>T</u> ext Export Texts Import Texts	No.         Port         Board         Model         Description           0         Ether         CoDe	
Parameter List Manager Help Manager	Add Remove Modify	Meta Data
Notes Settin <u>a</u> s Language	bree Star IP A Port: Ethernet	Ethernet
User Management Color Palette	Description:	OK Cancel
CE Configuration	OK Cancel	Help

Рис. 46. Выбор коммуникации

4. Добавьте теги. В дереве проекта выбрать вкладку Таg, нажать правой клавишей мыши на соответствующем типе данных и задать имя тэгу (рис. 47). ВНИМАНИЕ! Наименование САN переменных и тип данных должны совпадать в проекте ПЛК, ПЛК панели и проекте визуализации.

Tags		<del>▼</del> ‡ ×
🗖 Ma   🖅	Scri 🕸 Tags 🕙 Us   🖨 Pri	🖄 Gra   🔲 Pri   🛄 Rec
<filter></filter>		
E 🛞 bit	<u>N</u> ew Tag	
X	New Array	
×	Duplicate Ctrl+D	
	Rename F2	
<b>x</b>	2 Cut	-
X   9	Copy	
- X 4	Pasta	
x x	Palste	
	Delete	
wor	Expand tags Shift+Num +	
	Collapse tags Shift+Num -	
(f) floa	Properties	
ab strir		-
stru	Find tag	
⊕⊚ syst	Find/replace address	
	Find/move address	
	Move address	
<u>a</u>	Search Ctrl+F	
	Import	
		-

Рис. 47. Добавление тега

5. *Настройте тег*. Два раза кликните на созданном тэге. Перейдите на вкладку Address и в соответствующей строке нажмите клавишу выбора. После этого задайте параметр <prog> и <tag> (рис. 48).

🗖 M   🔄 Scr 🔧 Tag	Igs 🛞 Us   🔁 Pri   🖄 Gr   🗐 Pri   🛄 Re
<filter> ⊡·(X) bit</filter>	Tag-Settings
X rotation_dowr	n Address Limits
start_button	Read Write CoDeSys XSoft-CoDeSys-2/MXpro:
···· X stop_button	Invarie Type On Demand At Startup Polling [s] On Demand Enable M/S Address
× work_motor	start_button bit 🛛 🖾 fast 🔀 🖾 Master PLC_PRG.start_buttor
- by byte - w word - w dword - f float - e error - a string - s struct - sy system	Setting address       Image: Constant Setting address         Use tag/struct name as part of the address         prog       PLC_PRG         tag       start_button         PLC_PRG.start_button         Clear Address       OK         OK       Apply         Cancel       OK

Рис. 48. Задание параметров тега

Аналогичным образом, добавьте и настройте все остальные теги в соответствии со следующим списком необходимых тегов:

- rotation\_down
- rotation\_up
- start\_button
- stop\_button
- work\_motor

6. Добавьте маску. В дереве проекта выберите вкладку Mask, нажмите правой клавишей мыши и в диалоговом окне выберите New. Задайте имя маски (рис. 49).

Masks			🔺 û 🗙		General.xmsk*	x
🔲 Masks 🖹 Scri   🕸 Tag	s   🕙 User  🗎 Print	🖄 Gra   🔲 Print	. 🛄 Reci	<u> </u>		
<filter></filter>						
🖃 🔚 Masks (standard)	New	Ctrl+R				
General.xmsk	New Folder					
Masks (keyboard)	Open					
Masks (printer rep	Close					
Masks (sub)	Close All Masks					
	Save	Ctrl+S				
	Save Copy As	Curts				
100	Save Copy As					
4	Save All Masks					
	Rename	F2				
7	( Delete					
	Define As Start Mask					
	Expand All Folders	Shift+Num +				
	Collapse All Folders	Shift+Num -				
	Mask Info					
	Mask Settings					
	Project Masks					
	Print					
	Print Setup					
	Search Replace					
	Find					

Рис. 49. Добавление маски

7. Нарисуйте необходимые графические элементы (рис. 50). Для этого используйте такие типы элементов, как «Button» и «Flag Display» (рис. 51).



Рис. 50. Графические элементы управления на главной маске



Рис. 51. Инструменты рисования графических элементов

8. *Настройте кнопки управления*. Для этого двойным кликом левой клавишей мыши перейдите в настройки данного графического элемента. Пример настроек для кнопки «drain\_hot\_button» (DRAIN HOT) приведён на рис. 52, 53.

I Button			
General Size / Position Accessibility Text	1		
Tag: 🗙 start_button		<b>-</b>	
Address: PLC_PRG.start_button			
Style: Text	3D Frame		Click
$\sim$			
© SET Bit Stream			
P# 0			
	OK Apply	Cancel	Help

Рис. 52. Основные настройки кнопки «start\_button»

💷 Buttor	n											x
General	General   Size / Position   Accessibility Text											
	lo Tout			Font			Foregr	Foreground		Background		
	io. Text	Style	Size	H-Orient.	V-Orient.	B I <u>U</u> <del>S</del>	Color	Blink.	Color	Blink.	Transp.	
OFF 1	START	00: Aria	• 32 •	center 🝷	center 🝷		0		7			
ON 2		00: Aria	• 32 •	center -	center 🝷		0		7			
Please use the context menus (press right mouse button) of the different grid areas.												
					[	ОК		Apply		Cancel	Help	

Рис. 53. Настройки текста кнопки «start\_button»

Аналогичным образом настройте все остальные кнопки управления, считая, что кнопки должны привязываться к тегам, имена которых заканчиваются на «\_button».

9. *Настройте флаг отображения*. Для этого двойным кликом левой клавишей мыши перейдите в настройки данного графического элемента (красный прямоугольник на рис. 51). Пример настроек для флага «work\_motor» приведён на рис. 54, 55.

💷 Flag Display		×
General Size / Position Visibility Color		
Tag: X work_motor	▼ ▶	
Address: PLC_PRG.work_motor	-	
Style: Color	▼ 3D Frame ▼	
		,
States per Bit		
C States per Value		
No. of States:  2		
	OK Apply Cancel Hel	

Рис. 54. Основные настройки флага «work\_motor»

🗈 Flag Display 🛛 🗙
General Size / Position Visibility Color Background Bits OFF 23 Bit 0 ON 46 Please use the context menus (press right mouse button) of the different grid areas.
OK Apply Cancel Help

Рис. 55. Цветовые настройки флага «drain\_hot\_work»

10. Откомпилируйте проект перед загрузкой в НМІ панель Build/Compile.

11. На панели оператора включите FTP-сервер Start/Communication/FTP-Server.

12. Загрузите проект визуализации в панель оператора. Для этого выберите Build/Download и в открывшемся окне нажмите кнопку FTP Path. Затем New Connection. В открывшемся окне задайте имя, укажите IP адрес панели (в данном случае 192.168.119.24 в соответствии со схемой по рис. 3) и нажмите ОК (рис. 56). Далее подтвердите загрузку кнопкой «Download» (рис. 57). Для успешного проведения загрузки необходимо, чтобы путь к сохранненым файлам проекта не содержал русских букв.

Draw Obj	jects Config Extras Build Window Help							
BIAD								
Download								
Project Path: f:\work\_live_projects\konveyer\ec4p_hmi_tm\galileo_hmi\ Browse								
Local/FTP Pa	th: FTP: HMI   Local Path							
	Clear before download							
	Recipe Data							
	Password Data Memory							
1	Operating System and Components							
	Close							
FTP-Connec	tions STOP							
	New Connection							
	Edit							
	Сору							
	Properties: FTP-Connections							
	Server / IP.4ddraes - 192 168 119 24							
	Path :  \InternalStorage  Browse							
	OK Cancel							

Рис. 56. Настройка FTP соединения с НМІ панелью для загрузки проекта

Download		8 23
Project Path:	f:\work\_live_projects\konveyer\ec4p_hmi_tm\galileo_hmi\	Browse
Local/FTP Path:	FTP: HMI	Local Path
	Clear before download	FTP Path
	Password Data	Memory
	Source project as zip     Operating System and Components	Download
		Close

Рис. 57. Подтверждение FTP загрузки проекта в HMI панель

## 2.6. Создание графического SCADA интерфейса управления в среде TRACE MODE

В режиме исполнения проекта SCADA системы будет установлен двусторонний обмен данными с ОРС сервером. При этом сигналы будут поступать по следующим направлениям:

- Нажатие какой-либо SCADA кнопки ► передача сигнала («TRUE» в случае дискретного сигнала) в выходной канал – ► передача сигнала в выходной ОРС источник – ► воздействие сигнала на ОРС тег символьного файла – ► передача значения сигнала через тег символьного файла в контроллер – ► воздействие сигнала на переменную в контроллере.
- Изменение значения переменной в контроллере (например в «TRUE» в случае дискретного сигнала) ▶ передача значения переменной через тег символьного файла в ОРС сервер ▶ передача сигнала во входной ОРС приёмник ▶ передача сигнала во входной канал ▶ воздействие сигнала на графический элемент SCADA мнемосхемы (например светящуюся лампу).

1. Откройте Trace Mode 6 и создайте новый проект Файл/Новый (рис. 58).



Рис. 58. Создание нового проекта Trace Mode

Для начала обратите внимание на внешний вид мнемосхемы в режиме исполнения проекта, которая в дальнейшем должна получиться путём рисования графических элементов (рис. 59).

	MANAGER PANEL							
START STOP								
	ROTATION UP ROTATION DOWN							
	ALARMS							
8	Work motor							
•	Left dimention offset							
9	Right dimention offse	et						

Рис. 59. Мнемосхема с элементами управления

2. Добавьте каналы. Для этого нужно добавить как входные, так и выходные каналы с типами HEX16 и FLOAT (рис. 60, 61). Примеры натсроек каналов start\_button и work\_motor приведены на рис. 62, 63.

3. Добавьте ОРС группу и ОРС сервер в данной группе (рис. 64, 65).



Рис. 60. Добавление нового канала



Рис. 61. Список всех необходимых каналов

🛐 Экран#1 🝳	start_button		
Имя Комментарий	start_button	Кодировка	ТС2 Справка
Параметры Размерность I П Инверсия Вид предс	в битах 16 🗼		Системные Основные Тип Размерность Период Единица измерения 1 Цикл CALC • Включить Индекс Отработать На старте Отработань На старте Основация Дополнительно

Рис. 62. Пример настроек выходного канала start\_button

Экран#1         Фр start_button         Фр work_moto           У         С         И	
Имя work_motor Комментарий	Кодировка ТС2 Справка
Параметры Размерность в битах 16 💭 Пинверсия Вид представления DEC	Системные Основные Тип Размерность Период Единица измерения 1 Цикл CALC Автопосылка Включить Индекс Отработать На старте О Архивация Дополнительно

Рис. 63. Пример настроек входного канала work\_motor



Рис. 64. Добавление ОРС группы



Рис. 65. Добавление ОРС сервера

4. Добавьте ОРС источники и приёмники (рис. 66). Для этого нужно добавить дискретные источники/приёмники сигналов в ОРС сервер. На рис. 66 приведён полный список источнико/приёмников, для удобного восприятия их имена совпадают с именами ранее добавленных каналов (имя можно задать в окне редактирования после двойного клика клавишей мыши по источнику/приёмнику). К каждому источнику/приёмнику в дальнейшем будет привязан конкретный канал.



Рис. 66. Добавление ОРС источника/приёмника



Рис. 67. Полный список источников/приёмников ОРС сервера

Создавая нужный источник или приёмник, задайте ему следующие настройки:

- Имя;
- Сервер CoDeSys.OPC.02 (выбирается кнопкой «Обзор» с последующим выбором тега символьного файла, рис. 68);
- Идентификатор устанавливается автоматически после выбора тего символьного файла в ОРС сервере.
- Направление (Output или Input, ориентируясь на рис. 67).
- Формат дискрет.

📑 Экр	ан#1 💇 s	tart_button				2
	<ul> <li>ковные</li> <li>мя</li> <li>одировка</li> <li>омментарий</li> <li>араметры</li> </ul>	start_button TW0			Справка	
Ce	ервер СоDeS	vs.OPC.02			Обзор	? 💌
Bba  Vir	брать сервер ( мя Э Э Локальнь ОТ МОЕL ОТ Соре:	DPC ій компьютер LER,EASYOPC Sys.OPC.02 жружение	Комментарий EasyOpc Da OPC Server	Выбрать пе Имя 	ременные OPC 1:PLC_PRG.rotation 1:PLC_PRG.rotation 1:PLC_PRG.scada_ 1:PLC_PRG.scada_ 1:PLC_PRG.scada_ 1:PLC_PRG.scada_	_down _up rotation_down rotation_up start_button
					Готово	Отмена

Рис. 68. Выбор тега символьного файла в ОРС сервере

**ВНИМАНИЕ!** Имя ОРС тега в символьном файле, к которому привязывается источник или приёмник, может отличаться от имени самого источника или приёмника. Здесь главное понять к какому тегу нужно привязаться на уровне здравого смысла и понимания проекта.

Для удобства можно воспльзоваться таблицей 3 для привязки ОРС источников/приёмников SCADA системы к тегам символьного файла.

Таблица 3

Имя ОРС источника/приёмника в SCADA системе	Направление сигнала	Имя тега в символьном ОРС файле
start_button	→	scada_start_button
stop_button	→	scada_stop_button
rotation_up	→	scada_rotation_up
rotation_down	<b>→</b>	scada_rotation_down
work_motor	<del>\</del>	work_motor
left_dimention	←	left_dimention
right_dimention	<del>\</del>	right_dimention
velocity_bit1	<del>\</del>	Q4
velocity_bit2	<del>\</del>	Q5
velocity_bit3	<del>\</del>	Q6

Привязка ОРС источников/приёмников к тегам символьного файла

Пример настроек дискретного OPC источника «start\_button» приведён на рис. 69, а пример дискретного OPC приёмника «work\_motor» на рис. 70.

Экран#1 📿 st	art_button	k_motor			
CH 10					
Основные					
Имя	start_button				
Кодировка	TWO				Справ
Комментарий					
Параметры					
Сервер CoDeSy	s.OPC.02		Обзор		
CLSID {7904C3	02-AC19-11d4-9E1E-00	)105A4AB1C6}			
Идентификатор	PLC1:PLC_PRG.sca	la_start_button			
Режим		SYNC/CACHE		•	
Направление			Output	•	
Формат			Дискрет	•	
				-	

Рис. 69. Настройки OPC источника «start\_button»

Экран#1 🗛	start_button 📿	work_motor				
(° 19						
Основные						
Имя	work motor					
K	T)A/O					6
кодировка	TVVU					Справк
Комментарий					 	
Сервер СоDes CLSID (7904C	iys.OPC.02 302-AC19-11d4-9E1 PLC1:PLC_PRG:	E-00105A4AB1C6}	Обзор			
Режим		SYNC/CACHE		-		
Направление			Input	-		
Формат			Дискрет	•		

Рис. 70. Настройки OPC приёмника «work\_motor»
5. Откройте свойства каждого канала и сделайте его привязку к одноимённым источникам/приёмникам (рис. 71). Пример привязки выходного канала «start\_button» к ОРС источнику «start\_button» показан на рис. 72.



Рис. 71. Открытие свойств канала

<u>Φ</u> а	айл <u>П</u> роект <u>В</u> ид	д <u>О</u> кно (	Оправка							
1	🗈 퉦 • 🗐   🤇	) 🧖	f 🔆							
Har	вигатор проекта			×						
6	•X 🔊 @		🛛 🛛 🖣 🖓	👪   🐮 🏥 🔠 »						
<b>.</b>	ể Ресурсы		8	start_button 2						
-	🐕 Система	(	述 Задание пр	ивязки для объектastart	_button				2	3
	🖻 📓 RTM_1	_	😐 🐕 Систе	ма		ID	Полное имя	Короткое имя	Тип данных	ł
	1 📑 Кан	алы	🖃 🏪 Источ	ники/Приемники	<	0	Значение	VALUE		
-	Источники/	Приемник	i 👘 👕 oi	PC_1		78	Формат	FORMAT	USINT	
	🖃 🎬 OPC_1			-		79	Кодировка	CODE	STRING	
		Conner	- 💌	OPC_Cepsep_1	E	80	Комментарий	CMNT	STRING	-
		сервер_		🛆 start_button 🔰 👍		81	Ина		USINT	-
<b>.</b>	🇞 Библиотеки	_компоне				12/	SBVB		STRING	_
				Stop_button		129	ID	ID	STRING	-
				v rotation_up		130	MODE	MODE	UINT	
						131	SRVR CLSID	SRVR CLSID	STRING	
				rotation_down			_	_		
				🔮 work_motor						
				☑ left_dimention	-	•				F
L			Удалить привя	зку			6	Готово	Отмена	
Сис	стема.RTM_1.Кан	алы.start_l								
×	Информация	Флаги	Атрибуты							
	Имя:	start butto	n				Узе	п: Система.RTN	11	
utton	Кодировка:	TC2					Тип:	Канал_НЕХ10	- 6 Число ссыло	ж: 2
tartb	Комментарий:									
PI.S	Поивязка:	start butt	ор:Значение (Ист	очники/Приемники ОРС	1 OPC Censen				2	
Кана	Ducon:	cturt_outt				~				Y
M-1	DBI30B.									
Aa.BTI	Иконка:	<b>1</b> 6	96							<b>)</b> #
Cucren										

Рис. 72. Привязка канала к ОРС источнику

6. Нарисуйте графические элементы (рис. 59), используя инструменты рисования (рис. 73).

Рис. 73. Элементы графического рисования

7. Сделайте настройки и привязки графических элементов к каналам. Пример настроек и привязки кнопки «START» приведён на рис. 74, 75. Чтобы задать привязку, нужно кликнуть левой клавишей мыши по «Результат» и «Источник», далее откроется окно (рис. 75), в котором нужно добавить нувою строку в таблице для текущей привязки, задать тип, тип данных и открыть окно привязки (рис. 76). ВНИМАНИЕ! В таблице (рис. 77) целый ряд привязок для отдельных графических элементов. Для конкретного графического элемента

строка, после её добавления и **выбора**, будет подсвечиваться синим фоном. Т.е. это некая карта привязок графических элементов. **Обратите внимание** – в таблице в колонке привязки может быть как входное значение, так и реальное значение, в зависимости от направления сигналов.



Рис. 74. Настройки кнопки «START»

🔀 Привязка 🔹 😨									
6 X 8	77 I.	00	b 🗈 🛛	å		<b>.</b> %	R		
Имя	Тип	Тип данных	Значение	по умолча	анию Прив	язка			
start_button_In	TU0 <sup>1</sup>	 BOOL			⊂ <mark>e</mark> sta	art_button:Bxod	ное значение (Си	істема.RTM_1.Ка	налы)
stop_button_In	10UT	<mark>않힌</mark> BOOL				ор_button:Вход	ное значение (Си	ютема.RTM_1.Ка	налы)
rotation_up_In	100T	RE BOOL			⊂lerot	ation_up:Bxod	ное значение (Си	стема.RTM_1.Кан	налы)
rotation_down_ln	100T	- <b>設</b> BOOL			Grot	ation_down:Bx	одное значение (	Система.RTM_1.	Каналы)
work_motor_R	🛃 IN	BOOL			9 <sub>6</sub> wo	ork_motor Pear	њное значение (C	истема.RTM_1.К	аналы)
left_dimention_R	🛃 IN	BOOL				t_dimention:Pe	альное значение	(Система.RTM_1	.Каналы)
right_dimention_R	l 🛃 IN	- BOOL			9 <sub>6</sub> rig	ht_dimention:P	еальное значени	е (Система.RTM_	1.Каналы)
velocity_bit1_R	🛃 IN	<b>設</b> BOOL			9 <sub>6</sub> ve	locity_bit1:Pea	льное значение (	Система.RTM_1.	Каналы)
velocity_bit2_R	🛃 IN	<b>設</b> BOOL			9 <sub>6</sub> ve	locity_bit2:Pea	льное значение (	Система.RTM_1.	Каналы)
velocity_bit3_R	🛃 IN	- 副 BOOL				locity_bit3:Pea	льное значение (	Система.RTM_1.	Каналы)
•									Þ
👿 Использоват	ъ привяз	анный атрибут	Атри	бут -1					
Homep ALL	CALL	. HEX_16	HEX_32	FLOAT	FLOAT_M	FLOAT_64	M-RESOURCE	D-RESOURCE	USER 🔺
0 R	R	R	R	R	R	R	R	R	R 👻
							]		Þ
			Готов	80	Отмена	Отвязат	•		

Рис. 75. Создание привязки графического элемента к каналу

MMR.		
tart button Ir		start button:Входное значение (Система, RTM 1.Каналы)
top_butto otation_up	<sup>8</sup> Сконфигурировать связь дляШаблоны_экранс	рв.Экран#1:start_button_In
otation_dc vork_moto	□ 鼎 Система	Атрибуты Аргументы
ft_diment	😑 🙀 RTM_1	ID Полное имя ^
elocity bi	🖃 Е <mark>е</mark> Каналы	Е 1_Аппаратное значение
elocity_bi	G start_button	2 Входное значение
elocity_bi	e stop button	4 Достоверность
	16	5 Период пересчета (значение)
	₩ <sub>6</sub> rotation_up	6 Тенденция
		7 Интервал
	······································	
•		
- · · ·	Атрибуты/Аргументы Тип атрибута	

Рис. 76. Настройка привязки к каналу

На рис. 77-82 приведены примеры свойств различных графических элементов мнемосхемы.



Рис. 77. Свойства фигуры показания кода скорости

<u>Ф</u> айл <u>П</u> роект <u>П</u> р	авка <u>С</u> ервис <u>В</u> ид <u>О</u> кно <u>С</u> правка	l
🗄 🎦 💘 - 🔙 🛛 🍳	🤊 🕫 🔯 🐇 🛧 🍢   🔇	X 🗈 🖺 🗙 100% 💌 🕵
Навигатор проекта	×	🛐 Экран#1
🍅 - 🗙 🤊 🤊	🛅 🖺   💐 🗟 🚰   🚜 »	
🗄 🖻 Ресурсы	A	
	🗐 🔚 Каналы	N
Система	Экран#1:1	
🖃 🏭 RTM_1	*	
•	4	
Система.RTM_1.Экр	ран#1:1	STAR
Свойства объекта	×	
📀 Вык	лючатель 4 Справка	
		ROTATIO
<u> </u>	Копировать Вставить Заменить	
Свойство	Значение	
Привязка	<4> work_motor_R	
Вид индикации	Агд & Конст	
Инверсия	True	
Константа	0x1	
Код доступа	0	
Значение (XOR)	0x0	
* Видимость	True	
<u>* Подсказка</u>		Work me
<u>* Прозрачность</u>	0	
*Слой	Слой	
* Выделение в МРВ	False	📕 🕑 🛛 Left dime
*Геометрия	Скрыть	
		Right dim

Рис. 78. Свойства флажкового индикатора работы двигателя

<u>Ф</u> айл <u>П</u> роект <u>П</u> р	авка <u>С</u> ервис <u>В</u> ид <u>О</u> кно <u>С</u> правка				
🗄 🎦 📲 🛛 🔇	)   🏓 🧶 🔆 🔆 🖗   👔	χI	b 🗈 🗙	100% 🖵 🔍	<u> </u>
Навигатор проекта	×	46	Экран#1*		
🍅 - 🗙   🤊  ୯	🗅 🗈   🕂 🗟 🚰   🗛 »				
🗎 💼 Ресурсы	🔺 📑 Каналы	1			
🖃 🐕 Система				N	<b>NAN</b>
🖻 🙀 RTM_1	С Экран#1:1				
•					-
Система.RTM_1.Экр	ран#1:1			STAR	R I
Свойства объекта	×				
• Стандарт	ный видеоклип 2 Справка			ROTATIC	
	<u>₽</u>				
	Копировать Вставить Заменить				
Свойство	Значение				
Привязка	<5> left_dimention_R				VLL
Вид индикации	Arg & Конст				~
Константа	Ox1				
Прозрачный фон	False				
<u>Naysa</u>	0				
* Видимость	True				
* Подсказка					
<u>* Прозрачность</u>	0		<b>A</b>	Work m	otor
* Слой	Слой			WOR III	5101
* Выделение в МРВ	False				
<u>* Геометрия</u>	Скрыть			Left dime	ention
				Right din	nentio

Рис. 79. Свойства сигнализационной лампы левого габарита

🔛 Привязка 📀 💌								
🍅 - 🗙   🔞	<u> </u>	000	b 🗈 🛛	<b>i</b>		• 14	1	
Имя	Тип	Тип данных	Значе Пр	ивязка				*
start_button_In	10UT	800L	96	start_buttor	:Входное зна	вчение (Систе	ма.RTM_1.Канал	ы)
stop_button_In	100T	800L	96	stop_buttor	:Входное зна	ачение (Систе	ма.RTM_1.Канал	ы)
rotation_up_In	100T	👪 BOOL	96'	rotation_up	Входное зна	чение (Систел	ма.RTM_1.Каналь	5I)
rotation_down_lr	n 🟦OUT	800L	96'	rotation_do	wn:Входное з	значение (Сис	тема.RTM_1.Кан	алы) 🔤
work_motor_R	📥 IN	800L	96'	work_motor	:Реальное зн	начение (Сист	ема.RTM_1.Кана	лы) 📒
left_dimention_R	🛃 IN	🔡 BOOL	9 <sub>6</sub> 1	left_dimenti	on:Реальное	значение (Си	стема.RTM_1.Кан	налы)
right_dimention_	R 占 IN	800L	96'	right_dimen	tion:Peaльнo	е значение (С	истема. <u>RTM_</u> 1.Ка	аналы)
velocity_bit1_R	<mark>&gt;↓</mark> _IN	BOOL		velocity_bit	1:Реальное з	начение (Сис	тема.RTM_1.Кан	алы
velocity_bit2_R	🛃 IN	BOOL	96	velocity_bit	2:Реальное з	начение (Сис	тема.RTM_1.Кана	алы)
velocity_bit3_R	🛃 IN	800L	96	velocity_bit	3:Реальное з	начение (Сис	тема.RTM_1.Кана	алы) 🔫
•								P.
Использова	ть привяза	анный атрибут	Атри	бут -1				
Homep ALL	CALL	HEX_16	HEX_32	FLOAT	FLOAT_M	FLOAT_64	M-RESOURCE	D-RE 🔺
0 R	R	R	R	R	R	R	R	R 👻
								Þ
		Γοτα	ово	Отмена	Отвя	азать		

Рис. 80. Привязка фигуры показания кода скорости к каналу

t	🖞 Привязка								?	×
	🍅 <b>- 🗙</b>   📷	32 AA	00		) <b>M</b> [			• %	X	
	Имя	Тип	Тип данных	Знач	Привязка					*
	start_button_In	100T	800L		ຊ <sub>6</sub> start_bເ	tton:Входн	юе знач	нение (Систем	иа.RTM_1.Канали	51)
	stop_button_In	100T	RE BOOL		Gestop_bu	ntton:Входн	юе знач	ение (Систем	иа.RTM_1.Канали	5I)
	rotation_up_In	100T	RE BOOL		9 <sub>6</sub> rotation	_up:Входн	ое знач	ение (Систем	а.RTM_1.Каналь	a)
	rotation_down_In	1U0 💼	🔡 BOOL		9 <sub>6</sub> rotation	down:Bxg	дное зн	начение (Сист	<u>ема.RTM_1</u> .Кана	алы _
<	work_motor_R	<mark>} ↓</mark> ]N	800L	<	□ </th <th>otor:Реаль</th> <th>ное зна</th> <th>зчение (Систе</th> <th>ма.RTM_1.Кана</th> <th>пыр</th>	otor:Реаль	ное зна	зчение (Систе	ма.RTM_1.Кана	пыр
	left_dimention_R	🛃 IN	800L		eleft_dim	ention:rea	льное з	начение (СИС	тема.КТМ_1.Кан	алы
	right_dimention_R	l 🛃 IN	800L		9 <sub>6</sub> right_di	mention:Pe	альное	значение (Си	стема.RTM_1.Ка	нал
	velocity_bit1_R	占 IN	BOOL		e <sub>6</sub> velocity	_bit1:Pean	ьное зн	ачение (Сист	ема.RTM_1.Кана	лы
	velocity_bit2_R	占 IN	BOOL		e <sub>6</sub> velocity	_bit2:Pean	ьное зн	ачение (Сист	ема.RTM_1.Кана	лы
	velocity_bit3_R	M 🛃	BOOL		Gevelocity	_bit3:Pean	ьное зн	ачение (Сист	ема.RTM_1.Кана	лы 👻
	•									•
	🚺 Использоват	ъ привяз	анный атрибут		Атрибут -	1				
	Homep ALL	CALL	. HEX_16	HEX	_32 FLO	AT FLO	AT_M	FLOAT_64	M-RESOURCE	[ _
	0 R	R	R	R	R	R		R	R	R -
	٠									Þ.
			Готов	0	Отме	на	Отвяза	ать		

Рис. 81. Привязка флажкового индикатора работы двигателя к каналу

🕍 Привязка							?	x
🍅 - 🗙   🔞	33 AA	000	) in (	<b>#</b>		• 14	Ä	
Имя	Тип	Тип данных	Знач Пр	ривязка				
start_button_In	10UT	800L	۹e	start_button	:Входное знач	чение (Систем	иа.RTM_1.Каналы	51)
stop_button_In	10UT	NBOOL	e e	stop_button	:Входное знач	чение (Систем	иа.RTM_1.Каналы	51)
rotation_up_In	10UT	👪 BOOL	96	rotation_up:	Входное знач	ение (Систем	ia.RTM_1.Каналь	a)
rotation_down_In	10UT	🔡 BOOL	96	rotation_dov	wn:Bxoднoe зн	начение (Сист	гема.RTM_1.Кана	элы _
work_motor_R	🛃 IN	RE BOOL	96	work motor	: <mark>Реальное зн</mark>	ачение (Систе	<u>ма.RTM_1.Кан</u> ал	пы) =
[left_dimention_R	<mark>,↓</mark> 1N	800L	9	left_dimentio	on:Peaльноe s	значение (Сис	тема.RTM_1.Кан	аль
right_dimention_F	R 🛃 IN	800L	96	right_diment	іоп:геальное	значение (Си	стема. КТМ_1.Ка	нал
velocity_bit1_R	🛃 IN	800L	96	velocity_bit	1:Реальное зн	начение (Сист	ема.RTM_1.Кана	лы
velocity_bit2_R	🛃 IN	800L	96	velocity_bit	2:Реальное зн	начение (Сист	ема.RTM_1.Кана	лы
velocity_bit3_R	🛃 IN	🔡 BOOL	96	velocity_bit	3:Реальное зн	начение (Сист	ема.RTM_1.Кана	алы 👻
•								-F
Использоват	ть привяз	анный атрибут	Ат	рибут -1				
Homep ALL	CALL	. HEX_16	HEX_32	2 FLOAT	FLOAT_M	FLOAT_64	M-RESOURCE	[ <u>^</u> ]
0 R	R	R	R	R	R	R	R	R 🚽
								1
		Готов	0	Отмена	Отвяза	ать		

Рис. 82. Привязка лампы левого габарита к каналу

8. Проверьте работоспособность мнемосхемы в режиме эмуляции (рис. 83).

экт <u>П</u> равка <u>С</u> ери	вис <u>В</u> и,	д <u>О</u> кно <u>С</u> правк	а		~	
📕   🚯   🦧 🤌	🕈 🖄	۳ 🕫 😼	X 🗈 🖻 🗙	(   100	% 💽 🔍 🔍 🙀 🌉	<u>}</u>
роекта		×	💱 Экран#1*		$\sim$	
🤊 ୯ 🕒 🗈	N	🗟 🔚 🙀 »				
урсы		Каналы				
тема		Экран#1:1			MANAGE	RI
RTM 1	<b>⊵</b> rc	orpun#1.1				
Значения аргуме	нтов					
Имя	Тип	Значение			START	
start_button_In	BOOL f	false				
stop_button_In	BOOL f	false				
rotation_up_In	BOOL f	false			TATIONUUD	
rotation_down_In	BOOL f	false			TATION UP	
work_motor_R	BOOL f	alse				
left_dimention_R	BOOL f	false				
right_dimention_R	BOOL f	false				
velocity_bit1_R	BOOL f	false			VELOCITY	' BIT
velocity_bit2_R	BOOL f	false				
velocity_bit3_R	BOOL f	false				~
						$\bigcirc$
					4	

Рис. 83. Проверка мнемосхемы в режиме эмуляции

### 2.7. Настройка ОРС сервера

Чтобы ОРС сервер знал с какими контроллерами нужно взаимодействовать, необходимо добавить их в конфигурацию ОРС.

1. Запустите OPCConfig и добавьте контроллер (рис. 84).



Рис. 84. Добавление контроллера в ОРС конфигурацию

2. Задайте настройки контроллера (рис. 85).

Рис. 85. Настройки контроллера

3. Отредактируйте соединение с контроллером (рис. 86). Задайте его IP адрес, по которому ОРС сервер будет обращаться к контроллеру в соответствии с картой IP адресов (рис. 3), в данном случае 192.168.119.23.

🤹 OPCConfig - Multi-PLC Configuration	_		$\times$
File Edit ?			
Setver Setver Connection Settings for connection to PLC1 Edit Edit Edit Device: Tcp/lp (Level 2 Route) Parameter Address 1192.188.119.23 Port 1200 Targetid 0 Motorola byteorder No Communication Parameters Channels Tcp/lp (Level 2 Route) Name Value Comment Address 192.168.119.23 Paddress or hostname Pot 1200 Targetid 0 Name Value Comment Address 192.168.119.23 Paddress or hostname Pot 1200 Targetid 0 Motorola byteorder No		OK Cancel New Remove àateway Update	

Рис. 86. Редактирование соединения с контроллером

## 2.9. Апробация созданной системы управления

1. Создайте электрическое соединение лабораторной установки с контроллером. Соедините необходимые провода между датчиками физической модели и входами «І» контроллера (рис. 87). Также соедините необходимые провода между входами исполнительных элементов физической модели и выходами «Q» контроллера. Данные соединения должны производиться в соответствии с назначением переменных физических входов/выходов в редакторе объявлений PLC\_PRG. Объявленные переменные соответствуют определённым датчикам и исполнительным элементам физической модели.



Рис. 87. Подключение контроллера к модели конвейера

Соедините провод передачи данных CAN Open между контроллером XC-CPU201 и HMI панелью XV102 (рис. 88).



Рис. 88. Подключение провода CAN Open

2. Проверьте соединение с контроллером XC-CPU201. Во вкладке «Онлайн» (Online) откройте диалог «Параметры соединения» (Communication parameters) и удостоверьтесь в наличии настройки TCP/IP соединения с IP-адресом в соответствии с картой IP адресов (рис. 3), в данном случае 192.168.119.23. Установите соединение с контроллером: «Онлайн/Соединение» (Online/Login). CODESYS попросит вас подтвердить загрузку (download) кода проекта. Также во вкладке «Онлайн» (Online) выберите опцию «Создать загрузочный проект» (Create boot project).

4. Запустите контроллер XC-CPU201: «Онлайн/Запуск» (Online/Run).

5. Проверьте соединение с внутренним контроллером НМІ панели XV102. Во вкладке «Онлайн» (Online) откройте диалог «Параметры соединения» (Communication parameters) и удостоверьтесь в наличии настройки TCP/IP соединения с IP-адресом в соответствии с картой IP адресов (рис. 3), в данном случае 192.168.119.24. Установите соединение с контроллером: «Онлайн/Соединение» (Online/Login). CODESYS попросит вас подтвердить загрузку (download) кода проекта. Также во вкладке «Онлайн» (Online) выберите опцию «Создать загрузочный проект» (Create boot project). 6. Запустите внутренний контроллер НМІ панели XV102: «Онлайн/Запуск» (Online/Run).

7. Запустите профайлер среды SCADA TRACE MODE в режиме реального времени. Для этого сначала сохраните проект для монитора реального времени (рис. 89), а затем запустите профайлер (рис. 90).



Рис. 89. Сохранение для монитора реального времени



Рис. 90. Запуск профайлера TRACE MODE

В итоге запустится профайлер с работающим монитором реального времени (MPB), где можно просматривать состояние компонентов и полноценно управлять всей системой конвейера, нажимая на кнопки, задавая нужную скорость вращения двигателя (рис. 91).



Рис. 91. Работающий монитор реального времени (МРВ)

8. Проверьте работоспособность системы. С кнопок №3-4 стенда задайте необходимую частоту вращения двигателя привода ленты конвейера. С помощью кнопки №1 на стенде запустите двигатель. Обратите внимание, что при низкой частоте вращения двигателя, в программе управледолжна сработать автоматическая остановка двигателя (это ния блокировка от пониженной частоты вращения). Поставьте транспортируемые грузы разной высоты на движущуюся ленту конвейера, грузы должны от сортироваться по разным корзинам. Расположение груза за пределами контроля габаритов ленты конвейера должно привести к остановке двигателя.

Результат всех выполняемых манипуляций должен отображаться на мнемосхеме в SCADA системе TRACE MODE (рис. 59).

Дополнительно проверьте передачу значений тегов (переменных) через OPC сервер при включении тех или иных механизмов конвейера. Для этого запустите OPC обозреватель (MatrikonOPC Explorer) и установите соединение с OPC сервером CoDeSys.OPC.02 (рис. 92).



Рис. 92. Соединение с ОРС сервером через MatrikonOPC Explorer

Добавьте нужные теги для отображения (рис. 93, 94). Далее можно просматривать состояние добавленных тегов в режиме реального времени (рис. 95).



Рис. 93. Вход в раздел добавления тегов



Рис. 94. Добавление тегов в обозреватель

🥸 MatrikonOPC Explorer - [Untitled*]									
File Server Group Item View Help									
£ 💥 🖆   💣 🔗 📐 🔒 📝 ♥   🎂 🛥 🛶 🌳 🖆									
Group_15	Contents of 'Group_15'								
Localhost '\\ENTERPRISE-PC'	Item ID	Access Path	Value	Quality					
CoDesys.OPC.02	BPLC1:PLC_PRG.Q4		True	Good, non-specific					
	BPLC1:PLC_PRG.Q5		True	Good, non-specific					
Metwork Neighborhood	BPLC1:PLC_PRG.Q6		True	Good, non-specific					
Other Network Computers	BLC1:PLC_PRG.scada_rotati		False	Good, non-specific					
	BPLC1:PLC_PRG.scada_rotati		False	Good, non-specific					
	BPLC1:PLC_PRG.scada_start		False	Good, non-specific					
	PLC1:PLC_PRG.scada_stop		False	Good, non-specific					
	BLC1:PLC_PRG.work_motor		False	Good, non-specific					
				•					
Server Info	4000		Group	Info					
		Group: G	Group_15						
Server: CoDeSys.OPC.02		Connect	ed (Async I/	0): Yes (2.0)					
Connected: Yes		connect		<b>51</b> . 103 (2.0)					
State: Running		Active: \	Yes						
Groups: 14 Total Items: 8		Items: 8	Jundate Date	1000 ms					
Current Local Time: 11.16.2016 5:15:	11.185	Percent	Deadband: 0,	.00%					
Update Local Time: 11.16.2016 5:15:10.5761 Bandwidth Usage: 54									

Рис. 95. Отображение состояния тегов в режиме реального времени

## 3. УСТАНОВКА ПОГРАММНОГО ОБЕСПЕЧЕНИЯ

Внимание! В аудитории учебного центра на стендах данную процедуру выполняет только преподаватель один раз. Обучающийся, при наличии данного программного обеспечения, может его установить только на своём личном компьютере.

### 3.1. Установка и настройка CODESYS 2.3.9 SP8

1. Запустите файл Setup\_XSOFT\_CODESYS\_V2.3.9\_SP8.exe.

2. Если автоматически не запустился установщик таргетов Setup\_XC\_XV\_Targets\_V2.3.9\_SP8.exe, запустите его.

3. Активируйте таргеты для контроллера XC-CPU201 и НМІ панели оператора XV102. Для этого откройте специальную CODESYS утилиту InstallTarget и следуйте следующим шагам (рис. 96-98).

😰 InstallTarget	$\times$
Installation directory:	
Possible Tar     Choose installation directory     X       Directory:     0K     Ition V2.3.9 SP8	_
files\caa-targets\eaton automation\v2.3.9 sp8       Cancel	
Сеть	
Close	

Рис. 96. Определение области видимости выбора таргет-файла

🐞 Inst	tallTarget			×
	Installation directory: c:\program files (x86)\commo	]		
Po:	ssible Targets:	Installed Targets:		
	Open	E aton Automation	n V2.3.	3 SP8
	🚳 Открыть		×	
	Папка: 🚺 V2.3.9 SP8 💌	← 🗈 💣 📰▼		
	Имя	Дата изменения	^	
	XVS-4xx	28.06.2024 0:05		
	Eaton Automation.tnf	28.06.2024 0:06		
	EC4P.tnf	27.06.2019 9:18		
	🖕 XC-101.tnf	27.06.2019 9:18		
	121 tnf	27.06.2019 9:18		
	∠ 1 XC-201.tnf	27.06.2019 9:18		
	XN-PLC-CANopen.tnf	27.06.2019 9:18		
	XVC-6xx.tnf	27.06.2019 9:18	×	Close
	<	>	· -	0.000
-	Имя файла: XC-201.tnf	Открыть		
	Тип файлов: Target Information File (*.TNF)	• Отмена		

Рис. 97. Выбор TNF таргет-файла для установки таргетов



Рис. 98. Установка таргетов для контроллеров

#### 3.2. Установка Trace Mode 6.09

1. Установка программного комплекса Trace Mode 9.09 достаточно проста. Запустите установщик «Setup» и следуйте его дальнейшим инструкциям.

#### 3.3. Установка и лицензирование Galileo 8.1

1. Получите лицензионный ключ для использования его при установке Galileo. Для выполнения этой операции необходимо иметь лицензионный сертификат, который необходимо ввести на сайте http://www.eaton-automation.com/license, вместе с личными данными (рис. 99). В ответном письме на указанную электронную почту будет выслан лицензионный ключ.

2. *Установите Galileo* 8.1.10. Запустите файл GalileoV8110.exe и в ходе установки введите лицензионный ключ.



Рис. 99. Процесс получения лицензионного ключа

### 3.4. Обновление прошивки НМІ панели XV102

1. Подайте электрическое питание на панель оператора. После загрузки Windows CE, нажмите кнопку *Start* и выберите в меню *Programs/Control Panel* (рис. 100).

My Device	
Bograns Communication	
☐ geoments     Image Contracts Prompt       Ø sectings     Image Contract Same       Ø secting	Ello Yerw ? X
	Keyboard Leonse Network Owner
	Storagetta Santam Touch

Рис. 100. Открывание Control/Panel

- 2. Два раза кликните по вкладке Network.
- 3. В появившемся окне два раза кликните по вкладке *ONBOARD1* (рис. 101).

Connection	? ×
Make New Connection	
≹Stat D.C	14:37

Рис. 101. Открывание Onboard1

4. При этом откроется окно *FEC Internet Driver*, в котором будут отображены сетевые настройки контроллера панели оператора (рис. 102).

Conr FEC Et	thernet Driver'	OK >	
P 0	NS WINS		-
Come Come	tan an IP addres	s via DHCP	-
1P addre	HSS: 192	168 119 24	
Subnet	masic (255)		
Gatewa			11

Рис. 102. Сетевые настройки контроллера панели оператора

- 5. Задайте IP адрес 192.168.119.24 и нажмите «ОК».
- 6. Откройте Codesys 2.3.9. При выборе целевой платформы, необходимо указать XV-1xx-V2.3.9 SP8.
- 7. Перейдите во вкладку ресурсов, зайдите в конфигурацию ПЛК, выберите Firmware и нажмите кнопку Start (рис. 103).

🔩 XSOFT-CODESYS-2 - (Untitled)* - [PLC C	Configuration]		
💷 File Edit Project Insert Extras O	nline Window	Help	
12 <b></b>			
Resources Global Variables Ibrary SysLibFile.lib: global variables Ibrary SysLibPlcCtrl.lib: global variables Ibrary SysLibRtc.lib: global variables Ibrary SysLibTasks.lib: global variable Ibrary SysLibTasks.lib: global variable Ibrary Util.lib 20.3.15 10:30:08: globa Alarm configuration Library Manager Log PLC - Browser PLC configuration Target Settings Task configuration Vatch- and Recipe Manager Vorkspace PD I Da Vis. I Re	< Configu	ration	Settings Firmware Update operating system Start.

Рис. 103. Запуск прошивки НМІ панели

 В появившемся окне выберите файл TargetFirmwareWinCE\_V2.4.21, расположенный по адресу Program Files\Common Files\CAA-Targets\Eaton Automation\V2.3.9 SP8\Firmware\XV-1xx.

🎭 Открытие				×
← → ×  📙 « Eaton A	utomation > V2.3.9 SP8 > Firmware > XV-1xx	√ Ō	Поиск: XV-1xx	<i>م</i>
Упорядочить 🔻 Создать п	апку			
🔮 Документы	🖈 ^ Имя ^ Дата изме	енения	Тип	Размер
📰 Изображения 🛖 Новый том (E:)	TargetFirmwareWinCE_V2.4.21.exe 04.05.2022	2 14:57	Приложение	56 005 KE
\delta Мой диск	*			
🛆 Google Drive (G:)	*			
🗸 💻 Этот компьютер				
> 📑 Видео				
> 🔮 Документы				
🔉 🖊 Загрузки				
> 📰 Изображения				
> 🎝 Музыка	v <			>
Имя файл		~	Firmware for WinCE	$\sim$
			Открыть 🔻	Отмена

Рис. 104. Выбор TargetFirmwareWinCE\_V2.4.21

9. После этого запустится окно инсталлятора. В первом приветственном окне, мы нажимаем кнопку Next. В следующем выбираем режим FTP-Installation и нажимаем кнопку Next (рис. 105).



Рис. 105. FTP установка

10. В следующем окне введите IP-адрес панели оператора (192.168.119.24), остальные поля оставьте неизменным (рис. 106).

记 Setup - Tar	getFirmwareWinCE		- = x
FTP Parame What are t	t <b>ers</b> the FTP parameters?		•
Please spe IP Address 192,168,1 Username: anonymou	cify the login information and 119.24	click "Next" to continue.	
Password: guest			
www.eato	n.eu	< Back Next >	> Cancel

Рис. 106. Задание IP адреса для FTP установки

- 11. Перед выполнением следующего шага необходимо запустить FTP сервер на панели оператора, Start/Programs/Communication/FTP server/OK.
- 12. Если IP-адрес был указан верно и в панели оператора запущен FTP-сервер, то инсталлятор автоматически определит тип панели (рис. 107). Далее нажмите кнопку Next.

Setup - TargetFirmwareWinCE	
Target Type Select Target Type	
Please specify the Target Type, then dick "Next".	
( XV-lox	
English	lext > Cancel

Рис. 107. Автоматическое определение типа НМІ панели

13. Далее происходит выбор устанавливаемых компонентов. выберите все компоненты и нажмите кнопку *Next* (рис. 108).

Select components	
which components should be installed?	C.
Select the components you want to install.	
Windows CE 5.0 Core	OS 2.26.7 (4028)
Boot behaviour	
Jautoexec.bat	
PLC Runtime System	2.4.13 (2008)
🚛 🗹 Webserver	2.4.13 (2008)
L	
alish	

Рис. 108. Выбор устанавливаемых компонентов

14. Завершающим этапом подтвердите установку выбранных компонентов нажатием кнопки Install. Дождитесь завершения установки, панель оператора должна автоматически переза-грузиться.

# СПИСОК ЛИТЕРАТУРЫ

1. Гирник А.С., Федянин А.Л., Шилин А.А. Системы автоматики и управления на базе программируемых логических контроллеров: учебное пособие [Электронный ресурс] / А.С. Гирник. – Томск: Изд-во ТПУ, 2024. – Заглавие с титульного экрана. – URL: <u>https://portal.tpu.ru/departments/otdel/publish/catalog/2024/method\_2024/Tab/GirnikFedyaninShilin.pdf</u>.

2. Руководство пользователя по программированию ПЛК в CoDeSys 2.3. 3S – Smart Software Solutions GmbH 2008. – URL: https://owen.ru/uploads/373/cds23\_manual\_v2.8.pdf.

3. Первые шаги с CoDeSys. 3S – Smart Software Solutions GmbH 2004. – URL: https://owen.ru/uploads/373/cds23\_firststeps.pdf.

4. Визуализация CoDeSys. Дополнение к руководству пользователя по программированию ПЛК CoDeSys 2.3. 3S – Smart Software Solutions GmbH 2008. – URL: https://owen.ru/uploads/373/cds23\_visu\_v1.7.pdf.