

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение
высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

Направление 09.03.01 «Информатика и вычислительная техника»
подготовки:
Профиль подготовки: «Вычислительные машины, комплексы, системы и сети»

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
БАКАЛАВРА

на тему

Визуальная имитационная модель
производственной системы на основе
функциональных блоков

Студент

Кукушкин
(подпись, дата)

Кукушкин А. Ю.

Руководитель

Дубинин
(подпись, дата)

Дубинин В. Н.

Нормоконтролёр

Кучин
(подпись, дата)

А.В. Кучин

Работа допущена к защите (протокол зас. кафедры от 03.06.2020 г. № 12)

Заведующий кафедрой

Митрохин
(подпись)

М.А. Митрохин

Работа защищена с отметкой
23.06.2020 № 22

хорошо (протокол заседания ГЭК от

Секретарь ГЭК

Гурин
(подпись)

Е.И. Гурин

Пенза 2020

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

вычислительной техники

Кафедра

«Вычислительная техника»

«Утверждаю»

Заведующий кафедрой

 М.А. Митрохин

«30» апреля 2020 г.

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ БАКАЛАВРА

1. Студент Кукушкин Антон Юрьевич гр.16ВВ1 факультета вычислительной техники.
2. Тема работы: «Визуальная имитационная модель производственной системы на основе функциональных блоков»

Тема утверждена приказом ПГУ №0335а/о от "30" апреля 2020 г.
3. Руководитель бакалаврской работы д.т.н., доцент каф. ВТ ПГУ Дубинин В. Н.
4. Задание на работу (назначение разработки, исходные данные и т.п.)
ПС FESTO предназначена для изготовления некоторых законченных деталей из поступающих на ее вход заготовок. Она состоит из трех станций: распределительной (Distribution Station), тестирующей (Testing Station) и обрабатывающей (Processing Station). Необходимо разработать визуальную модель тестирующей и обрабатывающей станций.
5. Перечень подлежащих разработке вопросов
Анализ требований к разрабатываемой системе, разработка и реализация управляющих алгоритмов для тестирующей станции, разработка и реализация управляющих алгоритмов для обрабатывающей станции, отладка и тестирование системы ФБ тестирующей станции, отладка и тестирование системы ФБ обрабатывающей станции, разработка визуальной модели.
6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ требований к разрабатываемой системе	24	30.04.2020 – 07.05.2020	
2.	Разработка и реализация управляющих алгоритмов для тестирующей станции	12	08.05.2020 – 11.05.2020	
3.	Разработка и реализация управляющих алгоритмов для обрабатывающей станции	10	12.05.2020 – 14.05.2020	
4.	Отладка и тестирование системы ФБ тестирующей станции	12	15.05.2020 – 18.05.2020	
5.	Отладка и тестирование системы ФБ обрабатывающей станции	15	19.05.2020 – 23.05.2020	
6.	Разработка визуальной модели ТПВУ	6	24.05.2020 – 25.05.2020	
7.	Выполнение задания по БЖД	6	26.05.2020 – 27.05.2020	
8.	Оформление пояснительной записки	15	28.05.2020 - 01.06.2020	

Дата выдачи задания "30" апреля 2020 г.

Руководитель бакалаврской работы  Дубинин В. Н.
(подпись, дата)

Задание к исполнению принял студент  Кукушкин А. Ю.
(подпись, дата)

Работу к защите допустить
Декан факультета  Фионова Л.Р.
(подпись, дата)

Реферат

Работа содержит 108 листов, 29 рисунков, 13 таблиц, 19 источников, 6 приложений.

FESTO, MVS-ШАБЛОН, САТ-БЛОК, БАЗИСНЫЙ БЛОК, ТЕСТИРУЮЩАЯ СТАНЦИЯ, ОБРАБАТЫВАЮЩАЯ СТАНЦИЯ

Цель ВКР - разработать визуальную модель тестирующей и обрабатывающей станций.

ПС FESTO предназначена для изготовления некоторых законченных деталей из поступающих на ее вход заготовок. Она состоит из трех станций: распределительной (Distribution Station), тестирующей (Testing Station) и обрабатывающей (Processing Station).

В ходе выполнения ВКР были разработаны, протестированы и отлажены тестирующая и обрабатывающая станции ПС FESTO.

Данное устройство позволяет выполнить обработку детали, поступившей от внешнего производителя. Обработанная деталь может быть передана внешнему потребителю для дальнейшей эксплуатации.

					ПГУ 2.09.03.01.23.001 ПЗ				
Изм.	Лист	№ докум.	№	Подпись	Дата				
Разраб.		Кукушкин А. Ю.				Лит.		Лист	Листов
Провер.		Дубинин В. Н.						2	100
Н. Контр.		Кучин А.В.				ФВТ, каф. ВТ гр. 16ВВ1			
..									
Утверд.		Митрохин М.А.				Визуальная имитационная модель производственной системы на основе функциональных блоков Пояснительная записка			

Содержание

Введение	5
1 Описание структуры и алгоритмов функционирования	7
1.1 Обзор станций производственной системы	8
1.1.1 Распределительная станция	8
1.1.2 Тестирующая станция	9
1.1.3 Обработывающая станция	10
2 Описание MVC- шаблона проектирования	12
2.1 Общее описание	12
2.2 Модель	12
2.2 Вид	13
2.2 Контроллер	14
3 Разработка тестирующей станции	14
3.1 Разработка базисного блока	14
3.1.1 Разработка интерфейса базисного блока	16
3.1.2 Разработка алгоритма работы базисного блока	19
3.1.3 Тестирование и отладка базисного блока	20
3.2 Разработка составных и САТ-блоков тестирующей станции	20
3.2.1 Разработка САТ-блоков	21
3.2.2 Блок Closer – заслонка	22
3.2.3 Блок Lift	23
3.2.4 Блок Power Button	23
3.2.5 Блок Sensor	24
3.2.6 Блок Throw	25
3.2.7 Блок Controller	26
3.3 Разработка приложения	27
3.4 Разработка визуальной модели	30
3.5 Визуализация работы	31
3.6 Запуск приложения в среде NXTStudio 2.0	34

3.7 Тестирование приложения	37
4 Разработка обрабатывающей станции	37
4.1 Разработка базисного блока	37
4.1.1 Разработка интерфейса базисного блока	39
4.1.2 Разработка алгоритма работы базисного блока	42
4.1.3 Тестирование и отладка базисного блока	45
4.2 Разработка составных и САТ-блоков тестирующей станции	45
4.2.1 Разработка САТ-блоков	45
4.2.2 Блок Test	47
4.2.3 Блок Spin	48
4.2.4 Блок Power Button	49
4.2.5 Блок Sensor test	50
4.2.6 Блок Sensor place	51
4.2.7 Блок Grab	52
4.2.7 Блок Drop	53
4.2.7 Блок Drel	54
4.2.7 Блок Controller	55
4.3 Разработка приложения	56
4.4 Разработка визуальной модели	57
4.5 Визуализация работы	60
4.6 Запуск приложения в среде NXTStudio 2.0	61
4.7 Тестирование приложения	66
5 Освещение места работы инженера-разработчика	69
Заключение	73
Список использованных источников	74
Приложение А – Презентация	75

Введение

Техпроцесс обработки деталей является одним из определяющих факторов при разработке системы управления. Ниже приводится краткое описание техпроцесса, положенного в основу создания системы управления FESTO.

Внешний производитель помещает детали в магазин распределительной станции, где они могут накапливаться. Буферирование деталей в магазине способствует сглаживанию темпов работы внешнего производителя и ПС FESTO. При помощи специального выталкивателя (поршня) деталь выталкивается из дна магазина и сдвигается в выходное место магазина, откуда шарнирный рычаг перемещает ее на тестирующую станцию. В данном случае под выходным местом магазина понимается площадка, на которую из магазина выталкивается деталь. Выходное место магазина не является исполнительным магазином и с ним не связаны никакие датчики. Удерживание детали на шарнирном рычаге осуществляет расположенная на его конце присоска. Для присасывания детали включается вакуум, для отсоединения детали вакуум отключается.

При поступлении детали на тестирующую станцию (а именно, на тестирующее место) датчики-детекторы определяют цвет и материал, из которого изготовлена деталь. При ненормальном цвете или материале деталь выталкивается (отбраковывается) из тестирующей станции. Если цвет и материал детали находятся в норме, то деталь поднимается лифтом к устройству измерения высоты детали (измерителю высоты). Если в результате измерения оказывается, что высота детали нормальная, то деталь выталкивается в скользящий желоб, по которому она поступает на обрабатывающую станцию. В противном случае (ненормальная высота) деталь опускается на лифте вниз, где она выталкивается из тестирующей станции и таким образом исключается из дальнейшей обработки. Скользящий желоб имеет заслонку, которая позволяет заблокировать поступление детали на обрабатывающую станцию.

На обрабатывающую станцию деталь поступает в лунку рабочего стола, имеющего форму круга. Рабочий стол поворачивается вокруг своей оси на одну

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		5

позицию по часовой стрелке, перенося тем самым деталь в позицию для сверления. Сверление осуществляется с помощью дрели, способной двигаться по направляющим вверх и вниз. Перед сверлением деталь прижимается специальным прижимным стержнем, входящим в боковую прорезь лунки. После окончания сверления рабочий стол поворачивается еще на одну позицию по часовой стрелке и деталь оказывается в позиции для проверки отверстия. Проверка осуществляется с помощью специального проверочного стержня, опускаемого вниз по центру детали. Если отверстие нормального качества, то стержень проходит сквозь него и касается специального датчика, что свидетельствует о норме. После окончания проверки отверстия рабочий стол поворачивается на одну позицию по часовой стрелке, перенося деталь в позицию для выгрузки детали из обрабатывающей станции. Выгружаемая деталь передается внешнему потребителю деталей.

В своей выпускной квалификационной работе мне необходимо визуализировать тестирующую и обрабатывающие станции установки на базе функциональных блоков IEC 61499 в среде NXTCtrl.

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		6

1 Описание структуры и алгоритмов функционирования

Техпроцесс обработки деталей является одним из определяющих факторов при разработке системы управления. Ниже приводится краткое описание техпроцесса, положенного в основу создания системы управления FESTO.

ПС FESTO предназначена для изготовления некоторых законченных деталей из поступающих на ее вход заготовок. Она состоит из трех станций: распределительной (Distribution Station), тестирующей (Testing Station) и обрабатывающей (Processing Station). Заготовка представляет собой шайбу из пластмассы или металла. В процессе изготовления детали заготовка последовательно проходит через все три станции. Распределительная станция служит для буферирования поступающих от внешнего производителя заготовок, тестирующая станция – для контроля и отбраковки нестандартных заготовок, а обрабатывающая станция – для сверления в заготовке отверстия. Обработанная деталь может быть передана внешнему потребителю готовых деталей. В дальнейшем для краткости заготовку будут называть деталью (workpiece).

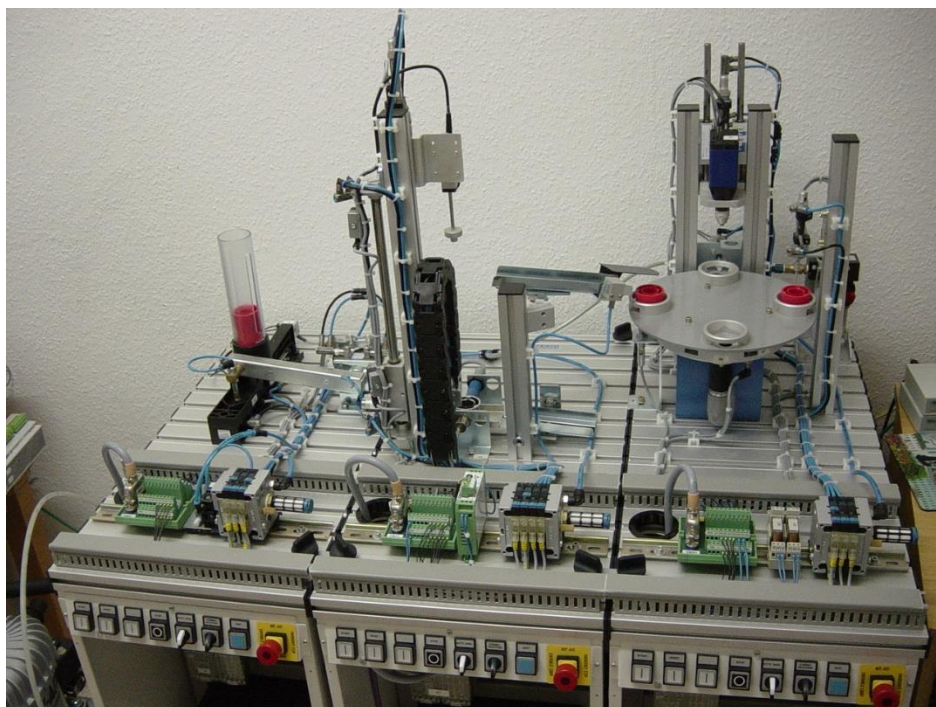


Рисунок 1.1 - Общий вид производственной системы FESTO

1.1 Обзор станций производственной системы

1.1.1 Распределительная станция

Внешний производитель помещает детали в магазин распределительной станции, где они могут накапливаться. Буферирование деталей в магазине способствует сглаживанию темпов работы внешнего производителя и ПС FESTO. При помощи специального выталкивателя (поршня) деталь выталкивается из дна магазина и сдвигается в выходное место магазина, откуда шарнирный рычаг перемещает ее на тестирующую станцию. В данном случае под выходным местом магазина понимается площадка, на которую из магазина выталкивается деталь. Выходное место магазина не является исполнительным магазином и с ним не связаны никакие датчики. Удерживание детали на шарнирном рычаге осуществляет расположенная на его конце присоска. Для присасывания детали включается вакуум, для отсоединения детали вакуум отключается.

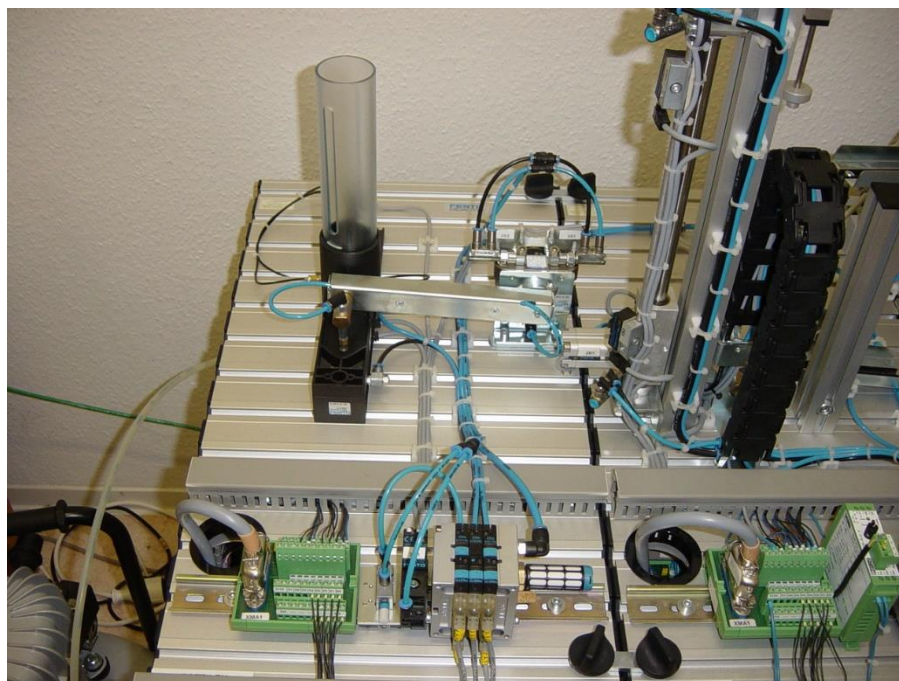


Рисунок 1.2 – Распределительная станция

1.1.2 Тестирующая станция

При поступлении детали на тестирующую станцию (а именно, на тестирующее место) датчики-детекторы определяют цвет и материал, из которого изготовлена деталь. При ненормальном цвете или материале деталь выталкивается (отбраковывается) из тестирующей станции. Если цвет и материал детали находятся в норме, то деталь поднимается лифтом к устройству измерения высоты детали (измерителю высоты). Если в результате измерения оказывается, что высота детали нормальная, то деталь выталкивается в скользящий желоб, по которому она поступает на обрабатывающую станцию. В противном случае (ненормальная высота) деталь опускается на лифте вниз, где она выталкивается из тестирующей станции и таким образом исключается из дальнейшей обработки. Скользящий желоб имеет заслонку, которая позволяет заблокировать поступление детали на обрабатывающую станцию.

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		9

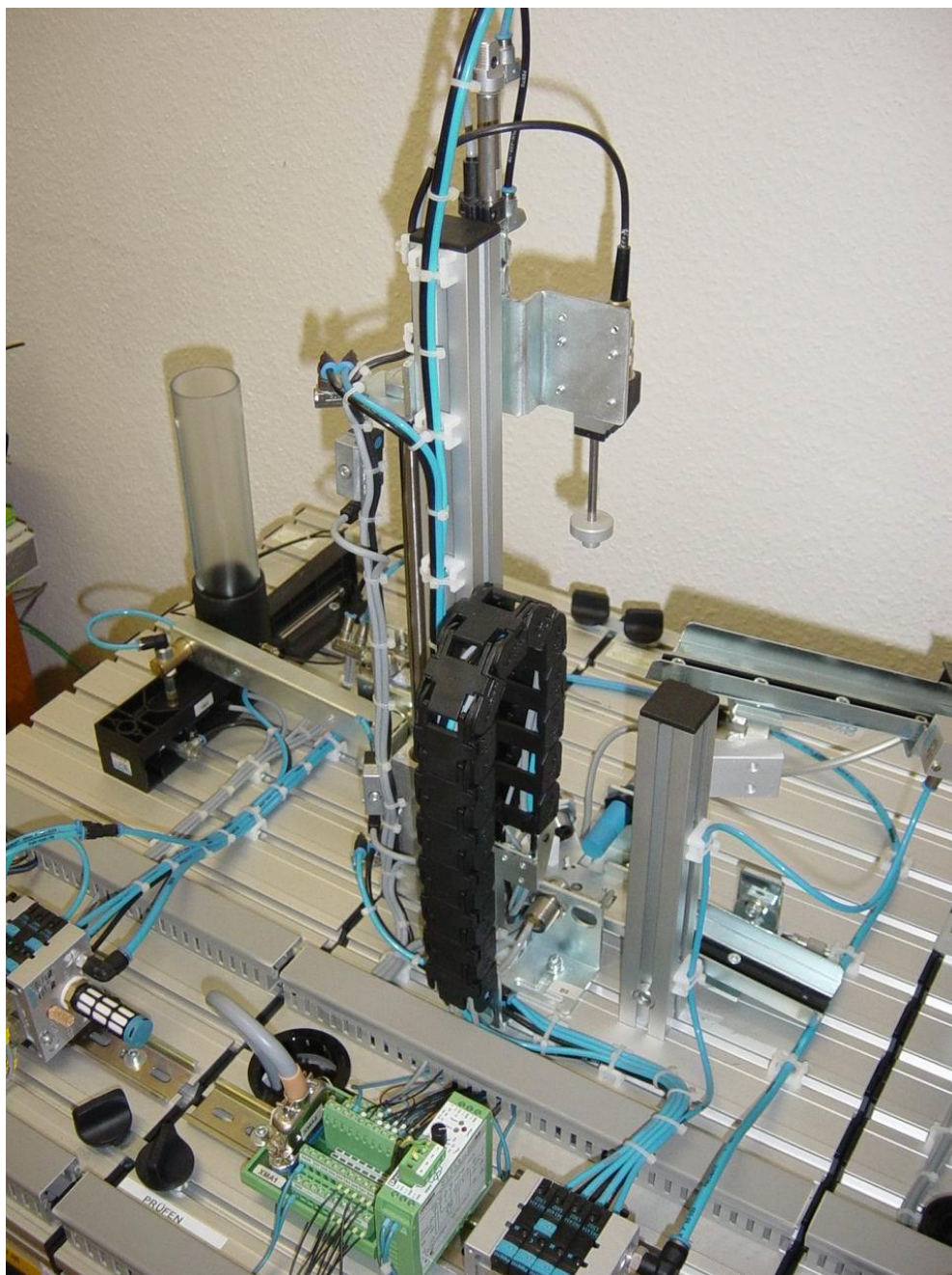


Рисунок 1.3 – Тестирующая станция

1.1.3 Обработка станция

На обрабатывающую станцию деталь поступает в лунку рабочего стола, имеющего форму круга. Рабочий стол поворачивается вокруг своей оси на одну позицию по часовой стрелке, перенося тем самым деталь в позицию для сверления. Сверление осуществляется с помощью дрели, способной двигаться по направляющим вверх и вниз. Перед сверлением деталь прижимается специальным

прижимным стержнем, входящим в боковую прорезь лунки. После окончания сверления рабочий стол поворачивается еще на одну позицию по часовой стрелке и деталь оказывается в позиции для проверки отверстия. Проверка осуществляется с помощью специального проверочного стержня, опускаемого вниз по центру детали. Если отверстие нормального качества, то стержень проходит сквозь него и касается специального датчика, что свидетельствует о норме. После окончания проверки отверстия рабочий стол поворачивается на одну позицию по часовой стрелке, перенося деталь в позицию для выгрузки детали из обрабатывающей станции. Выгружаемая деталь передается внешнему потребителю деталей.

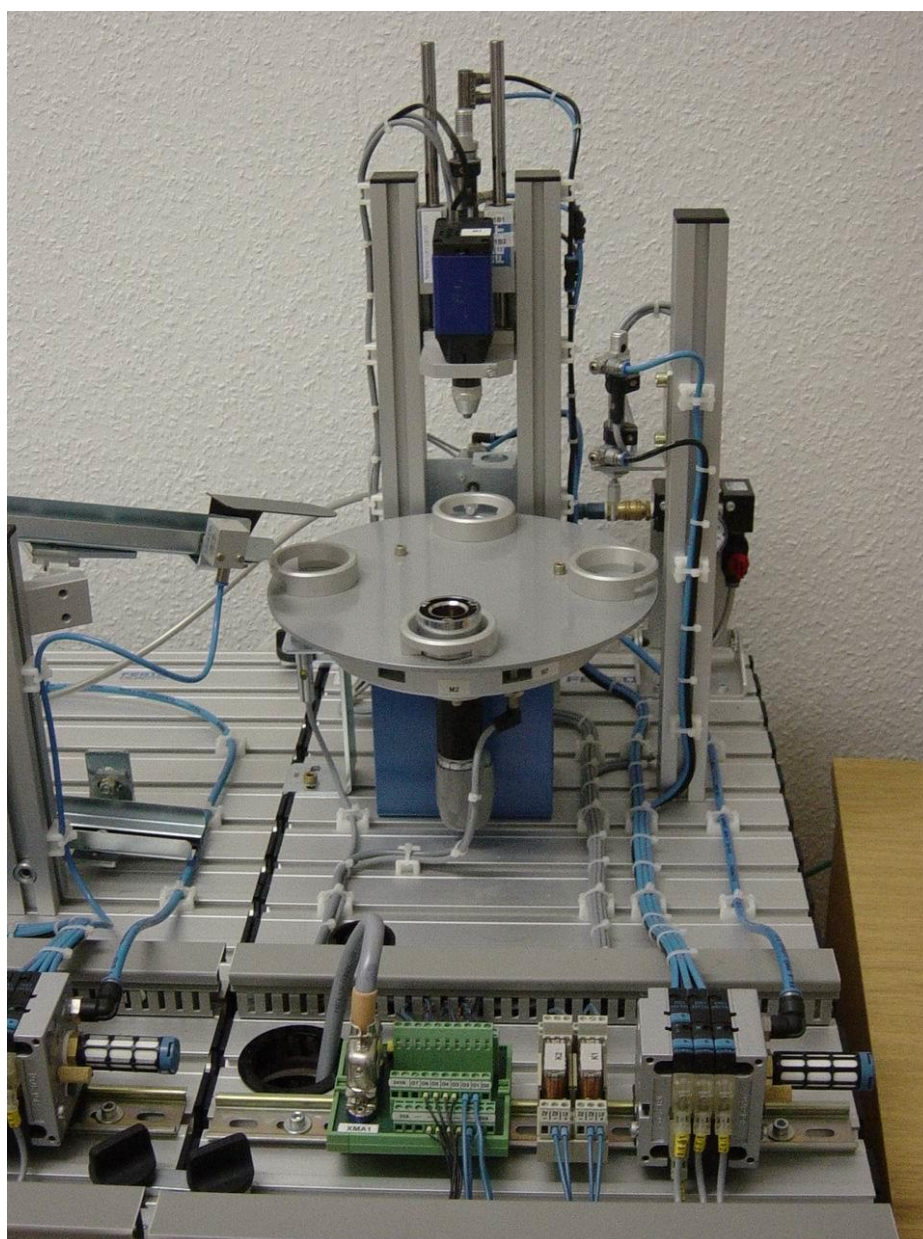


Рисунок 1.4 – Обрабатывающая станция

2 Описание MVC-шаблона проектирования

2.1 Общее описание

Model-view-controller (MVC, «модель-представление-контроллер», «модель-вид-контроллер») — схема использования нескольких шаблонов проектирования, с помощью которых модель приложения, пользовательский интерфейс и взаимодействие с пользователем разделены на три отдельных компонента таким образом, чтобы модификация одного из компонентов оказывала минимальное воздействие на остальные. Данная схема проектирования часто используется для построения архитектурного каркаса, когда переходят от теории к реализации в конкретной предметной области.

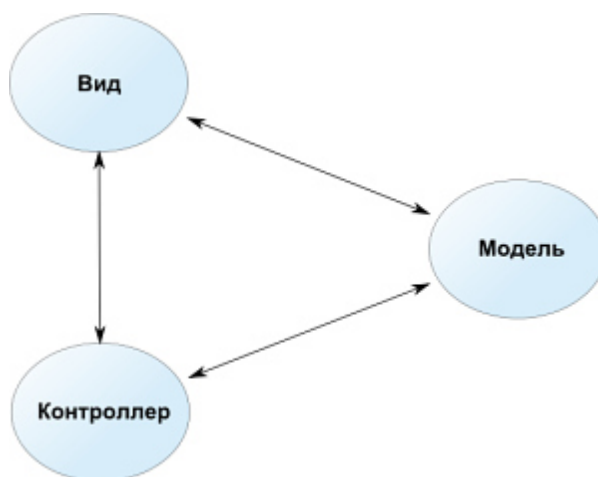


Рисунок 2.1 – Модель MVC

2.1 Модель

Модель представляет собой данные, с которыми оперирует приложение. Это могут быть как данные базы данных, так и любая другая структура данных описывающая некоторые объекты системы и их состояние.

2.2 Вид

Вид представляет собой компонент системы для отображения состояния модели в понятном человеку представлении. Это могут быть диалоги, формы и

другие визуальные и не визуальные (например, синтезатор речи) средства взаимодействия человека с системой. Вид не изменяет данные напрямую (режим только чтение), данные изменяются при помощи контроллера.

2.4 Контроллер

Контроллер является средством, при помощи которого пользователи взаимодействуют с системой. Это может быть клавиатура, манипулятор мышь и т. д. А также является управляющим элементом для обмена данными и сообщениями между видом и моделью.

Фактически, связка вида и контроллера является интерфейсом пользователя. Причем, если компоненты вида обычно можно повторно использовать в других компонентах системы, то контроллер часто является специфичным для данного конкретного случая

Рассмотрим модель MVC на примере CAT-блока в среде NXTStudio.

CAT-блок при создании имеет следующий вид:



Рисунок 2.2 – Модель MVC. CAT-блок в среде NXTStudio

Где CAT1 – сама модель MVC.

ITthis : CAT1_HMI – модель.

sDefault – контроллер и представление.

3 Разработка тестирующей станции

3.1 Разработка базисного блока

3.1.1 Разработка интерфейса базисного блока

Для начала работы нужно создать новый проект в среде NXTStudio.

Для создания базисного блока нужно в Solution Overview проекта найти папку «Basic» и в контекстном меню выбрать пункт «New Item».

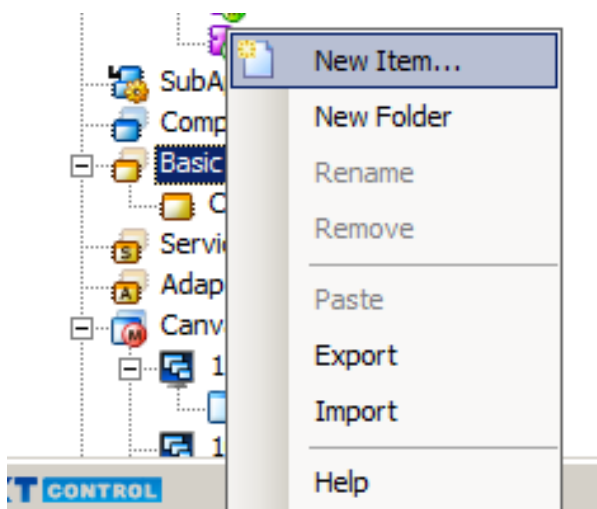


Рисунок 3.1 – Создание базисного блока

Таким образом, создастся базисный блок.

Для начала нам нужно определить входы и выходы, которые потребуются для реализации блока управления. В соответствии с общей схемой устройства, были определены входные и выходные сигналы в базисном блоке управления.

Interface		ECC	Algorithms
Name	Type		
EventInputs			
INIT			
REQ			
START			
<new interface>			
EventOutputs			
INITO			
CNF			
<new interface>			
InputVars			
HIGHT	REAL		
COLOR	REAL		
MATERIAL	REAL		
POWER1	BOOL		
<new variable>			
OutputVars			
POWER	BOOL		
LIFT_UP	BOOL		
LIFT_DOWN	BOOL		
CLOSER	BOOL		
THROW	BOOL		
<new variable>			
Sockets	<new adapter>		
Plugs	<new adapter>		
InternalVars			
POWER_INTERNAL	BOOL		

Рисунок 3.2 – Создание базисного блока

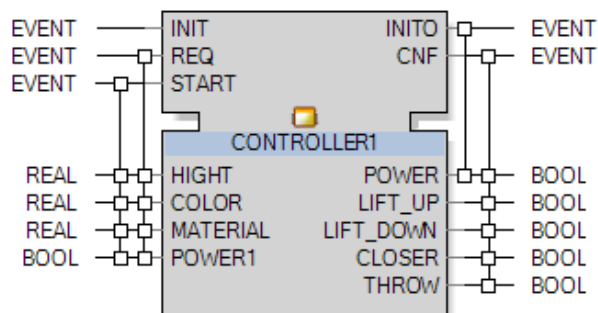


Рисунок 3.3 – Создание базисного блока

Входные сигналы передают данные с датчиков системы, на основе которых будет работать алгоритм оценки заготовки:

- сигнал HIGHT передает данный с датчика измерения высоты заготовки;
- сигнал COLOR передает данные с датчика оценки цвета;

- сигнал MATERIAL передает данные с датчика оценки материала, из которого сделана деталь.

Выходные сигналы в свою очередь нужны для управления устройствами в составе системы:

- сигнал POWER используется для индикации работы станции;
- сигналы LIFT_UP и LIFT_DOWN передают данные о движении лифта вверх и вниз;
- сигнал CLOSER передает данные о закрытии заслонки в желобе;
- сигнал THROW передает данные о сбросе детали при несоответствии требованию.

Также в состав базисного блока была включена переменная POWER_INTERNAL, которая хранила в себе данные о работе системы.

3.1.2 Разработка алгоритма работы базисного блока

Алгоритм базисного блока работает на основе ЕСС-диаграмм.

Чтобы создать новое состояние нужно кликнуть правой кнопкой мыши на холсте на вкладке ЕСС выбрать пункт «New State».

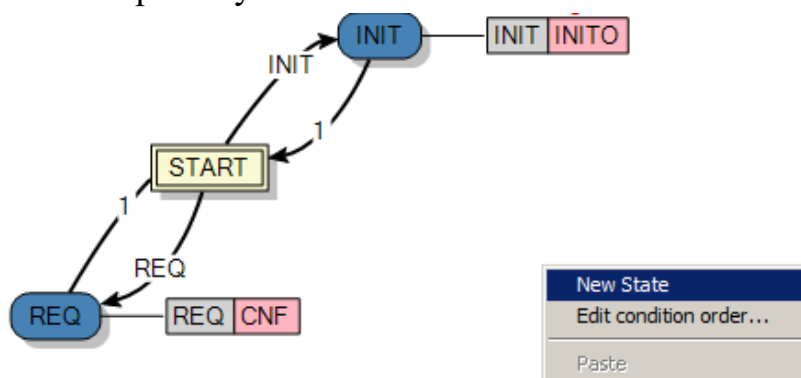


Рисунок 3.4 – Создание базисного блока

Чтобы добавить действие к состоянию, необходимо нажать правой кнопкой мыши на состояние и выбрать пункт «Add Action».

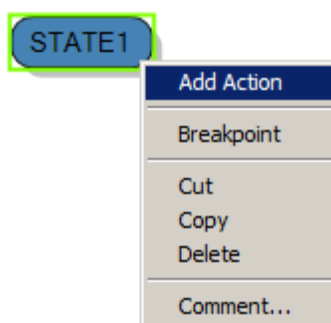


Рисунок 3.5 – Создание базисного блока

Теперь кликнув правой кнопкой мыши на действии, можно выбрать EventInput и EventOutput, которые будут выполняться при попадании в данное состояние.

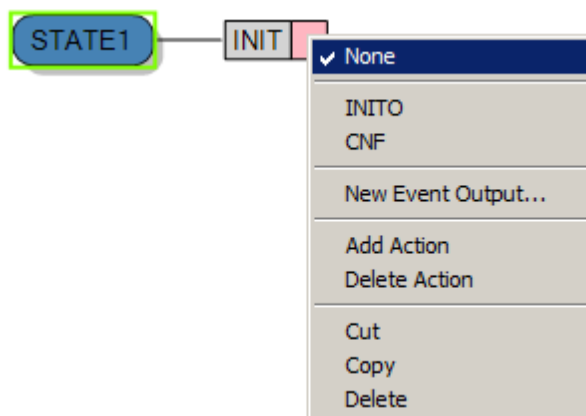


Рисунок 3.6 – Создание базисного блока

На основе разработанных нами интерфейсов базисного блока был разработан алгоритм работы базисного управляющего блока.

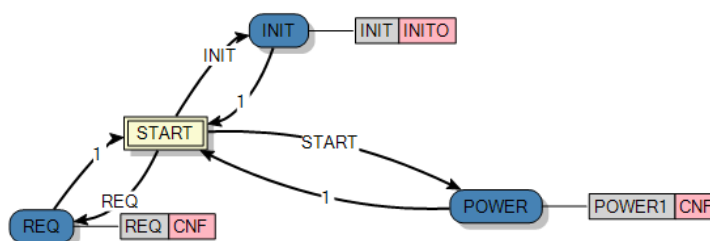


Рисунок 3.7 – ECC – диаграмма базисного блока

Блок имеет следующие EventInputs:

- INIT – инициализация блока. Значение POWER_INTERNAL устанавливается как FALSE, система готова к запуску.

```
ALGORITHM INIT IN ST:
(* Add your comment (as per IEC 61131-3) here
Initialization algorithm
*)
POWER_INTERNAL := FALSE;
END_ALGORITHM
```

- POWER1 – запуск алгоритма тестирования детали. Значение POWER_INTERNAL устанавливается на противоположное. При срабатывании данного EventInput алгоритм или начинает работу, или прекращает ее. Значения всех выходных сигналов сбрасываются в начальные.

```
ALGORITHM POWER1 IN ST:
(* Add your comment (as per IEC 61131-3) here
```

```

*)
IF POWER1 = TRUE THEN
    IF POWER_INTERNAL = FALSE THEN
        POWER_INTERNAL := TRUE;
        LIFT_DOWN := TRUE;
        LIFT_UP := FALSE;
        CLOSER := TRUE;
        THROW:= TRUE;

    ELSE
        POWER_INTERNAL := FALSE;
        LIFT_DOWN := FALSE;
        LIFT_UP := TRUE;
        CLOSER := TRUE;
        THROW:= FALSE;

    END_IF;
END_IF;
END_ALGORITHM

```

- REQ – происходит считывание данных со всех датчиков, анализ значений и передача выходных значений на устройства. Этот EventInput обеспечивает работу алгоритма тестирования заготовки.

Также блок имеет следующие EventOutputs:

- INITO – последующая инициализация устройств в составе ПВУ.
- CNF - EventOutput подающий сигнал на EventInput-входы датчиков в составе станции.

```

ALGORITHM REQ IN ST:
(* Add your comment (as per IEC 61131-3) here
Normally executed algorithm
*)
IF POWER_INTERNAL = TRUE THEN
    IF COLOR < 300 THEN
        IF COLOR > 100 THEN
            IF MATERIAL < 300 THEN
                IF MATERIAL > 100 THEN
                    LIFT_UP := TRUE;
                    LIFT_DOWN := FALSE;
                ELSE THROW := FALSE;
                END_IF;
            ELSE THROW := FALSE;
            END_IF;
        ELSE THROW := FALSE;
        END_IF;
    ELSE THROW := FALSE;
    END_IF;

    IF HIGHT < 300 THEN
        IF HIGHT > 100 THEN
            CLOSER := FALSE;
        ELSE THROW := FALSE;
        END_IF;
    ELSE THROW := FALSE;
    END_IF;
    THROW := TRUE;
    LIFT_UP := FALSE;
    LIFT_DOWN := TRUE;
END_IF;

```

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		18

```
POWER := POWER_INTERNAL;
END_ALGORITHM
```

3.1.3 Тестирование и отладка базисного блока

Среда NXTStudio позволяет производить отладку базисных блоков.

Для начала нам нужно проинициализировать базисный блок через EventInput «INIT», тем самым установив значение POWER_INTERNAL – FALSE. Это значит что система готова к запуску и ждет сигнала на EventInput «POWER».

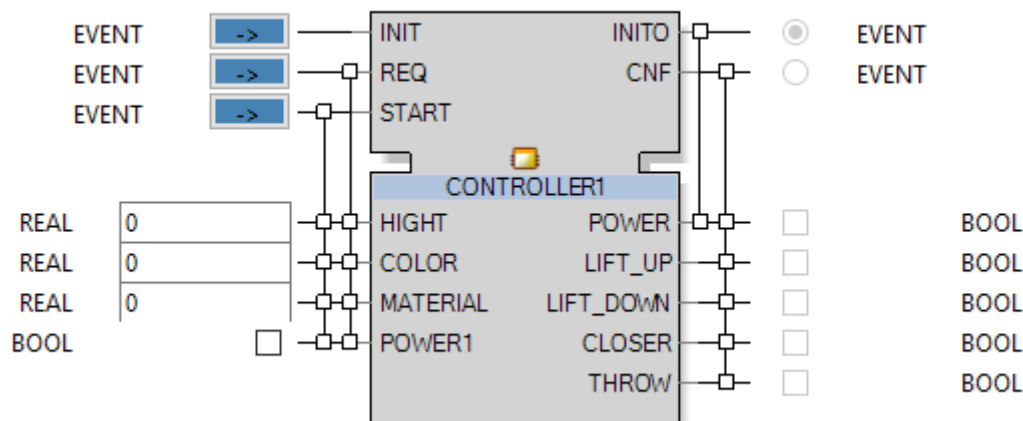


Рисунок 3.8 – Отладка базисного блока управления

Следующим шагом будет подача сигнала на EventInput «POWER», которая установит стартовые значения для всех устройств в составе станции и активирует EventOutput «CNF».

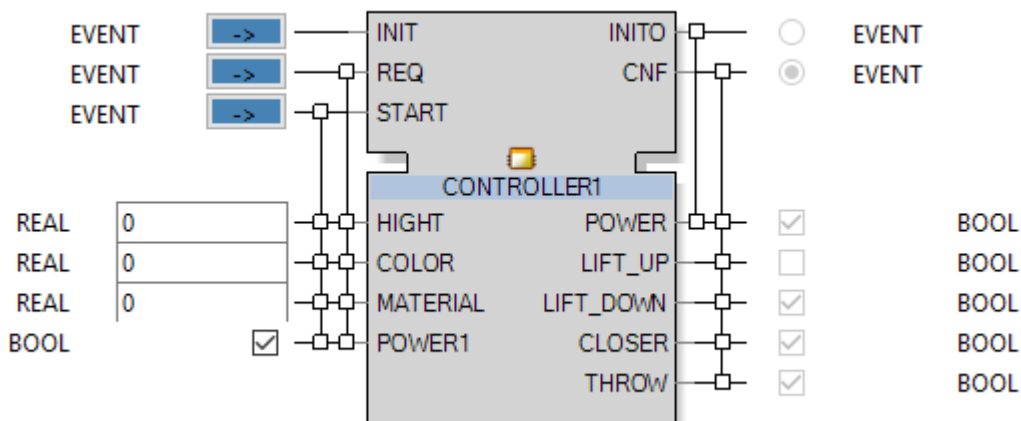


Рисунок 3.9 – Отладка базисного блока управления

Следующим шагом будет передача параметров со всех датчиков через EventInput «REQ», который запустит основной алгоритм базисного блока и в соответствии с входными параметрами выдаст на выходах соответствующие значения.

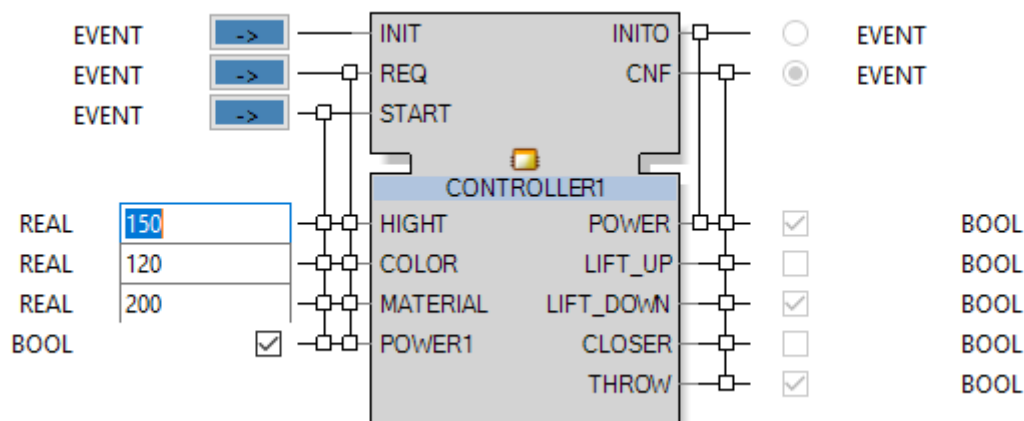


Рисунок 3.10 – Отладка базисного блока управления

3.2 Разработка составных и САТ-блоков тестирующей станции

3.2.1 Разработка САТ-блоков

Для создания САТ-блока нужно в среде NXTStudio 2.0 в «Solution Overview» необходимо найти папку «CAT», выбрать папку «Application», нажать правой кнопкой мыши и выбрать пункт «New Item»

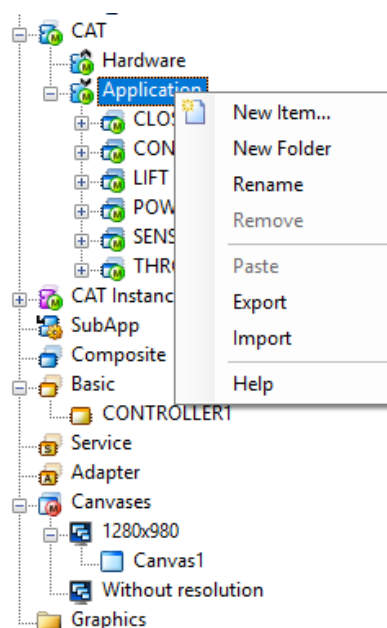


Рисунок 3.11 – Создание САТ-блока

В следующем окне нужно написать имя композитного блока и выбрать тип «Normal CAT»

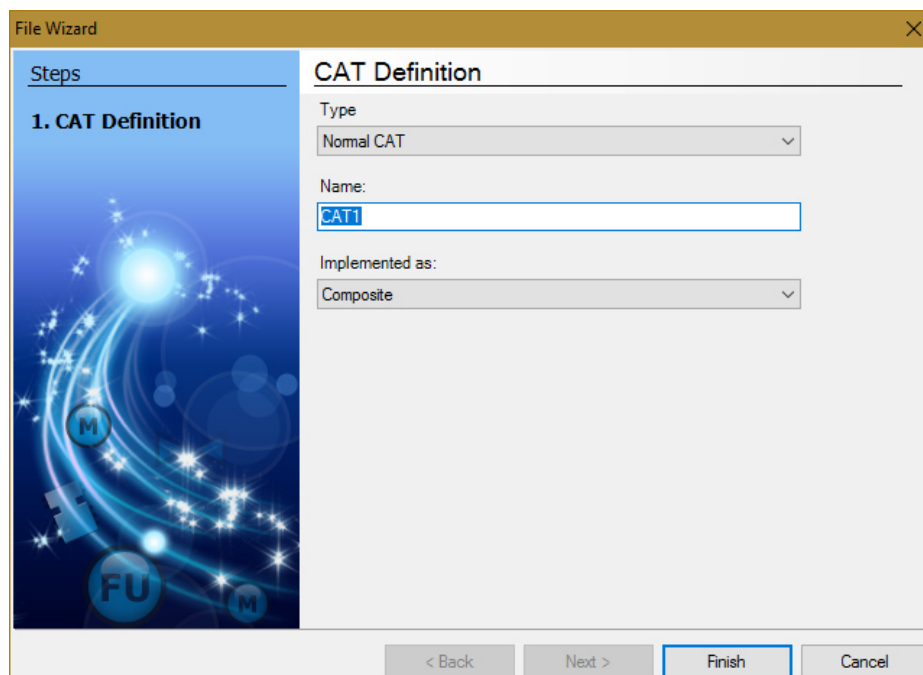


Рисунок 3.12 – Создание CAT-блока



Рисунок 3.13 – Создание CAT-блока

На рисунке можно увидеть как выглядит CAT блок в «Solution View» в проекте. CAT1 – непосредственно сам композитный блок, IThis : CAT1_HMI – блок визуализации HMI(Human Machine Interface), sDefault –графическое изображение блока визуализации.

Для каждого устройства станции необходимо создать композитный блок, который содержит входы и выходы, необходимые для ее работы.

3.2.2 Блок Closer – заслонка

Name	Type	Arra...	Initial v...	With
EventInputs				
INIT				QI
REQ				QI
<new interface>				
EventOutputs				
<new interface>				
InputVars				
QI	BOOL			
<new variable>				
OutputVars				
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 3.14 –CAT-блок Closer

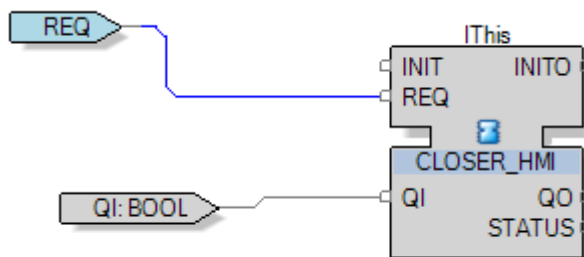


Рисунок 3.15 –CAT-блок Closer

CAT-блок содержит 1 вход – Q1(bool), EventInput REQ и блок визуализации. Значение Q1 может принимать 2 значения – TRUE(заслонка открыта) и FALSE(заслонка закрыта).

3.2.3 Блок Lift

Name	Type	Arra...	Initial v...	With
EventInputs				
INIT				QI
REQ				QI
< new interface >				
EventOutputs				
INITO				
CNF				
< new interface >				
InputVars				
QI	BOOL			
< new variable >				
OutputVars				
< new variable >				
Sockets				
< new adapter >				
Plugs				
< new adapter >				
AdapterInputs				
< new adapter >				
AdapterOutputs				
< new adapter >				

Рисунок 3.16 –CAT-блок Lift

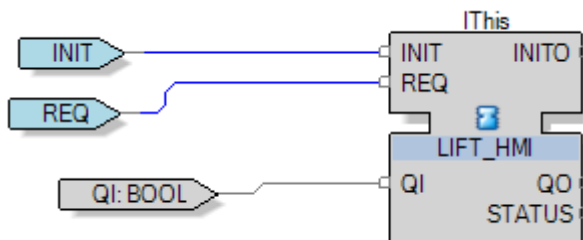


Рисунок 3.17 –CAT-блок Lift

CAT-блок содержит 1 вход – Q1(bool), EventInput REQ, INIT и блок визуализации. Значение Q1 может принимать 2 значения – TRUE(лифт поднят) и FALSE(лифт опущен).

3.2.4 Блок Power button

Name	Type	Arra...	Initial v...	With
EventInputs				
INIT				QI
REQ				QI
<new interface>				
EventOutputs				
INITO				QO
POWER_TRIGGER				QO
<new interface>				
InputVars				
QI	BOOL			
<new variable>				
OutputVars				
QO	BOOL			
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 3.18 –CAT-блок Power button

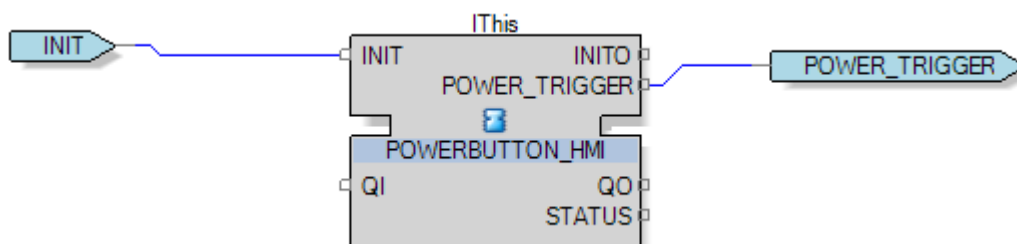


Рисунок 3.19 –CAT-блок Power button

Блок PowerButton не содержит входов и выходов, однако содержит EventOutput «POWER_TRIGGER» который будет вызываться из блока визуализации при нажатии клавиши.

3.2.5 Блок Sensor

Interface		Composite			
Name	Type	Arra...	Initial v...	With	
EventInputs					
INIT					
REQ					
< new interface >					
EventOutputs					
INITO				QO	
CNF				QO	
< new interface >					
InputVars					
< new variable >					
OutputVars					
QO	REAL				
< new variable >					
Sockets					
< new adapter >					
Plugs					
< new adapter >					
AdapterInputs					
< new adapter >					
AdapterOutputs					
< new adapter >					

Рисунок 3.20 –CAT-блок Sensor

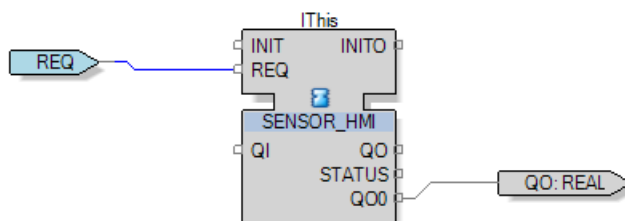


Рисунок 3.21 –CAT-блок Sensor

Блок Sensor содержит выход QO, который передает значение. Также содержит блок визуализации.

3.2.6 Блок Throw

Name	Type	A
EventInputs		
INIT		
REQ		
<new interface>		
EventOutputs		
<new interface>		
InputVars		
QI	BOOL	
<new variable>		
OutputVars		
<new variable>		
Sockets		
<new adapter>		
Plugs		
<new adapter>		
AdapterInputs		
<new adapter>		
AdapterOutputs		
<new adapter>		

Рисунок 3.22 –CAT-блок Throw

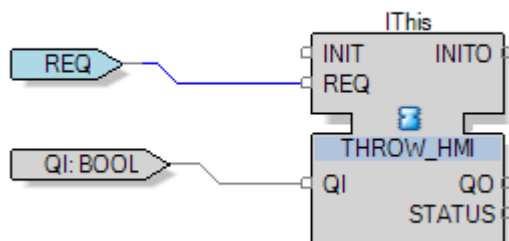


Рисунок 3.23 –CAT-блок Throw

CAT-блок содержит 1 вход – Q1(bool), EventInput REQ и блок визуализации. Значение Q1 может принимать 2 значения – TRUE(сброс) и FALSE(не сброс).

3.2.7 Блок Controller

Name	Type	Arra...	Initial v...	With	
EventInputs					
INIT					
REQ				HIGHT, COLO...	
START				HIGHT, COLO...	
<new interface>					
EventOutputs					
INITO				POWER	
CNF				POWER, LIFT_...	
<new interface>					
InputVars					
HIGHT	REAL				
COLOR	REAL				
MATERIAL	REAL				
POWER1	BOOL				
<new variable>					
OutputVars					
LIFT_UP	BOOL				
LIFT_DOWN	BOOL				
CLOSER	BOOL				
THROW	BOOL				
POWER	BOOL				
<new variable>					

Рисунок 3.24 –CAT-блок Controller

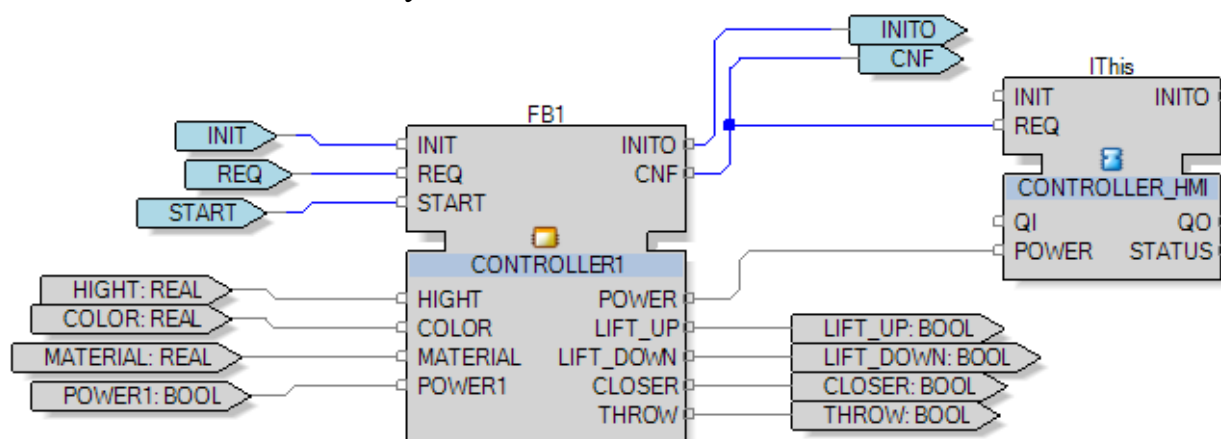


Рисунок 3.25 –CAT-блок Controller

Блок Controller – основной композитный блок, в котором содержится базисный блок Controller1 с алгоритмом работы. Композитный блок Controller копирует входы и выходы базисного блока Controller1.

3.3 Разработка приложения

Система композитных блоков содержится в файле приложения «APP1».

Для переноса компонентов в приложение необходимо создать CAT Instance Application на основе ранее созданных композитных блоков.

Для создания CAT Instance Application необходимо в Solution View проекта найти папку «CAT Instance» и пронаследовать объекты от композитных блоков.

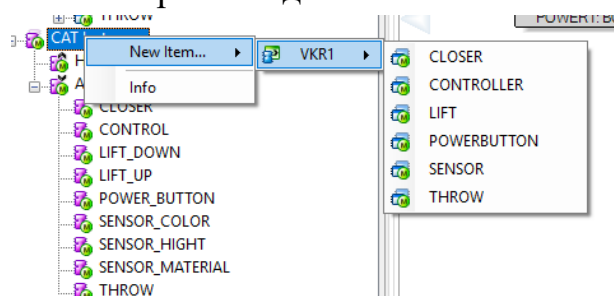


Рисунок 3.26 – Создание CAT Instance Application

Можно пронаследовать несколько CAT Instance Application от одного CAT-блока. Таким образом для создания однотипных объектов можно создать один CAT-блок.

Исходя из описания компонентов ПВУ были созданы необходимые CAT Instance Application.

Папка CAT Instance Application стала выглядеть следующим образом:

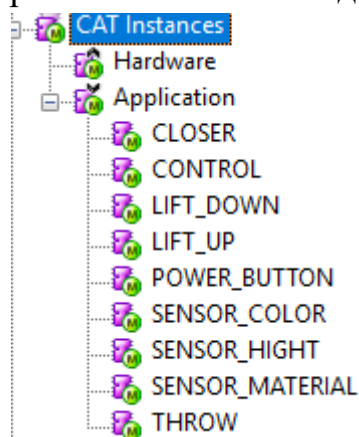


Рисунок 3.27 – Создание CAT Instance Application

В дальнейшем переносим объекты из папки Application в файл приложения «System.sys > APP» и организуем связи входов, выходов и событий. В конечном итоге файл «System.sys > APP» стал выглядеть следующим образом:

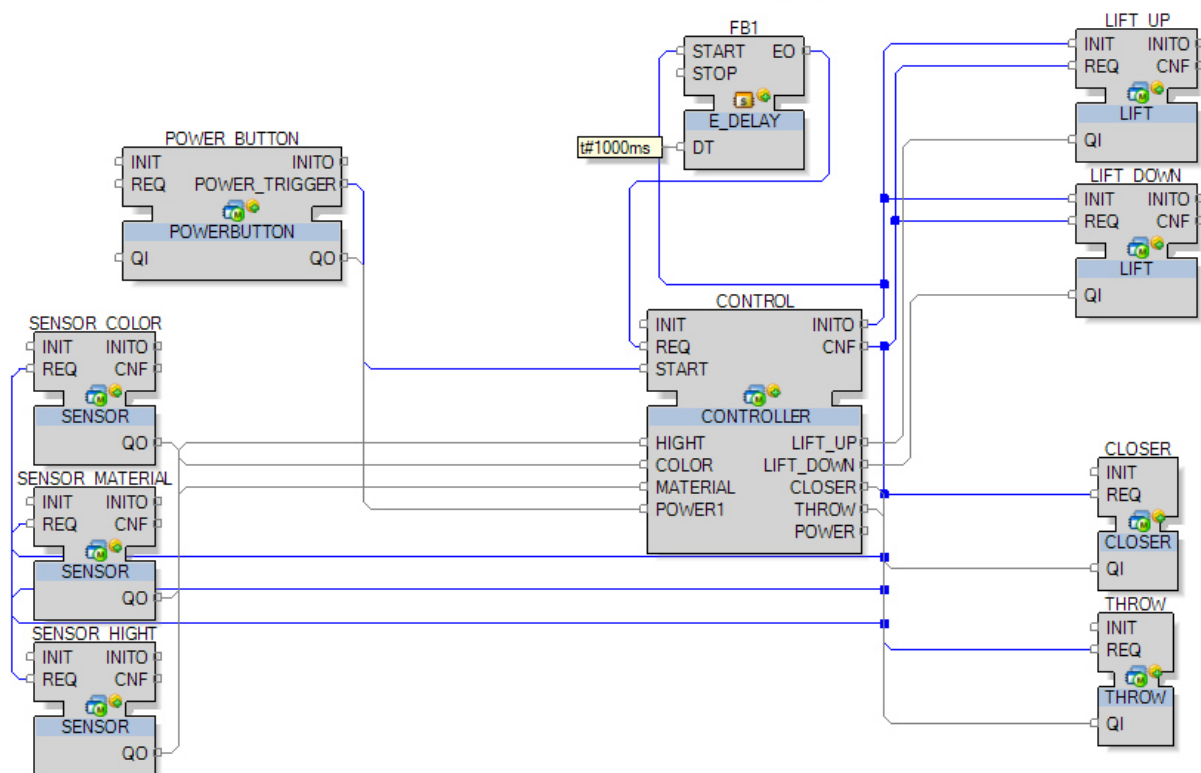


Рисунок 3.28 – Приложение System.sys > APP

3.4 Разработка визуальной модели

Элементы НМІ(Human Machine Interface) – элементы в составе композитных блоков, позволяющие организовать простой и понятный интерфейс пользователя, а также ввод и вывод нужных нам параметров на экран.

Элементы НМІ имеют возможность создания графического интерфейса блоков на языке C#.

При создании каждого композитного блока автоматически создается блок НМІ, который нам необходимо подключить к соответствующим входам и выходам. Блоки НМІ имеют возможность показа параметров на экране а также передачу параметров через элементы интерфейса.

Элементы интерфейса komponуются нужным образом на файле типа «*.cnv» и создают интерфейс для работы пользователя.

Для начала создадим композитный блок и выберем файл визуализации. Для визуализации какого-либо параметра необходимо перетащить мышкой нужную нам переменную на холст.

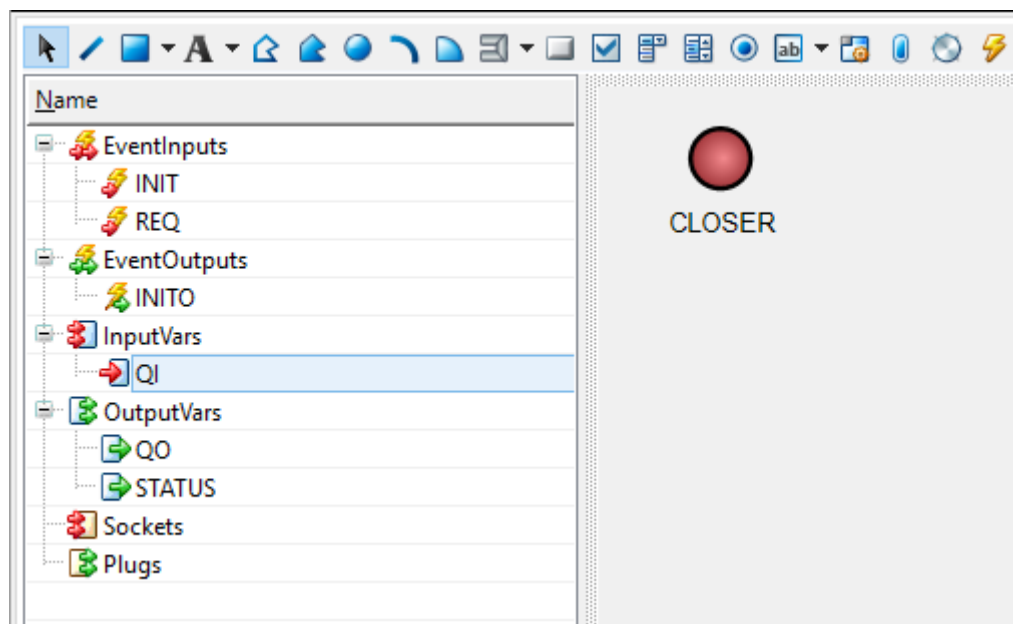


Рисунок 3.29 – Создание HMI

В зависимости от типа переменной появится список возможных элементов отображения переменной.

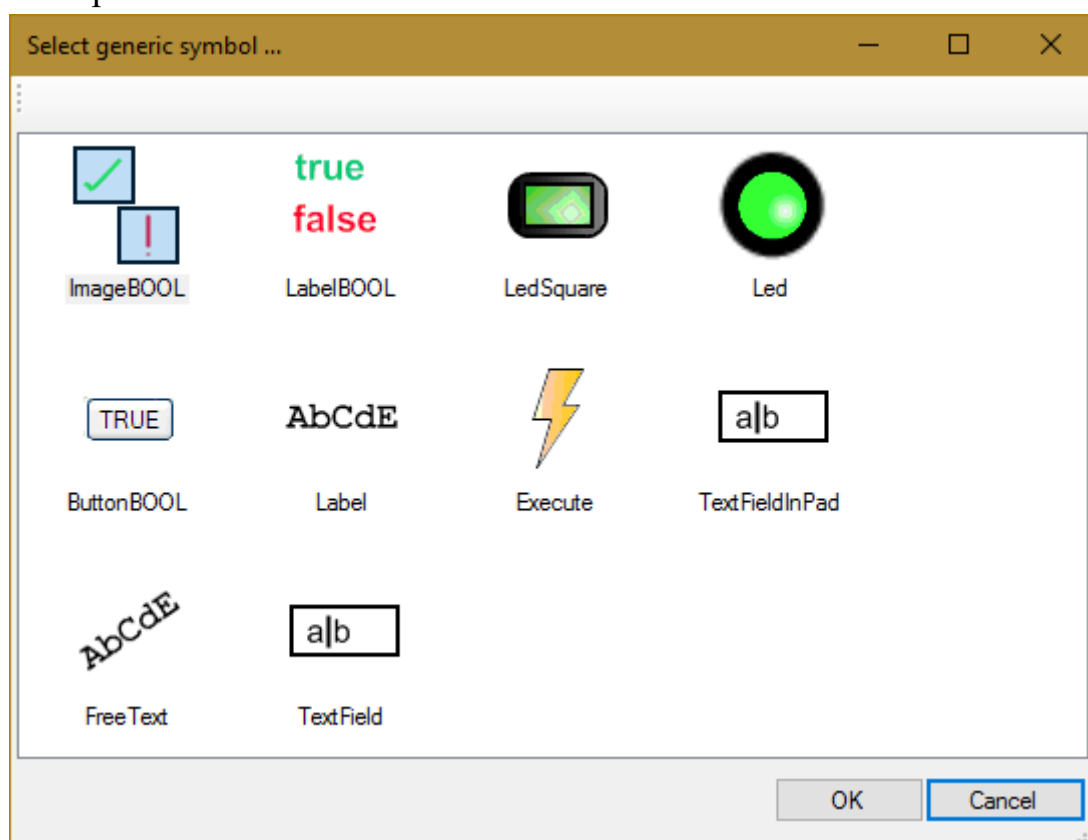


Рисунок 3.30 – Создание HMI

Таким образом создадим элементы визуализации HMI для каждого композитного блока:

- заслонка



Рисунок 5.3 – Создание HMI

LED-индикатор будет цветом отображать состояние заслонки. Зеленый – открыта, красный – закрыта.

- лифт



Рисунок 3.31 – Создание HMI

LED-индикатор будет цветом отображать состояние лифта (поднят или опущен).

- датчик цвета

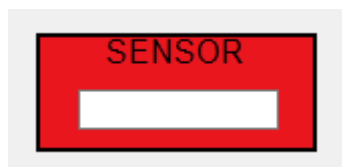


Рисунок 3.32 – Создание HMI

Поскольку при разработке нет возможности получать данные о цвете с физических датчиков, данные о цвете будут подаваться на контроллер вручную. На работу алгоритма это не повлияет

- датчик материала

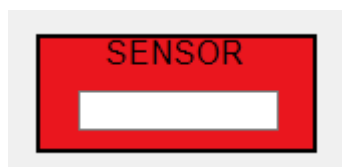


Рисунок 3.33 – Создание HMI

Поскольку при разработке нет возможности получать данные о материале с физических датчиков, данные о материале будут подаваться на контроллер вручную. На работу алгоритма это не повлияет

- датчик высоты

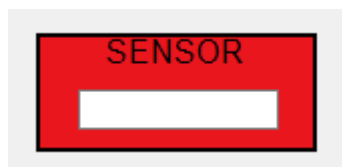


Рисунок 3.34 – Создание HMI

Поскольку при разработке нет возможности получать данные о высоте с физических датчиков, данные о высоте будут подаваться на контроллер вручную. На работу алгоритма это не повлияет

- индикатор сброса

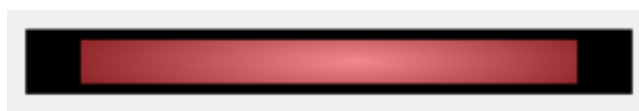


Рисунок 3.35 – Создание HMI

- кнопка включения

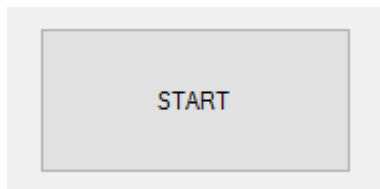


Рисунок 3.36 – Создание HMI

Кнопка включения при нажатии на нее будет вызывать EventOutput.

3.5 Визуализация работы

Для создания пользовательского интерфейса необходимо скомпоновать все графические представления на холсте. Сначала нужно создать холст, на котором будут отображаться необходимые нам индикаторы и элементы управления.

Для этого в Solution Overview найдем папку Canvases, выберем или создадим нужное нам разрешение, и в контекстном меню выберем пункт «New Item».

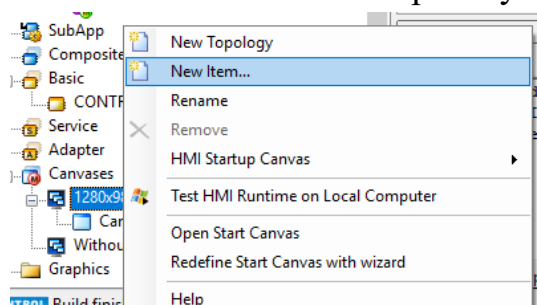


Рисунок 3.37 – Создание HMI. Добавление холста

Откроем полученный холст (Canvas) и перенесем наши блоки из папки CAT Instances : Application на холст. Скомпонуем все нужным образом и получим следующее представление:

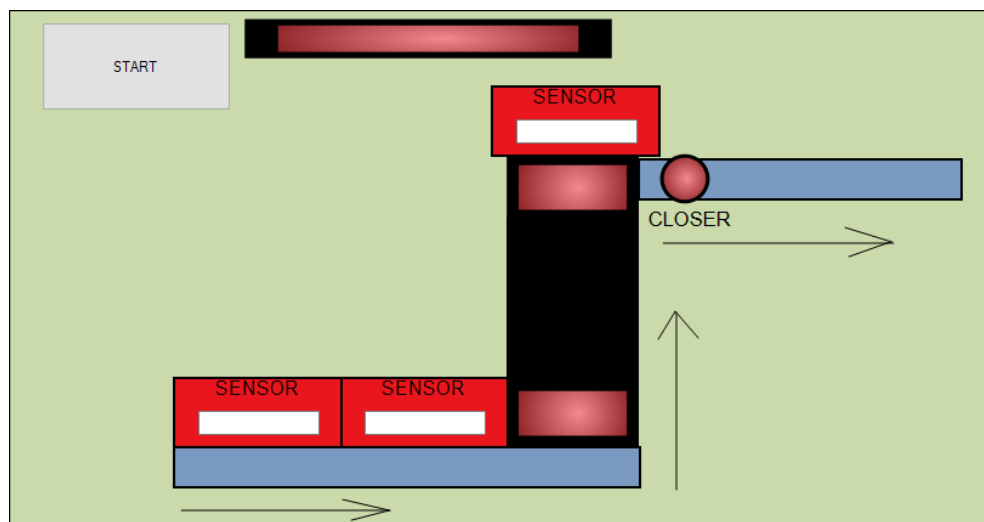


Рисунок 3.38 – Создание HMI

Интерфейс пользователя включает в себя визуализацию каждого устройства в составе тестирующей станции.

3.6 Запуск приложения в среде NXTStudio 2.0

Конечным шагом в разработке системы управления ПВУ в среде NXTStudio будет запуск приложения.

Перед началом работы следует сохранить все открытые файлы, иначе следующие шаги не будут возможными. Для запуска нашего приложения необходимо открыть файл System.sys и перейти на вкладку Devices. Далее следует выбрать Active Network Profile и из выпадающего списка выбрать «Local Test». Далее для каждого устройства необходимо в поле «Login» выбрать пункт «Yes». Далее выделим все устройства и в контекстном меню выберем пункт «Start Soft PLC».

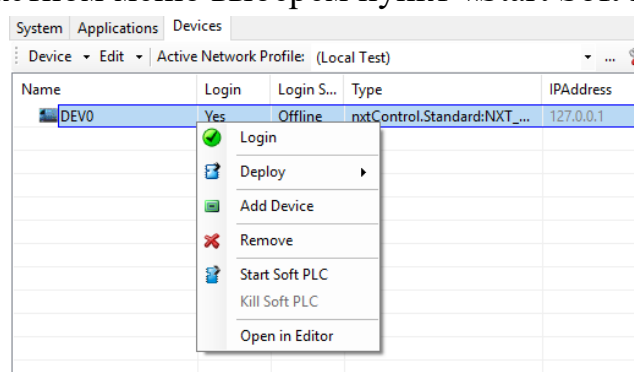


Рисунок 3.39 – Запуск приложения

После чего на компьютере будут запущены виртуальные PLC.

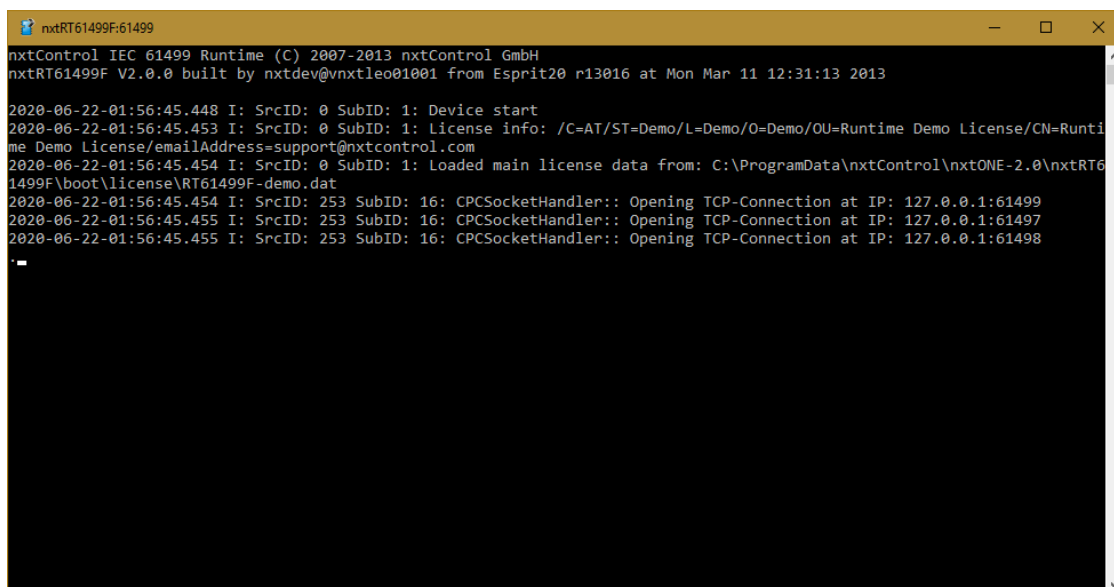


Рисунок 3.40 – Запуск приложения

Далее выберем в меню программы пункт Device > Deploy > Deploy Advanced.

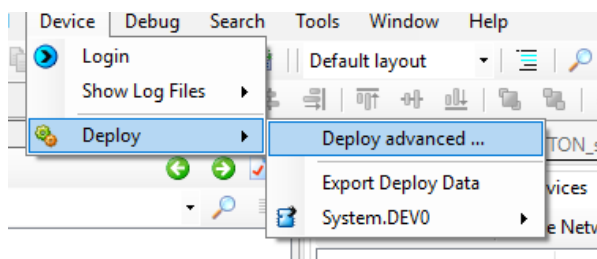


Рисунок 3.41 – Запуск приложения

После вышеописанных шагов выполнится Build проекта. В Открывшемся окошке нажимаем правой кнопкой мыши на устройстве из списка устройств и выбираем пункт «Delete Persistent Data». Данную процедуру выполняем для всех устройств в проекте.

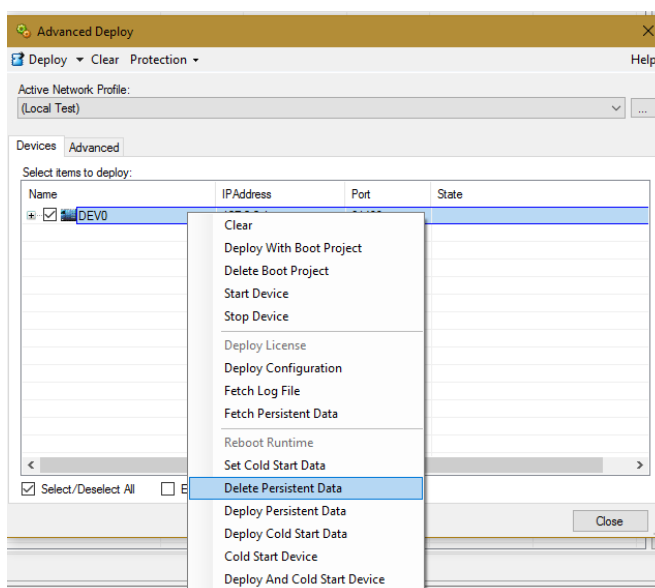


Рисунок 3.41 – Запуск приложения

Если все шаги были выполнены правильно, напротив каждого устройства в строке State появится запись «OK». Далее нужно выделить все устройства и нажать кнопку «Deploy».

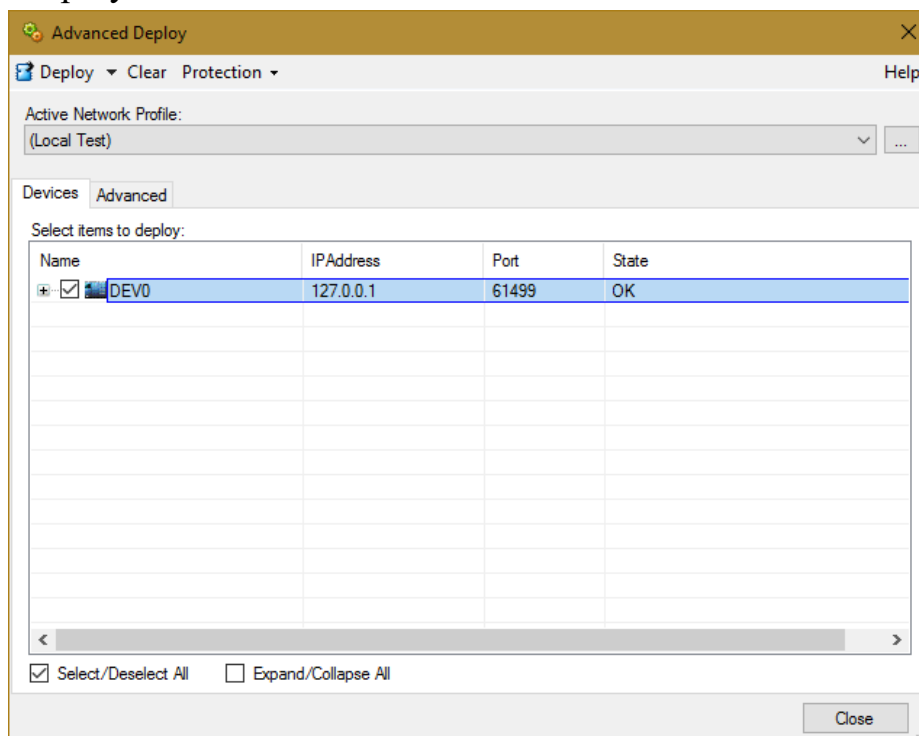


Рисунок 3.42 – Запуск приложения

После успешной операции «Deploy» данное окно должно автоматически закрыться. Теперь необходимо в файле System.sys > Devices нажать кнопку «Login». Название кнопки сменится на «Logout» а в графе Login State каждого из устройств появится запись «Online». После этого шага проект будет готов к запуску.

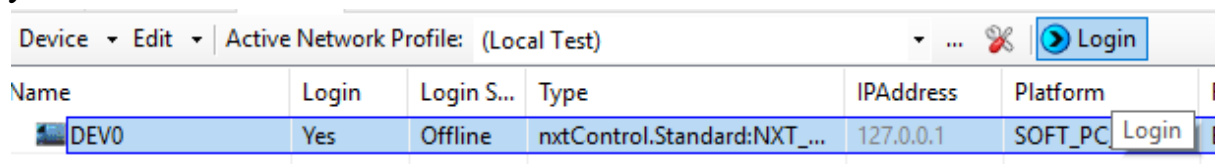


Рисунок 3.43 – Запуск приложения

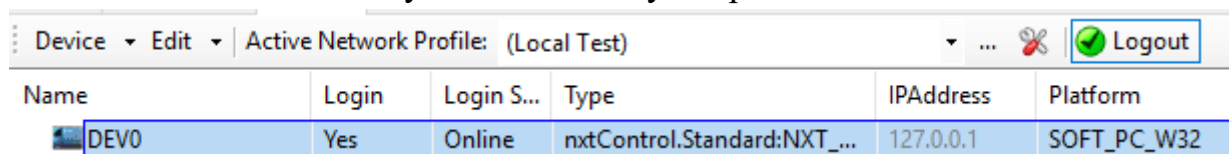


Рисунок 3.44 – Запуск приложения

Теперь в Solution Overview находим папку «Canvases», находим холст с нужным нам разрешением, нажимаем правой кнопкой и выбираем пункт «Test HMI Runtime on Local Computer».

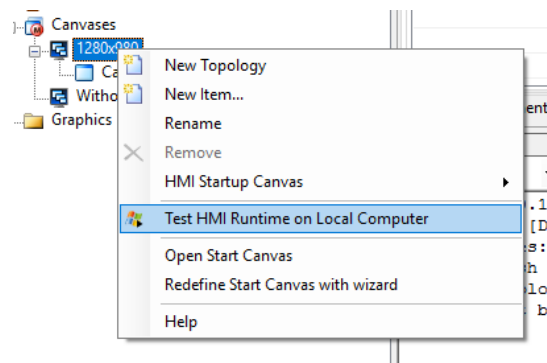


Рисунок 3.45 – Запуск приложения

Откроется холст с интерфейсом проекта.

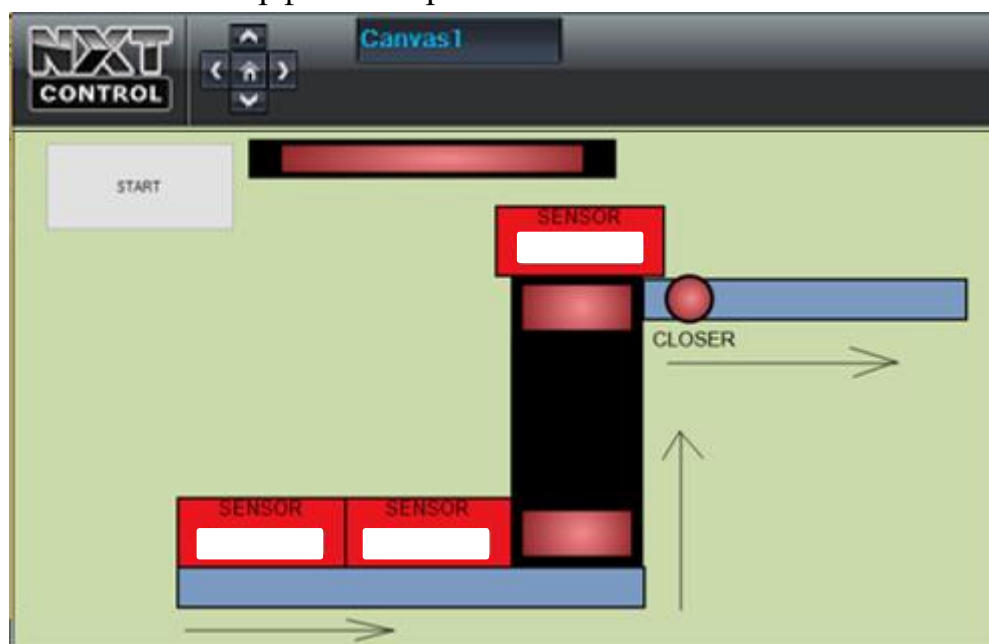


Рисунок 3.46 – Запуск приложения

3.7 Тестирование приложения

В процессе разработки был выяснен следующий аспект разработки алгоритма системы тестирующей станции – алгоритм напрямую зависит от данных с физических датчиков цвета, материала и высоты, к которым нет возможности получить доступ без наличия самой станции. Решением данной проблемы было добавление возможности ручного изменения данных, которые должны считываться с физических датчиков. Таким образом появилась возможность тестирования приложения без наличия станции.

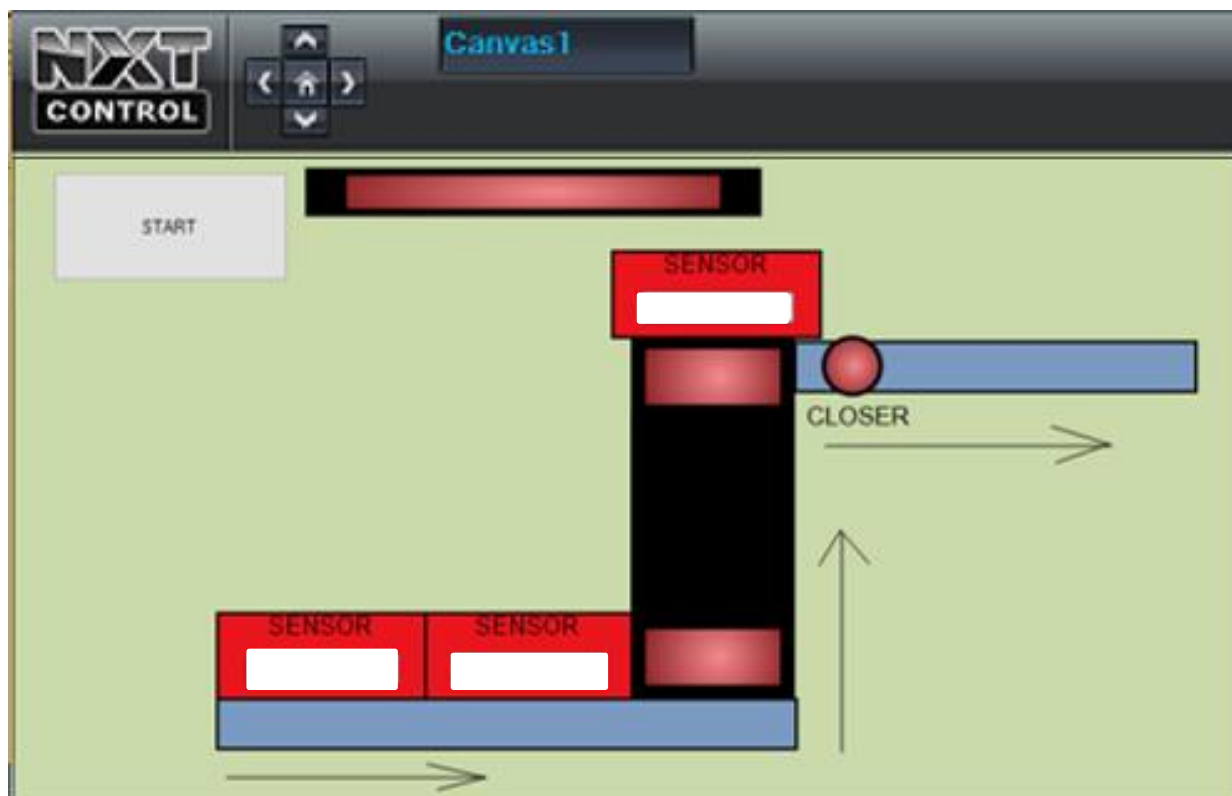


Рисунок 3.47 – Начальное состояние приложения
Начальным шагом будет запись данных с датчиков цвета и материала.

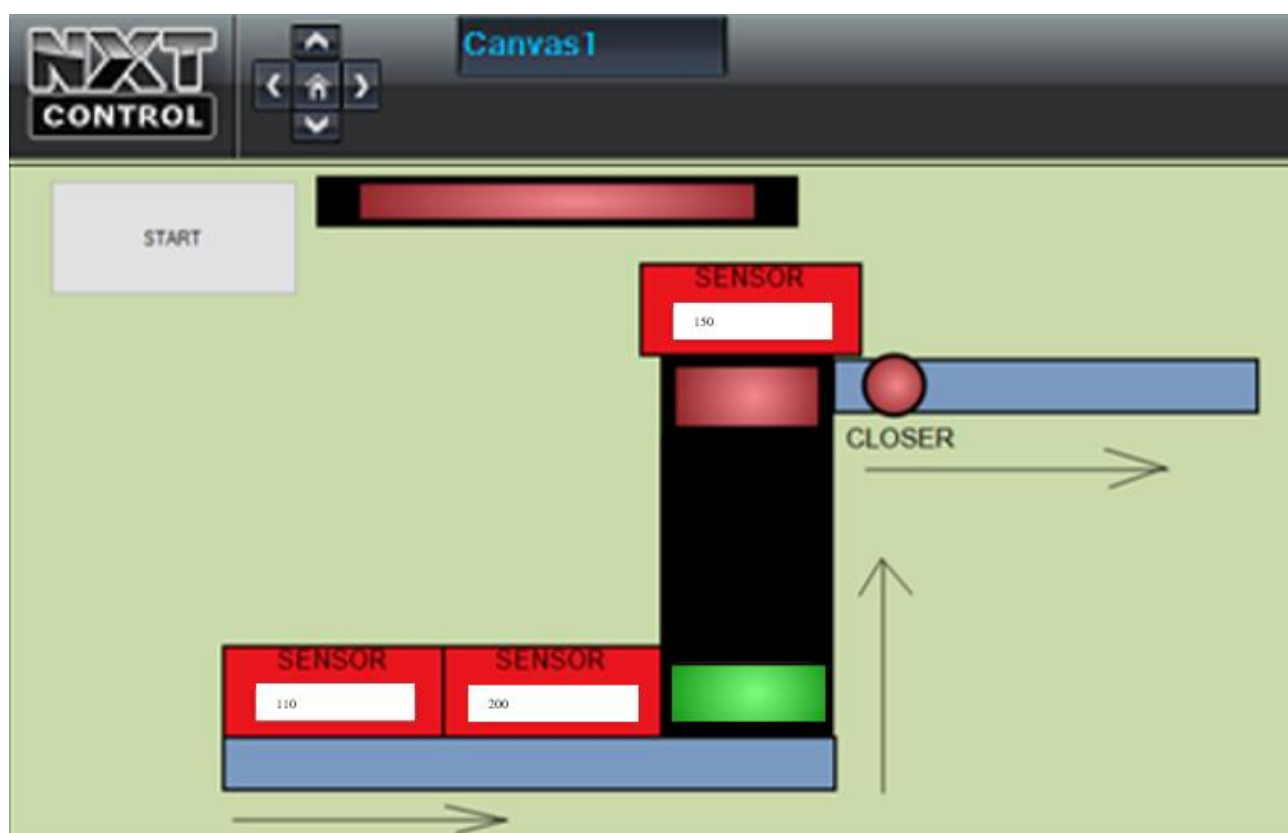


Рисунок 3.48 – Тестирование приложения
После считываются данные с датчика высоты.

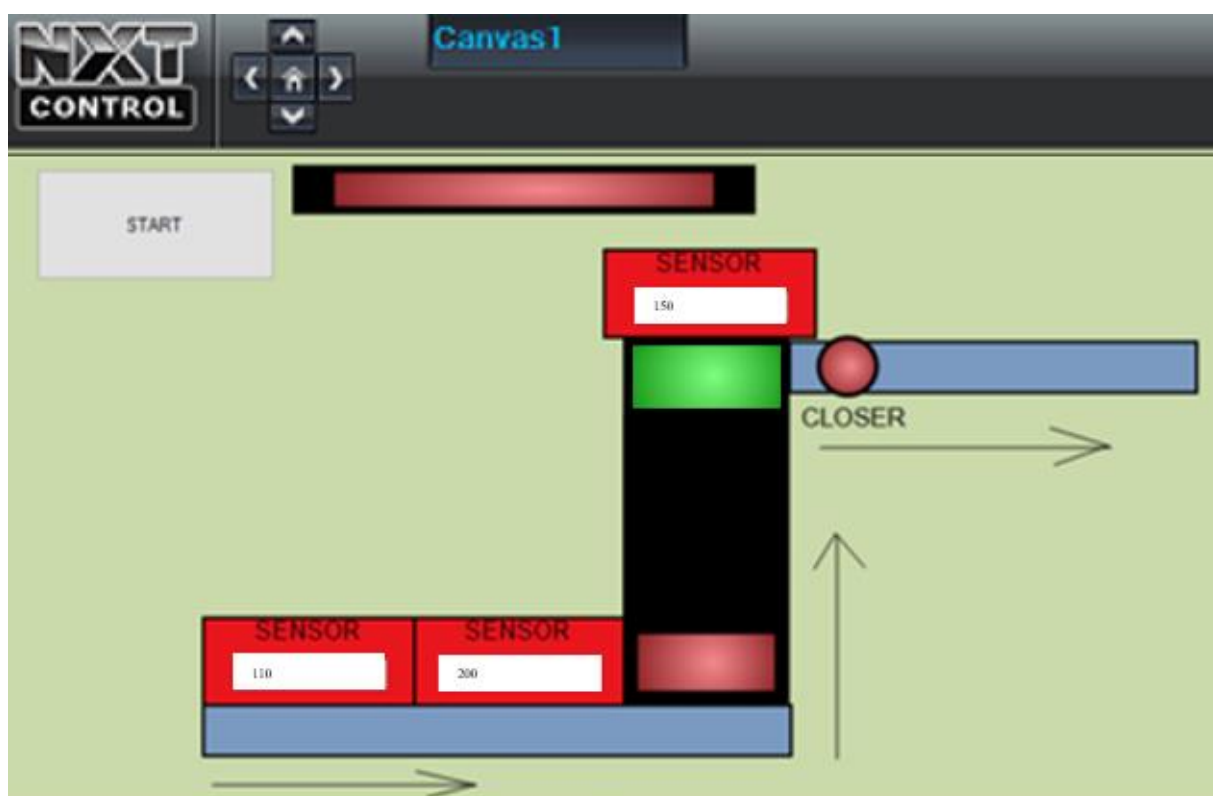


Рисунок 3.49 – Тестирование приложения
Если данные со всех датчиков удовлетворяют условиям, то заслонка открывается и деталь проходит дальше по желобу.

Рисунок 4.1 – Создание базисного блока

Таким образом, создастся базисный блок.

Для начала нам нужно определить входы и выходы, которые потребуются для реализации блока управления. В соответствии с общей схемой устройства, были определены входные и выходные сигналы в базисном блоке управления.

Name	Type	Arra...	Initial v...	With
EventInputs				
INIT				
REQ1				PLACE
REQ2				DREL_OK
REQ3				TEST_OK
REQ4				
START				DREL_OK, TES...
<new interface>				
EventOutputs				
INITO				POWER
CNF				GRAB, DREL, T...
CNF1				GRAB, SPIN, P...
CNF2				DREL, SPIN, P...
CNF3				TEST, SPIN, PO...
CNF4				GRAB, DROP, S...
<new interface>				
InputVars				
DREL_OK	BOOL			
TEST_OK	BOOL			
PLACE	BOOL			
<new variable>				
OutputVars				
GRAB	BOOL			
DREL	BOOL			
TEST	BOOL			
DROP	BOOL			
SPIN	BOOL			
POWER	BOOL			
<new variable>				

Рисунок 4.2 – Создание базисного блока

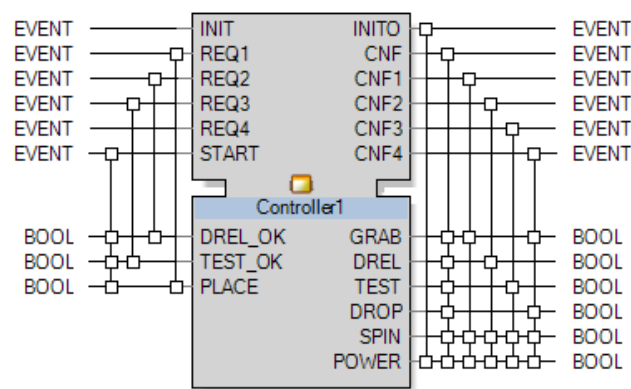


Рисунок 4.3 – Создание базисного блока

Входные сигналы передают данные с датчиков системы, на основе которых будет работать алгоритм оценки заготовки:

- сигнал DREL_OK передает данные об окончании работы дрели;
- сигнал TEST_OK передает данные об окончании проверки изделия;
- сигнал PLACE передает данные о том, что деталь помещена на стол.

Выходные сигналы в свою очередь нужны для управления устройствами в составе системы:

- сигнал POWER используется для индикации работы станции;
- сигналы GRAB передают данные о захвате детали зажимами;
- сигнал DREL передает данные о начале работы дрели;
- сигнал TEST передает данные о начале проверки детали;
- сигнал DROP передает данные о сбросе детали;
- сигнал SPIN передает данные о вращении стола.

Также в состав базисного блока была включена переменная POWER_INTERNAL, которая хранила в себе данные о работе системы.

4.1.2 Разработка алгоритма работы базисного блока

Алгоритм базисного блока работает на основе ЕСС-диаграмм.

Чтобы создать новое состояние нужно кликнуть правой кнопкой мыши на холсте на вкладке ЕСС выбрать пункт «New State».

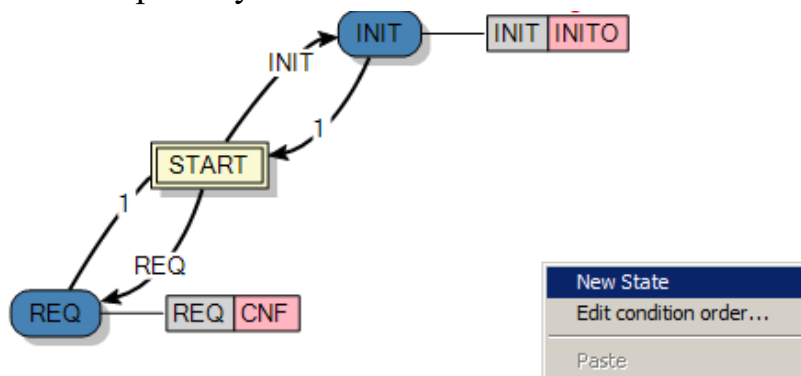


Рисунок 4.4 – Создание базисного блока

Чтобы добавить действие к состоянию, необходимо нажать правой кнопкой мыши на состояние и выбрать пункт «Add Action».

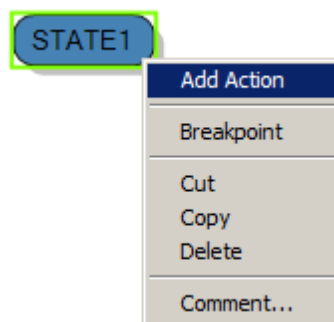


Рисунок 4.5 – Создание базисного блока

Теперь кликнув правой кнопкой мыши на действии, можно выбрать EventInput и EventOutput, которые будут выполняться при попадании в данное состояние.

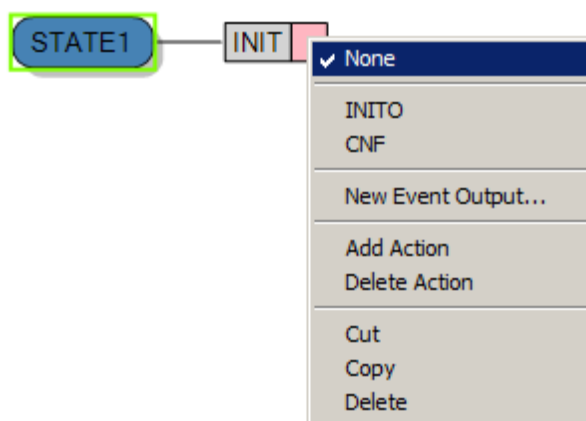


Рисунок 4.6 – Создание базисного блока

На основе разработанных нами интерфейсов базисного блока был разработан алгоритм работы базисного управляющего блока.

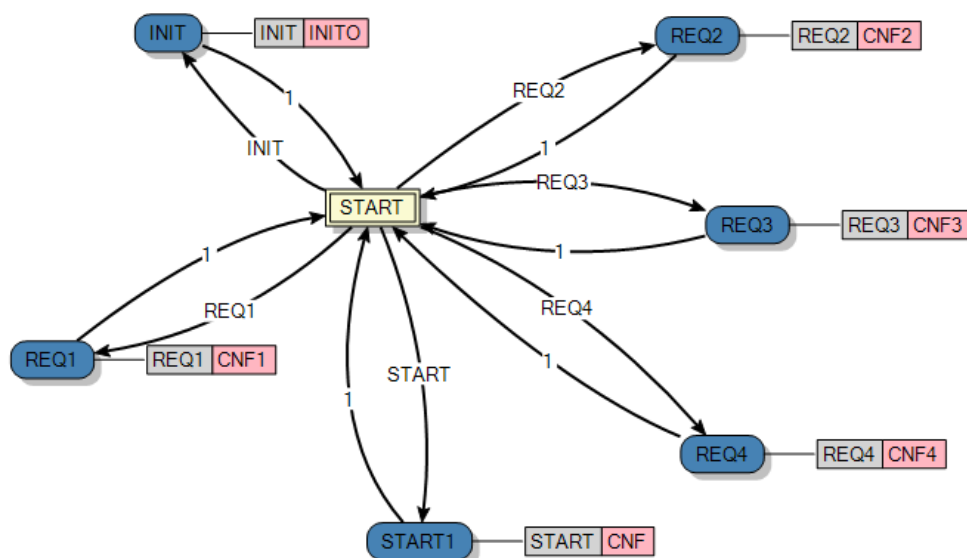


Рисунок 4.7 – ECC – диаграмма базисного блока

Блок имеет следующие EventInputs:

- INIT – инициализация блока. Значение POWER_INTERNAL устанавливается как FALSE, система готова к запуску.

```

ALGORITHM INIT IN ST:
(* Add your comment (as per IEC 61131-3) here
Initialization algorithm
*)
POWER_I := FALSE;
END_ALGORITHM
  
```

- START – запуск алгоритма тестирования детали. Значение POWER_INTERNAL устанавливается на противоположное. При срабатывании данного EventInput алгоритм или начинает работу, или прекращает ее. Значения всех выходных сигналов сбрасываются в начальные.

```

ALGORITHM POWER1 IN ST:
(* Add your comment (as per IEC 61131-3) here
*)
IF POWER_I = FALSE THEN
POWER_I := TRUE;
GRAB := FALSE;
DREL := FALSE;
TEST := FALSE;
SPIN := FALSE;
END_IF;
END_ALGORITHM
  
```

- REQ1 – происходит считывание данных со всех датчиков, анализ значений и передача выходных значений на устройства. Этот EventInput обеспечивает работу алгоритма тестирования заготовки.

```

ALGORITHM REQ1 IN ST:
(* Add your comment (as per IEC 61131-3) here
Normally executed algorithm
*)
;
IF POWER_I = TRUE THEN
  
```

```

IF PLACE = TRUE THEN
    GRAB :=TRUE;
    SPIN :=TRUE;
END_IF;
END_IF;
POWER := POWER_I;
END_ALGORITHM

```

- REQ2 – происходит считывание данных со всех датчиков, анализ значений и передача выходных значений на устройства. Этот EventInput обеспечивает работу алгоритма тестирования заготовки.

```

ALGORITHM REQ2 IN ST:
    (* Add your comment (as per IEC 61131-3) here
    *)
    ;
    IF POWER_I = TRUE THEN
        DREL := TRUE;
        IF DREL_OK = TRUE THEN
            SPIN:=TRUE;
        END_IF;
    END_IF;
END_ALGORITHM

```

- REQ3 – происходит считывание данных со всех датчиков, анализ значений и передача выходных значений на устройства. Этот EventInput обеспечивает работу алгоритма тестирования заготовки.

```

ALGORITHM REQ3 IN ST:
    (* Add your comment (as per IEC 61131-3) here
    *)
    ;
    IF POWER_I = TRUE THEN
        TEST := TRUE;
        IF TEST_OK = TRUE THEN
            SPIN := TRUE;
        END_IF;
    END_IF;
END_ALGORITHM

```

- REQ4 – происходит считывание данных со всех датчиков, анализ значений и передача выходных значений на устройства. Этот EventInput обеспечивает работу алгоритма тестирования заготовки.

```

ALGORITHM REQ4 IN ST:
    (* Add your comment (as per IEC 61131-3) here
    *)
    ;
    IF POWER_I = TRUE THEN
        DROP := TRUE;
        GRAB := FALSE;
        IF PLACE = FALSE THEN
            SPIN := TRUE;
        END_IF;
    END_IF;
END_ALGORITHM

```

Также блок имеет следующие EventOutputs:

- INITO – последующая инициализация устройств в составе ПВУ.

• CNF, CNF1, CNF2, CNF3, CNF4 - EventOutput подающий сигнал на EventInput-входы датчиков в составе станции.

4.1.3 Тестирование и отладка базисного блока

Среда NXTStudio позволяет производить отладку базисных блоков.

Для начала нам нужно проинициализировать базисный блок через EventInput «INIT», тем самым установив значение POWER_I – FALSE. Это значит что система готова к запуску и ждет сигнала на EventInput «START».

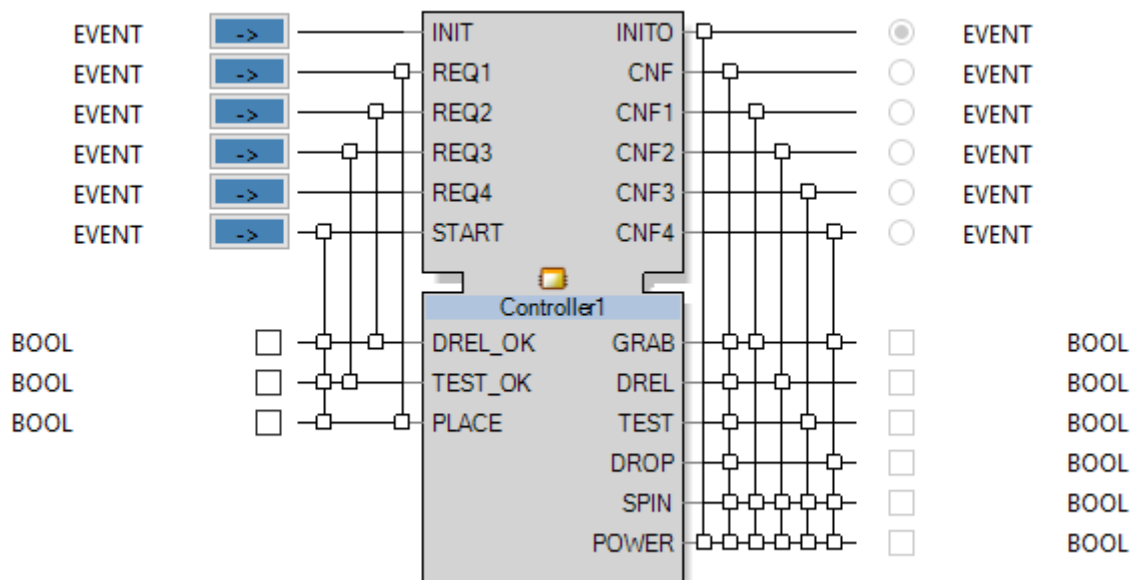


Рисунок 4.8 – Отладка базисного блока управления

Следующим шагом будет подача сигнала на EventInput «START», которая установит стартовые значения для всех устройств в составе станции и активирует EventOutput «CNF».

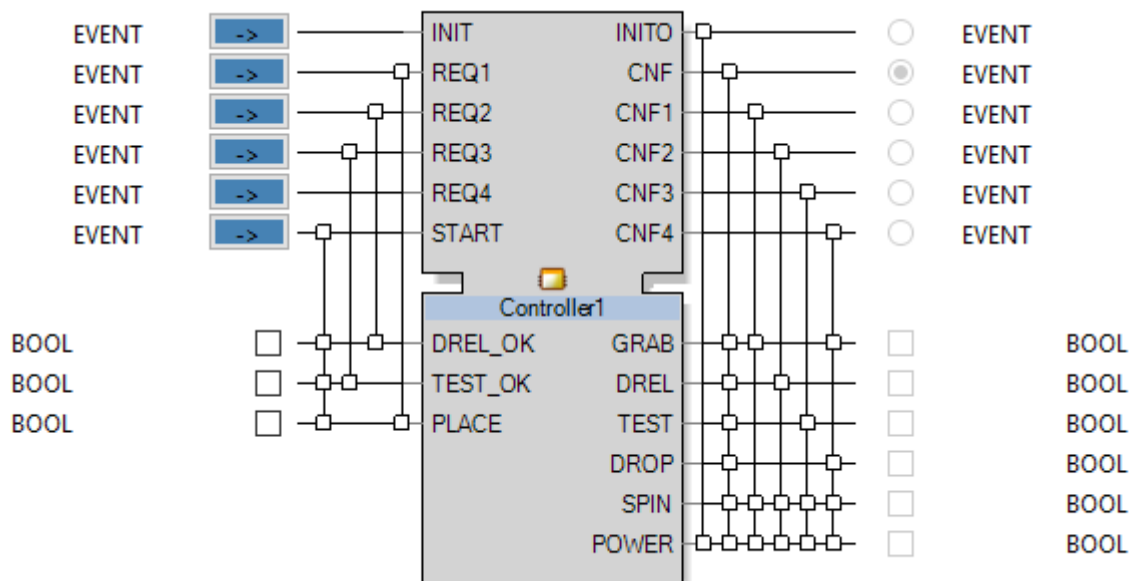


Рисунок 4.9 – Отладка базисного блока управления

Третьим шагом будет передача параметров с датчика «PLACE» через EventInput «REQ1», который запустит алгоритм базисного блока и в соответствии с входными параметрами выдаст на выходах соответствующие значения.

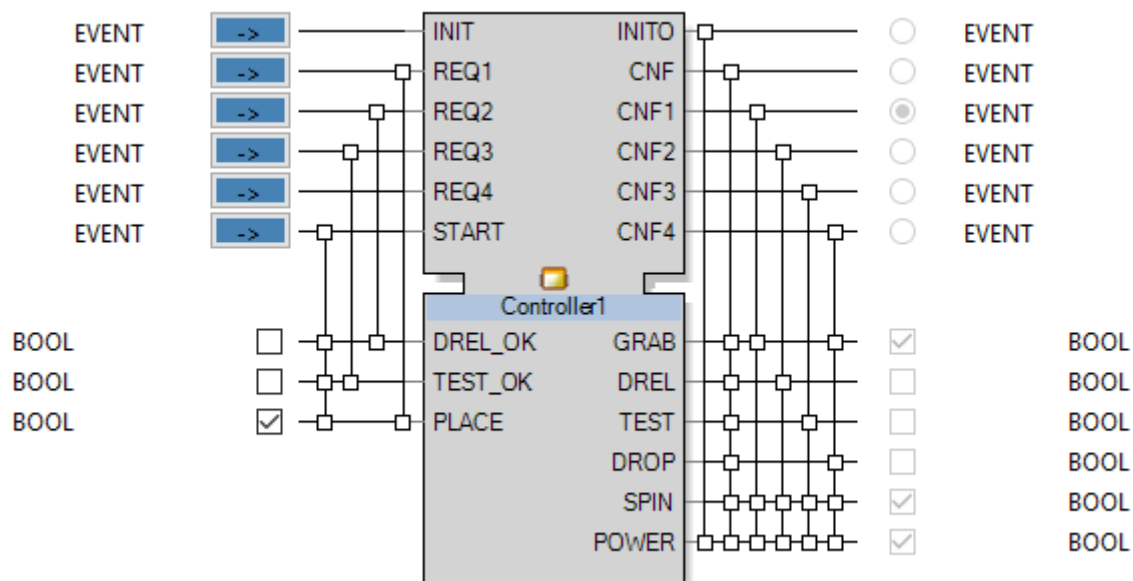


Рисунок 4.10 – Отладка базисного блока управления

Четвертым шагом будет передача параметров с датчика «DREL_OK» через EventInput «REQ2», который запустит алгоритм базисного блока и в соответствии с входными параметрами выдаст на выходах соответствующие значения.

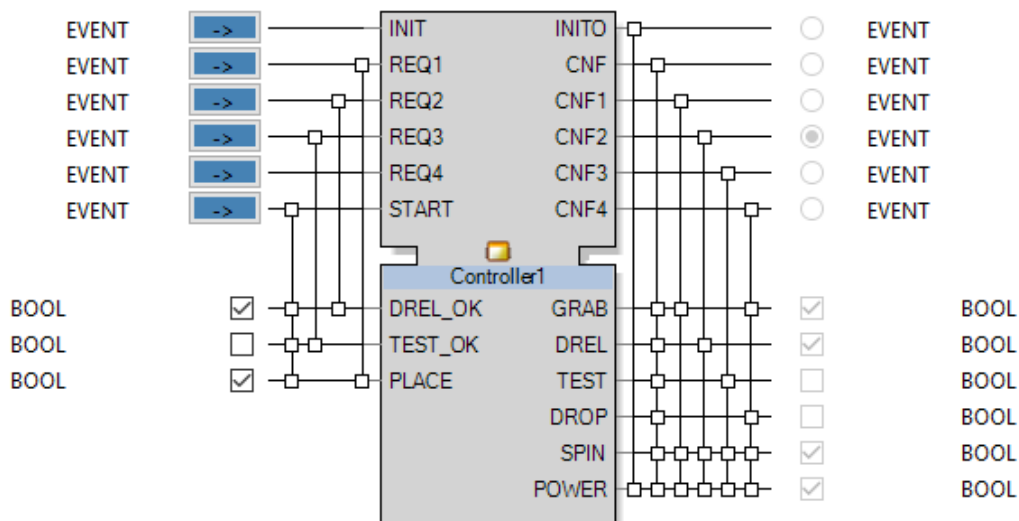


Рисунок 4.11 – Отладка базисного блока управления

Пятым шагом будет передача параметров с датчика «TEST_OK» через EventInput «REQ3», который запустит алгоритм базисного блока и в соответствии с входными параметрами выдаст на выходах соответствующие значения.

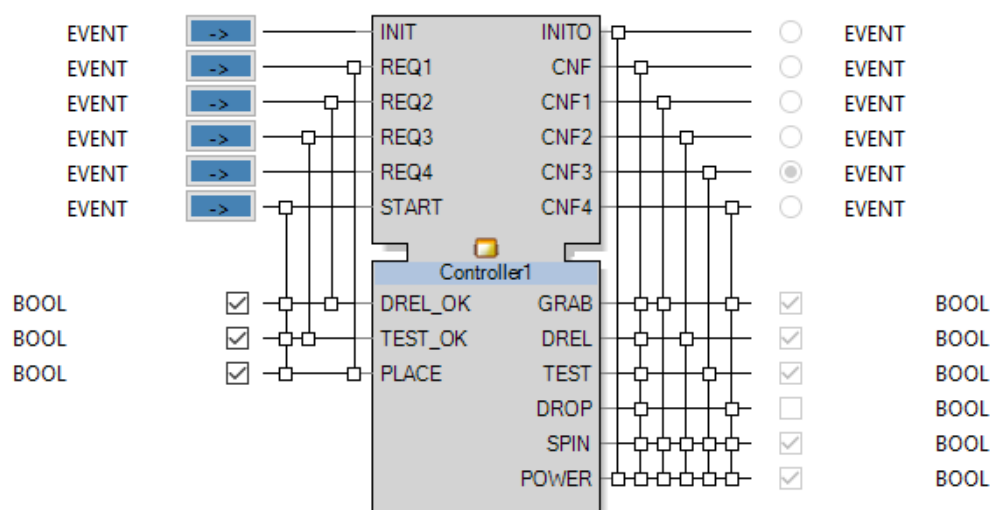


Рисунок 4.12 – Отладка базисного блока управления

Последним шагом будет передача параметров через EventInput «REQ4», который запустит алгоритм базисного блока и в соответствии с входными параметрами выдаст на выходах соответствующие значения.

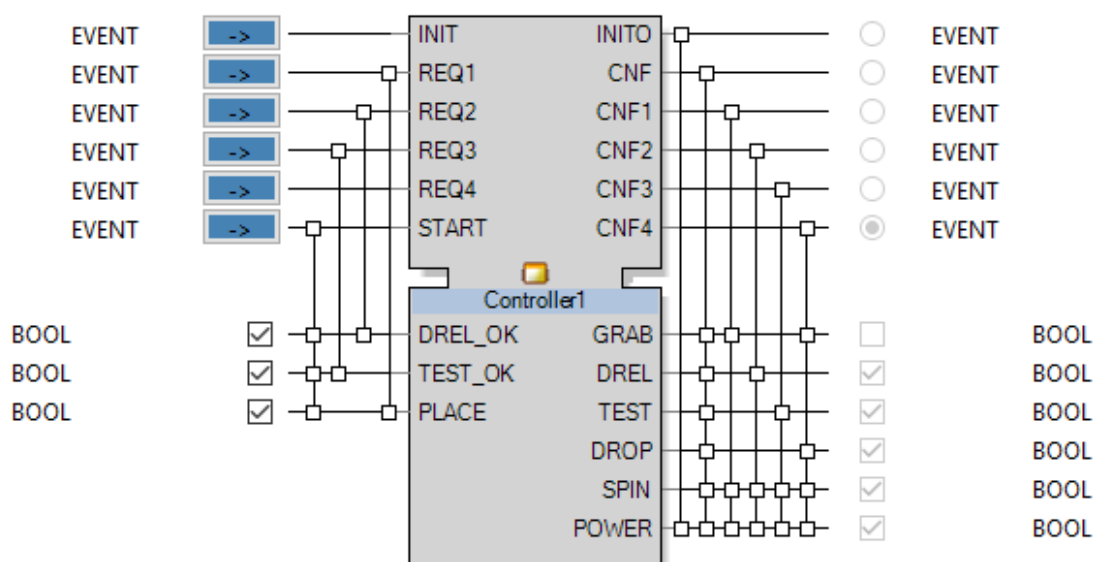


Рисунок 4.13 – Отладка базисного блока управления

4.2 Разработка составных и САТ-блоков тестирующей станции

4.2.1 Разработка САТ-блоков

Для создания САТ-блока нужно в среде NXTStudio 2.0 в «Solution Overview» необходимо найти папку «САТ», выбрать папку «Application», нажать правой кнопкой мыши и выбрать пункт «New Item»

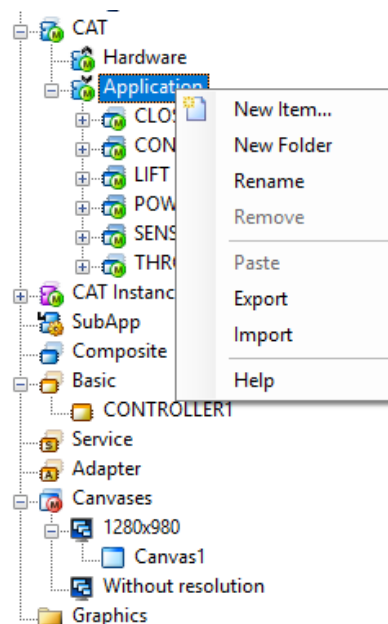


Рисунок 4.15 – Создание CAT-блока

В следующем окне нужно написать имя композитного блока и выбрать тип «Normal CAT»

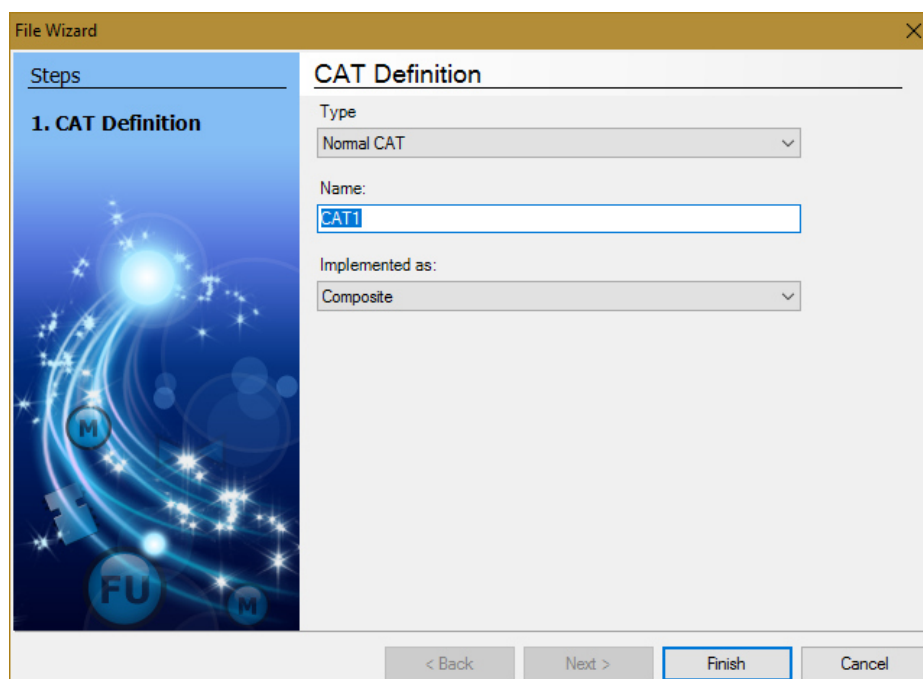


Рисунок 4.16 – Создание CAT-блока

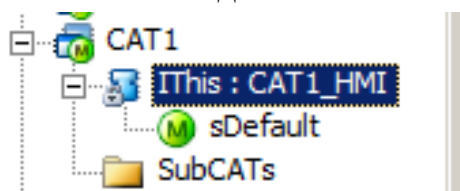


Рисунок 4.17 – Создание CAT-блока

На рисунке можно увидеть как выглядит CAT блок в «Solution View» в проекте. CAT1 – непосредственно сам композитный блок, IThis : CAT1_HMI –

блок визуализации HMI(Human Machine Interface), sDefault –графическое изображение блока визуализации.

Для каждого устройства станции необходимо создать композитный блок, который содержит входы и выходы, необходимые для ее работы.

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		48

4.2.2 Блок Test

Name	Type	Arra...	Initial v...	With
EventInputs				
INIT				TEST
REQ				TEST
<new interface>				
EventOutputs				
CNF				QO
<new interface>				
InputVars				
TEST	BOOL			
<new variable>				
OutputVars				
QO	BOOL			
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 4.18 –CAT-блок Test

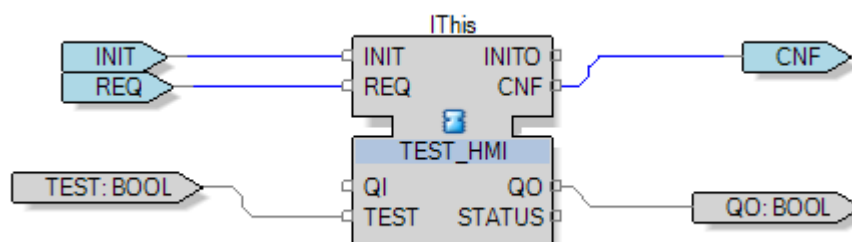


Рисунок 4.19 –CAT-блок Test

CAT-блок содержит 1 вход – Test(bool), EventInput REQ, INIT и блок визуализации. Значение Test может принимать 2 значения – TRUE и FALSE.

4.2.3 Блок Spin

Name	Type	Arra...	Initial v...	With
EventInputs				
INIT				QI
REQ				QI
<new interface>				
EventOutputs				
<new interface>				
InputVars				
QI	BOOL			
<new variable>				
OutputVars				
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 4.20 –CAT-блок Spin

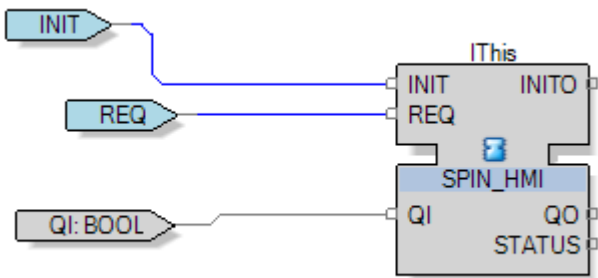


Рисунок 4.21 –CAT-блок Spin

CAT-блок содержит 1 вход – Q1(bool), EventInput REQ, INIT и блок визуализации. Значение Q1 может принимать 2 значения – TRUE и FALSE.

4.2.4 Блок Power button

Name	Type
EventInputs	
<new interface>	
EventOutputs	
CNF	
<new interface>	
InputVars	
<new variable>	
OutputVars	
<new variable>	
Sockets	
<new adapter>	
Plugs	
<new adapter>	
AdapterInputs	
<new adapter>	
AdapterOutputs	
<new adapter>	

Рисунок 4.22 –CAT-блок Power button

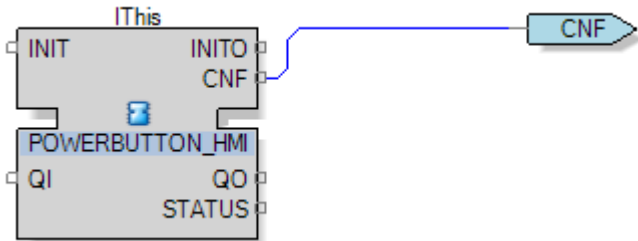


Рисунок 4.23 –CAT-блок Power button

Блок PowerButton не содержит входов и выходов, однако содержит EventOutput «CNF» который будет вызываться из блока визуализации при нажатии клавиши.

4.2.5 Блок Sensor test

Name	Type	Arra...	Initial v...	With
EventInputs				
REQ				QI
<new interface>				
EventOutputs				
CNF				QO
<new interface>				
InputVars				
QI	BOOL			
<new variable>				
OutputVars				
QO	BOOL			
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 4.24 –CAT-блок Sensor test

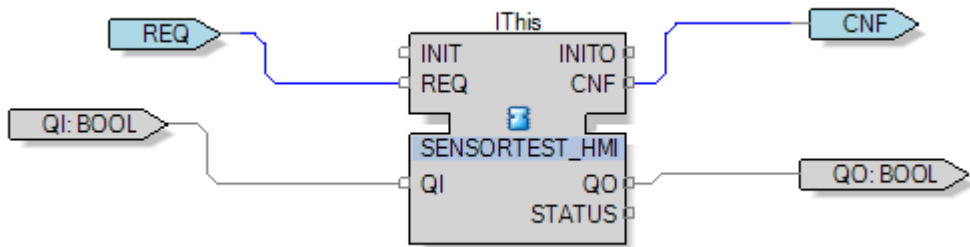


Рисунок 3.25 –CAT-блок Sensor test

CAT-блок содержит 1 вход – Q1(bool), EventInput REQ и блок визуализации. Значение Q1 может принимать 2 значения – TRUE и FALSE.

4.2.6 Блок Sensor place

Name	Type	Arra...	Initial v...	With
EventInputs				
REQ				
<new interface>				
EventOutputs				
<new interface>				
InputVars				
<new variable>				
OutputVars				
QO	BOOL			
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 4.26 –CAT-блок Sensor place

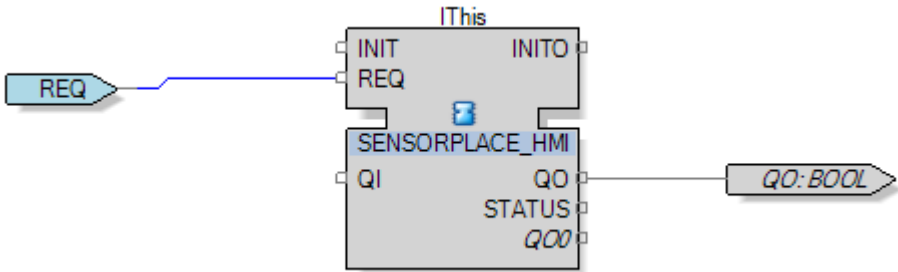


Рисунок 4.27 –CAT-блок Sensor place

Блок Sensor place содержит выход Q1, который передает значение.

4.2.7 Блок Grab

Name	Type	Arra...	Initial v...	With
EventInputs				
REQ				GRAB
<new interface>				
EventOutputs				
<new interface>				
InputVars				
GRAB	BOOL			
<new variable>				
OutputVars				
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 4.28 –CAT-блок Grab

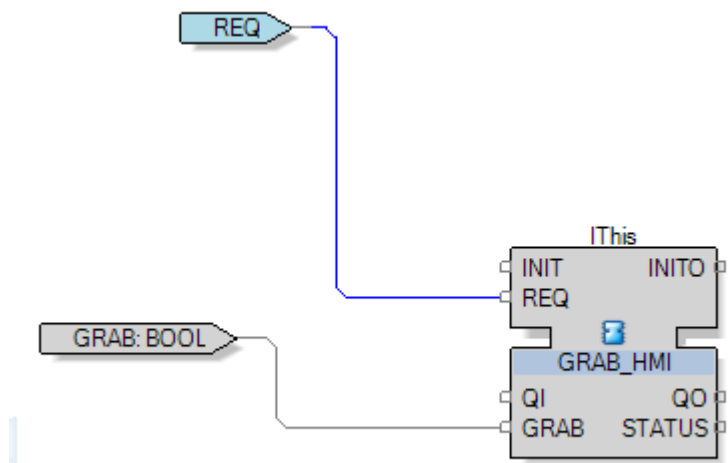


Рисунок 4.29 –CAT-блок Rrab

CAT-блок содержит 1 вход – GRAB(bool), EventInput REQ и блок визуализации. Значение Q1 может принимать 2 значения – TRUE и FALSE

4.2.8 Блок Drop

Name	Type	Arra...	Initial v...	With
EventInputs				
INIT				QI
REQ				QI
<new interface>				
EventOutputs				
<new interface>				
InputVars				
QI	BOOL			
<new variable>				
OutputVars				
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 4.30 –CAT-блок Drop

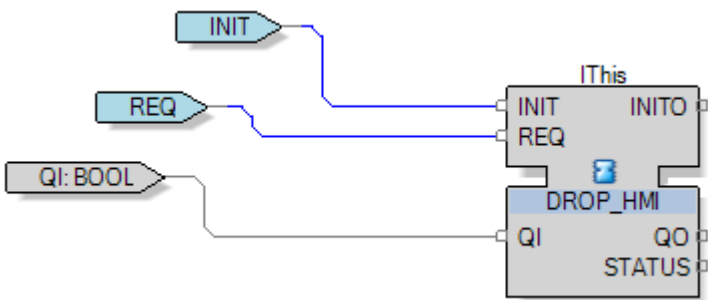


Рисунок 4.31 –CAT-блок Drop

CAT-блок содержит 1 вход – Q1(bool), EventInput REQ, INIT и блок визуализации. Значение Q1 может принимать 2 значения – TRUE и FALSE.

4.2.9 Блок Drel

Name	Type	Arra...	Initial v...	With
EventInputs				
INIT				
REQ				QI
<new interface>				
EventOutputs				
CNF				QO
<new interface>				
InputVars				
QI	BOOL			
<new variable>				
OutputVars				
QO	BOOL			
<new variable>				
Sockets				
<new adapter>				
Plugs				
<new adapter>				
AdapterInputs				
<new adapter>				
AdapterOutputs				
<new adapter>				

Рисунок 4.32 –CAT-блок Drel

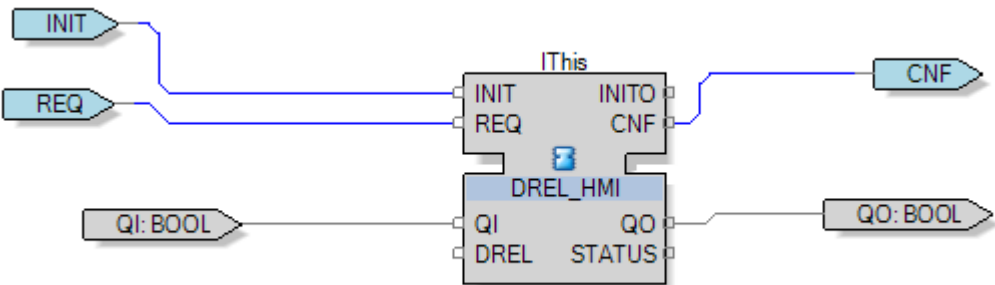


Рисунок 4.33 –CAT-блок Drel

CAT-блок содержит 1 вход – Q1(bool), EventInput REQ, INIT и блок визуализации. Значение Q1 может принимать 2 значения – TRUE и FALSE.

4.2.10 Блок Controller

Name	Type	Arra...	Initial v...	With	
EventInputs					
INIT					
REQ1				PLACE	
REQ2				DREL_OK	
REQ3				TEST_OK	
REQ4					
START				DREL_OK, TES...	
<new interface>					
EventOutputs					
INITO					
CNF				GRAB, DREL, T...	
CNF1				GRAB, SPIN	
CNF2				DREL, SPIN	
CNF3				TEST, SPIN	
CNF4				GRAB, DROP, S...	
<new interface>					
InputVars					
DREL_OK	BOOL				
TEST_OK	BOOL				
PLACE	BOOL				
<new variable>					
OutputVars					
GRAB	BOOL				
DREL	BOOL				
TEST	BOOL				
DROP	BOOL				
SPIN	BOOL				
<new variable>					

Рисунок 4.34 –САТ-блок Controller

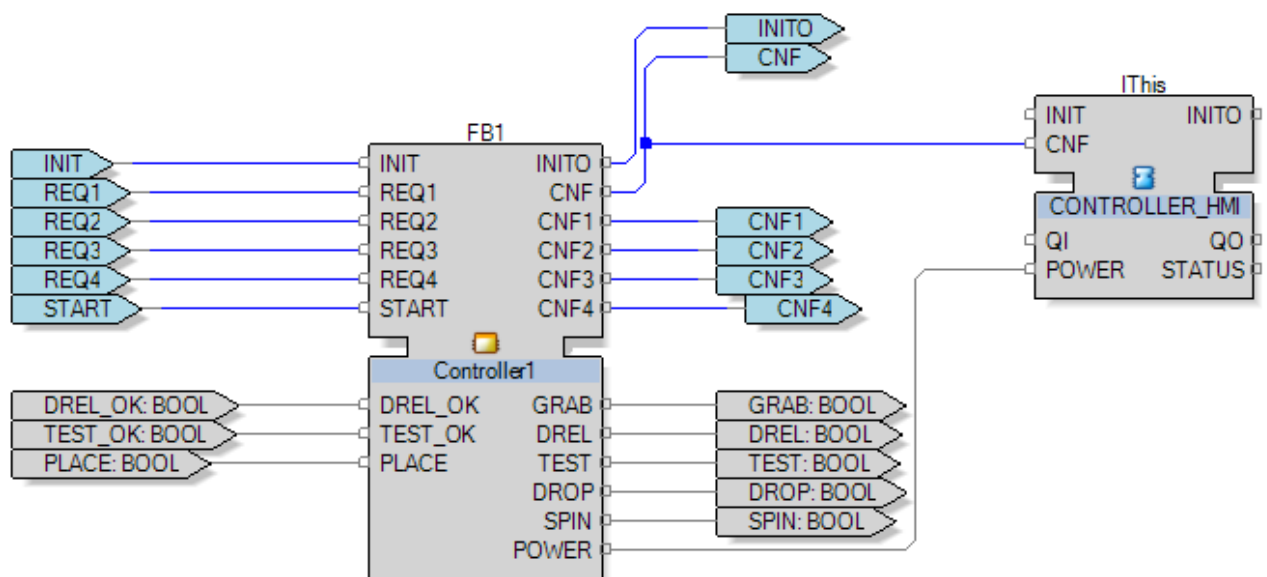


Рисунок 4.35 – CAT-блок Controller

Блок Controller – основной композитный блок, в котором содержится базисный блок Controller1 с алгоритмом работы. Композитный блок Controller копирует входы и выходы базисного блока Controller1.

4.3 Разработка приложения

Система композитных блоков содержится в файле приложения «APP1».

Для переноса компонентов в приложение необходимо создать CAT Instance Application на основе ранее созданных композитных блоков.

Для создания CAT Instance Application необходимо в Solution View проекта найти папку «CAT Instance» и протаскивать объекты от композитных блоков.

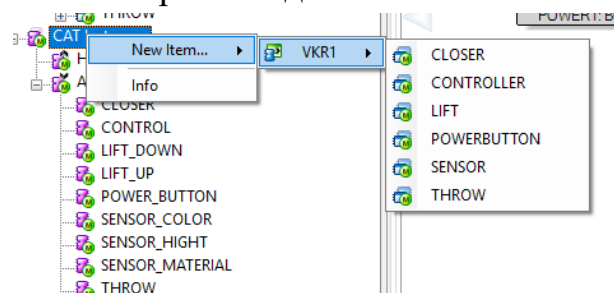


Рисунок 4.37 – Создание CAT Instance Application

Можно протаскивать несколько CAT Instance Application от одного CAT-блока. Таким образом для создания однотипных объектов можно создать один CAT-блок.

Исходя из описания компонентов ПВУ были созданы необходимые CAT Instance Application.

Папка CAT Instance Application стала выглядеть следующим образом:

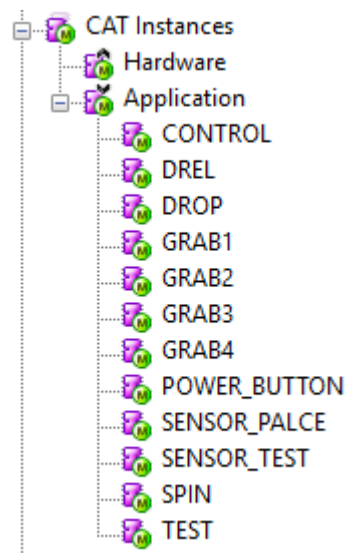


Рисунок 4.38 – Создание CAT Instance Application

В дальнейшем переносим объекты из папки Application в файл приложения «System.sys > APP» и организуем связи входов, выходов и событий. В конечном итоге файл «System.sys > APP» стал выглядеть следующим образом:

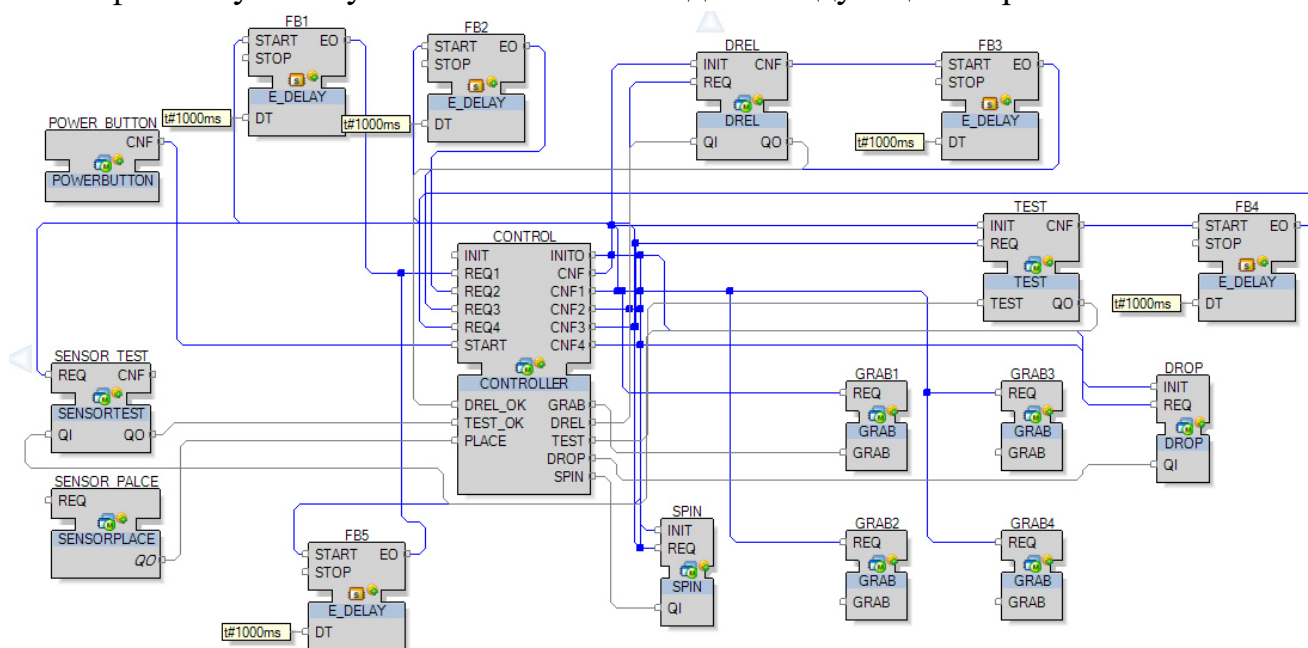


Рисунок 4.39 – Приложение System.sys > APP

4.4 Разработка визуальной модели

Элементы НМИ(Human Machine Interface) – элементы в составе композитных блоков, позволяющие организовать простой и понятный интерфейс пользователя, а также ввод и вывод нужных нам параметров на экран.

Элементы НМИ имеют возможность создания графического интерфейса блоков на языке C#.

При создании каждого композитного блока автоматически создается блок НМИ, который нам необходимо подключить к соответствующим входам и

выходам. Блоки НМІ имеют возможность показа параметров на экране а также передачу параметров через элементы интерфейса.

Элементы интерфейса компонуются нужным образом на файле типа «*.cnv» и создают интерфейс для работы пользователя.

Для начала создадим композитный блок и выберем файл визуализации. Для визуализации какого-либо параметра необходимо перетащить мышкой нужную нам переменную на холст.

В зависимости от типа переменной появится список возможных элементов отображения переменной.

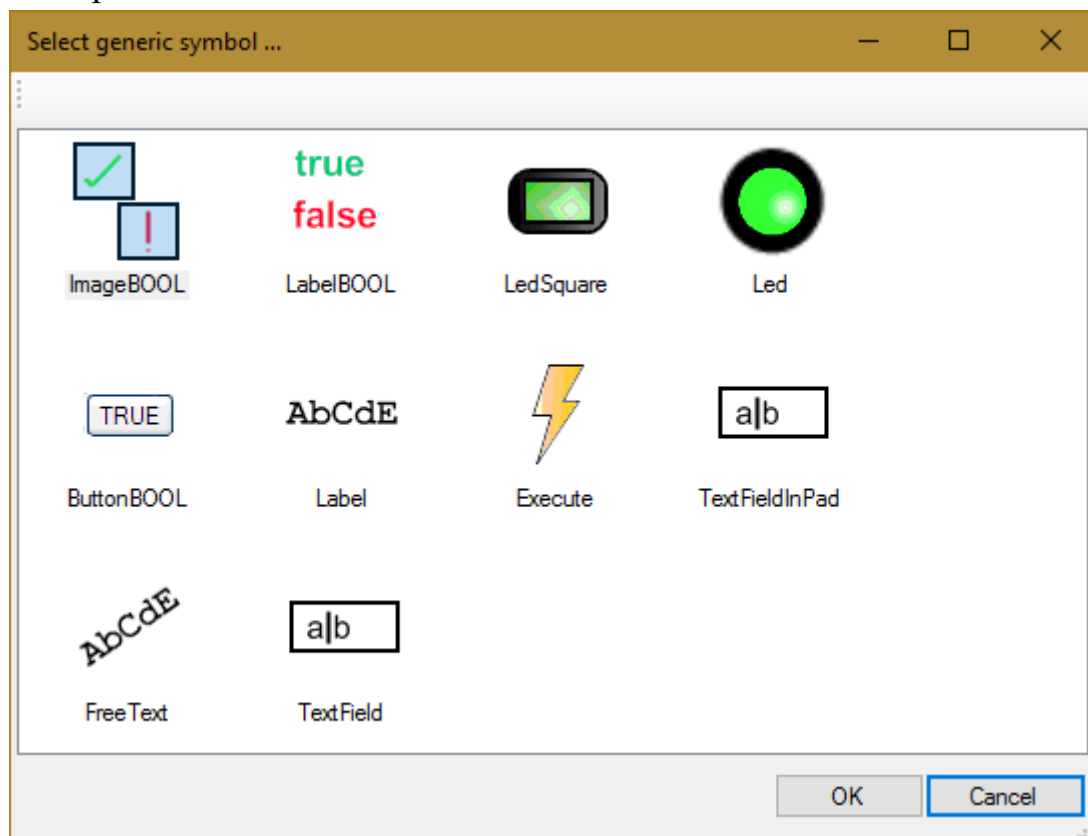


Рисунок 4.40 – Создание НМІ

Таким образом создадим элементы визуализации НМІ для каждого композитного блока:

- дрель

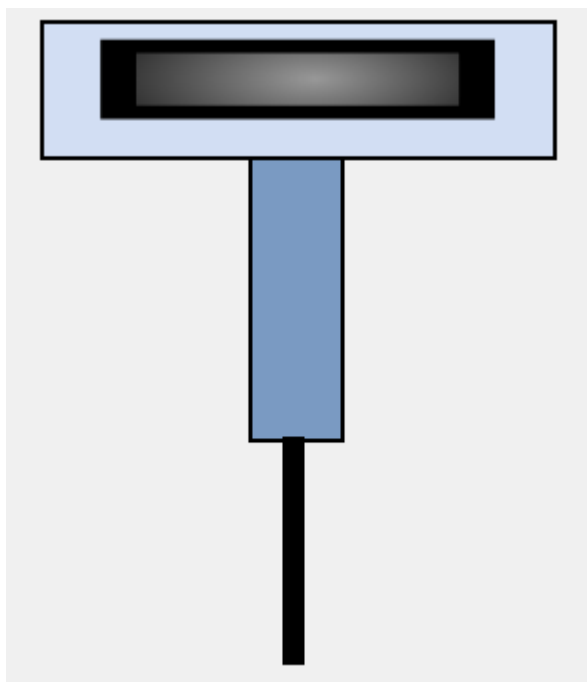


Рисунок 4.41 – Создание HMI

LED-индикатор будет цветом отображать состояние устройства. Зеленый – работает, красный – стоит.

- индикатор сброса

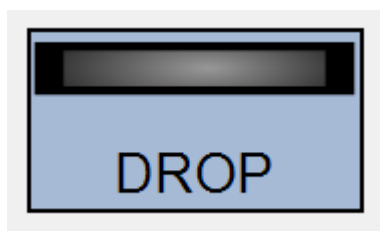


Рисунок 4.42 – Создание HMI

LED-индикатор будет цветом отображать сброс детали в случае готовности или брака.

- датчик захвата детали



Рисунок 4.43 – Создание HMI

LED-индикатор будет цветом захват детали в случае на столе.

- датчик индикации теста



Рисунок 4.44 – Создание HMI

LED-индикатор будет цветом успешность тестирования детали после ее обработки.

- тестер

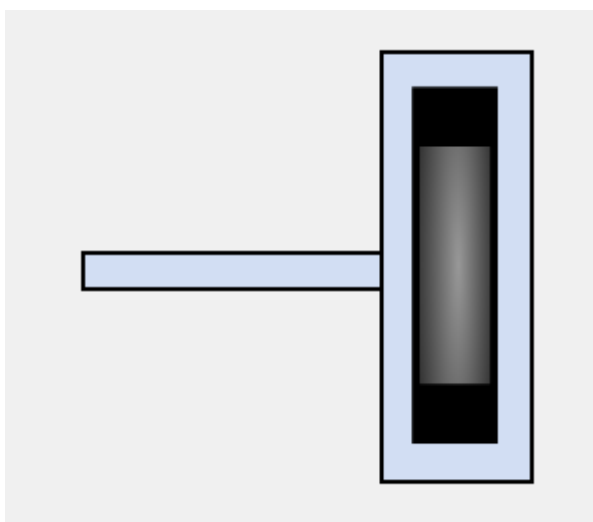


Рисунок 4.45 – Создание HMI

LED-индикатор будет цветом отображать состояние устройства. Зеленый – работает, красный – стоит.

- кнопка включения

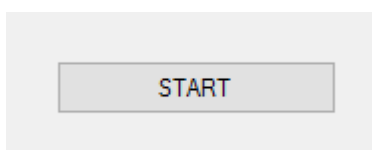


Рисунок 4.46 – Создание HMI

Кнопка включения при нажатии на нее будет вызывать EventOutput.

4.5 Визуализация работы

Для создания пользовательского интерфейса необходимо скомпоновать все графические представления на холсте. Сначала нужно создать холст, на котором будут отображаться необходимые нам индикаторы и элементы управления.

Для этого в Solution Overview найдем папку Canvases, выберем или создадим нужное нам разрешение, и в контекстном меню выберем пункт «New Item».

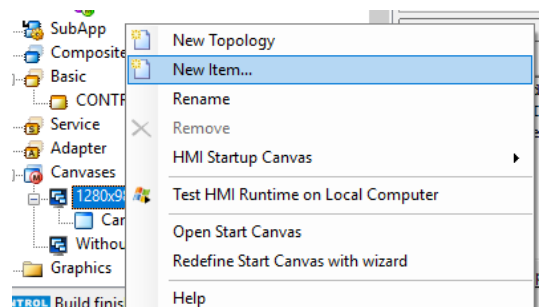


Рисунок 4.47 – Создание HMI. Добавление холста

Откроем полученный холст (Canvas) и перенесем наши блоки из папки SAT Instances : Application на холст. Скомпоуем все нужным образом и получим следующее представление:

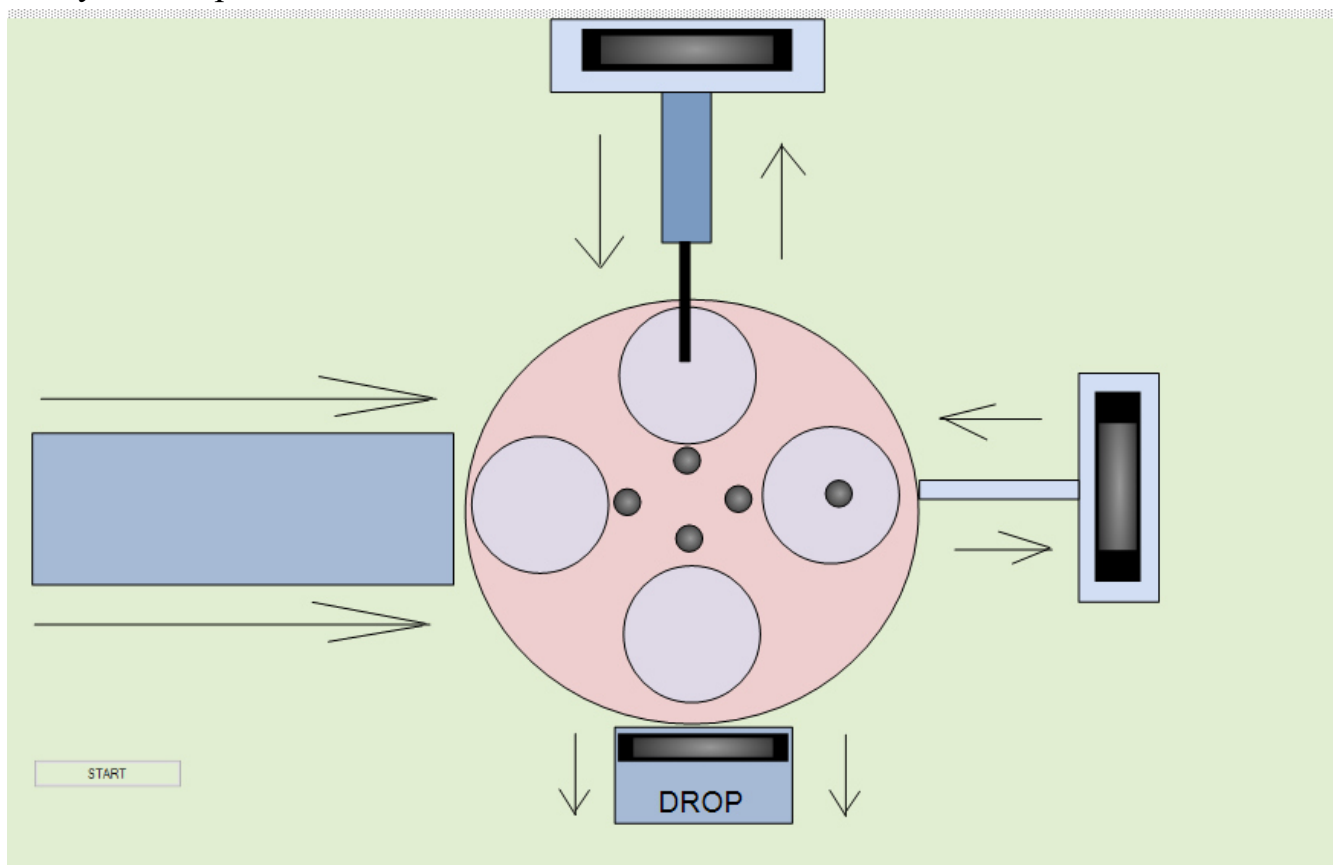


Рисунок 4.48 – Создание HMI

Интерфейс пользователя включает в себя визуализацию каждого устройства в составе тестирующей станции.

4.6 Запуск приложения в среде NXTStudio 2.0

Конечным шагом в разработке системы управления ПВУ в среде NXTStudio будет запуск приложения.

Перед началом работы следует сохранить все открытые файлы, иначе следующие шаги не будут возможными. Для запуска нашего приложения необходимо открыть

файл System.sys и перейти на вкладку Devices. Далее следует выбрать Active Network Profile и из выпадающего списка выбрать «Local Test». Далее для каждого устройства необходимо в поле «Login» выбрать пункт «Yes». Далее выделим все устройства и в контекстном меню выберем пункт «Start Soft PLC».

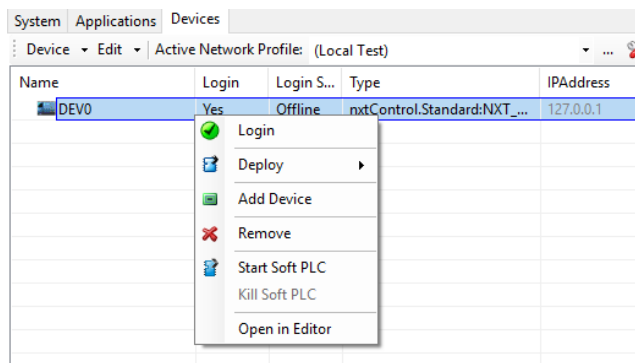


Рисунок 4.49 – Запуск приложения

После чего на компьютере будут запущены виртуальные PLC.

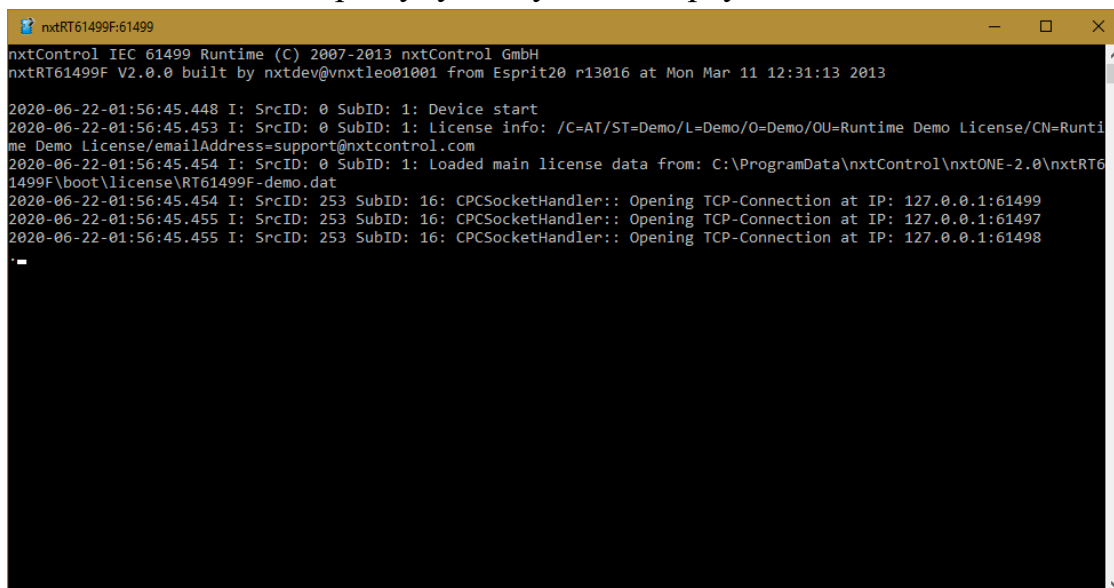


Рисунок 4.50 – Запуск приложения

Далее выберем в меню программы пункт Device > Deploy > Deploy Advanced.

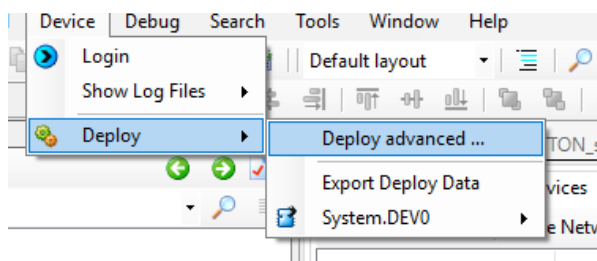


Рисунок 4.51 – Запуск приложения

После вышеописанных шагов выполнится Build проекта. В Открывшемся окошке нажимаем правой кнопкой мыши на устройстве из списка устройств и

выбираем пункт «Delete Persistent Data». Данную процедуру выполняем для всех устройств в проекте.

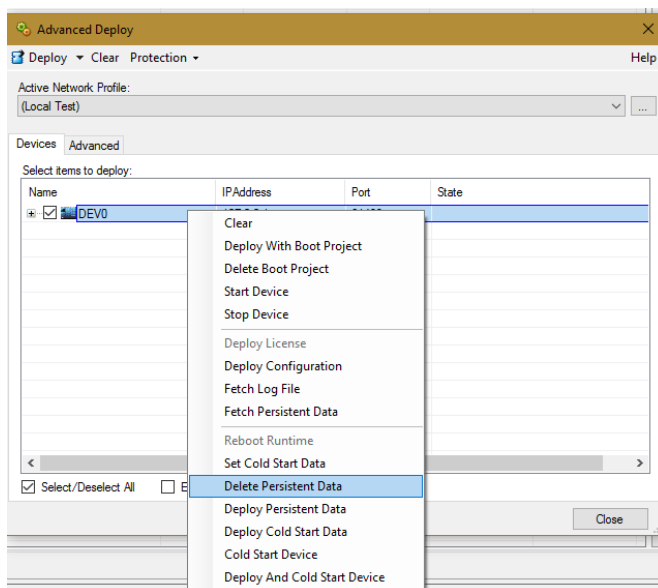


Рисунок 4.52 – Запуск приложения

Если все шаги были выполнены правильно, напротив каждого устройства в строке State появится запись «OK». Далее нужно выделить все устройства и нажать кнопку «Deploy».

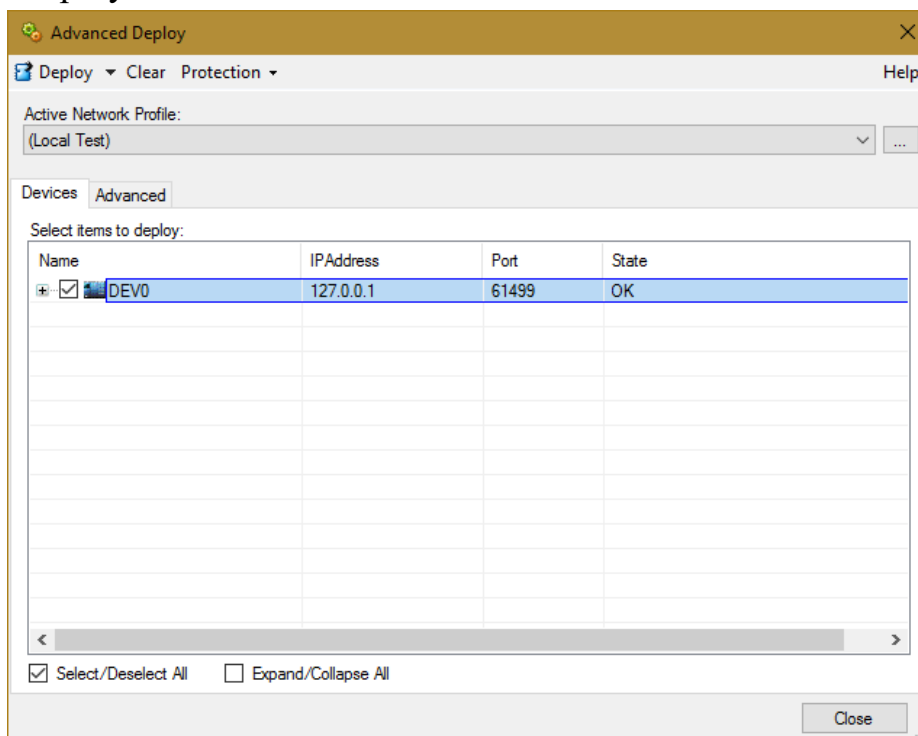


Рисунок 4.53 – Запуск приложения

После успешной операции «Deploy» данное окно должно автоматически закрыться. Теперь необходимо в файле System.sys > Devices нажать кнопку «Login». Название кнопки сменится на «Logout» а в графе Login State каждого из устройств появится запись «Online». После этого шага проект будет готов к запуску.

Device ▾ Edit ▾		Active Network Profile: (Local Test) ▾ ... ✂			Login	
Name	Login	Login S...	Type	IPAddress	Platform	R
DEVO	Yes	Offline	nxtControl.Standard:NXT_...	127.0.0.1	SOFT_PC	Login

Рисунок 4.54 – Запуск приложения

Device ▾ Edit ▾		Active Network Profile: (Local Test) ▾ ... ✂			Logout	
Name	Login	Login S...	Type	IPAddress	Platform	
DEVO	Yes	Online	nxtControl.Standard:NXT_...	127.0.0.1	SOFT_PC_W32	

Рисунок 4.55 – Запуск приложения

Теперь в Solution Overview находим папку «Canvases», находим холст с нужным нам разрешением, нажимаем правой кнопкой и выбираем пункт «Test HMI Runtime on Local Computer».

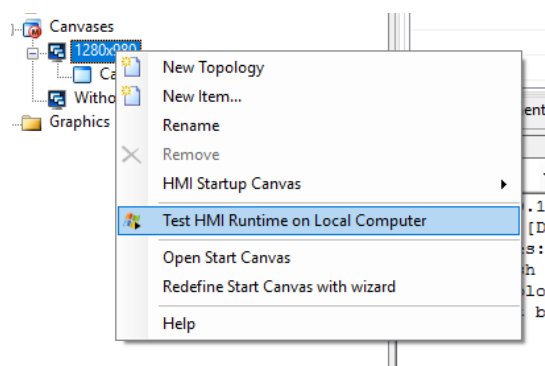


Рисунок 4.56 – Запуск приложения

Откроется холст с интерфейсом проекта.

4.7 Тестирование приложения

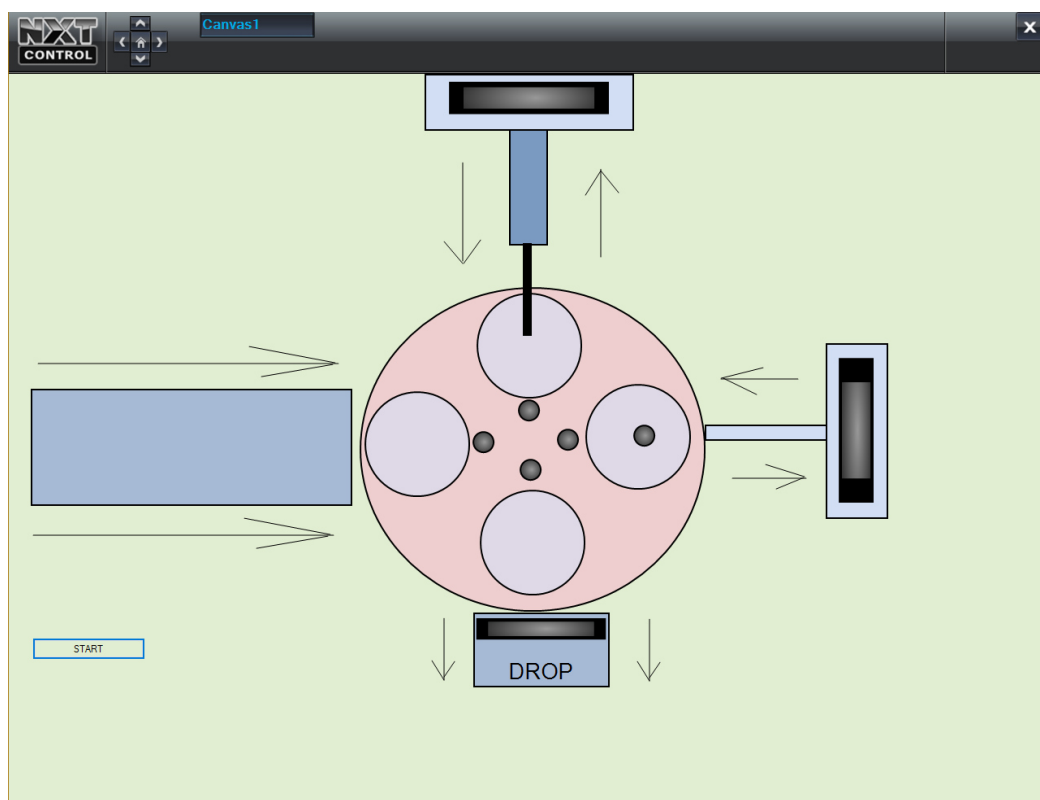


Рисунок 4.58 – Начальное состояние приложения

Начальным шагом будет зажим детали на столе.

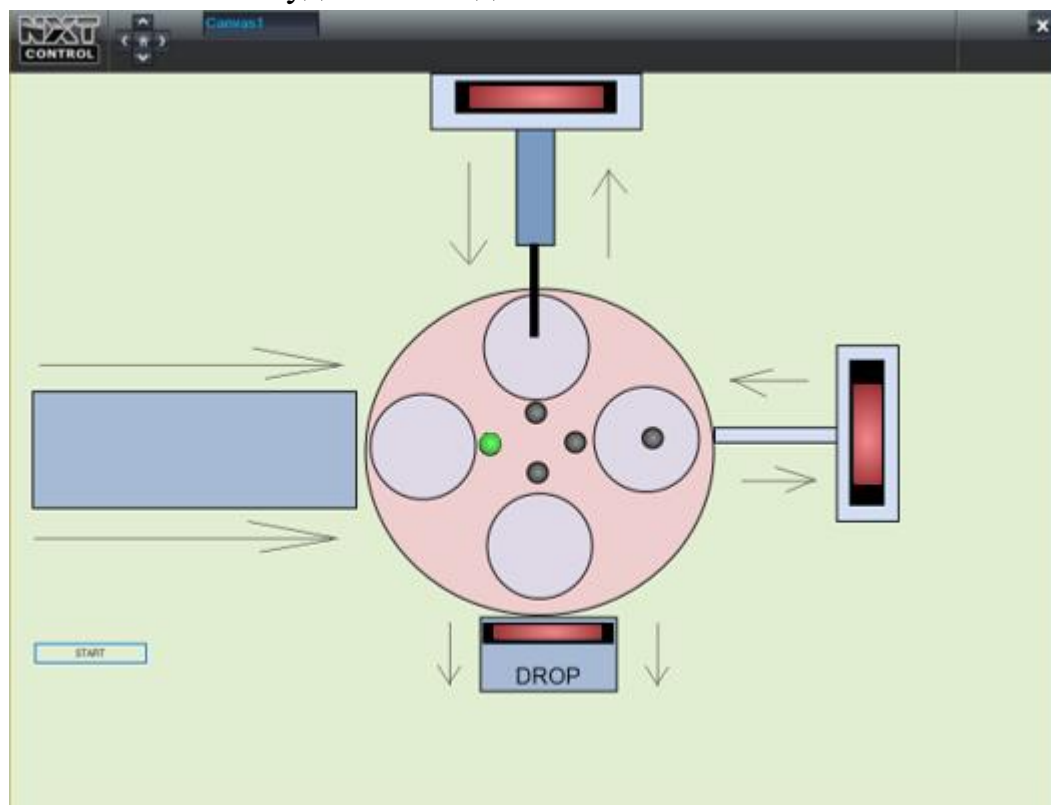


Рисунок 3.48 – Тестирование приложения

После она переходит к дрели для изготовления отверстия в детали.

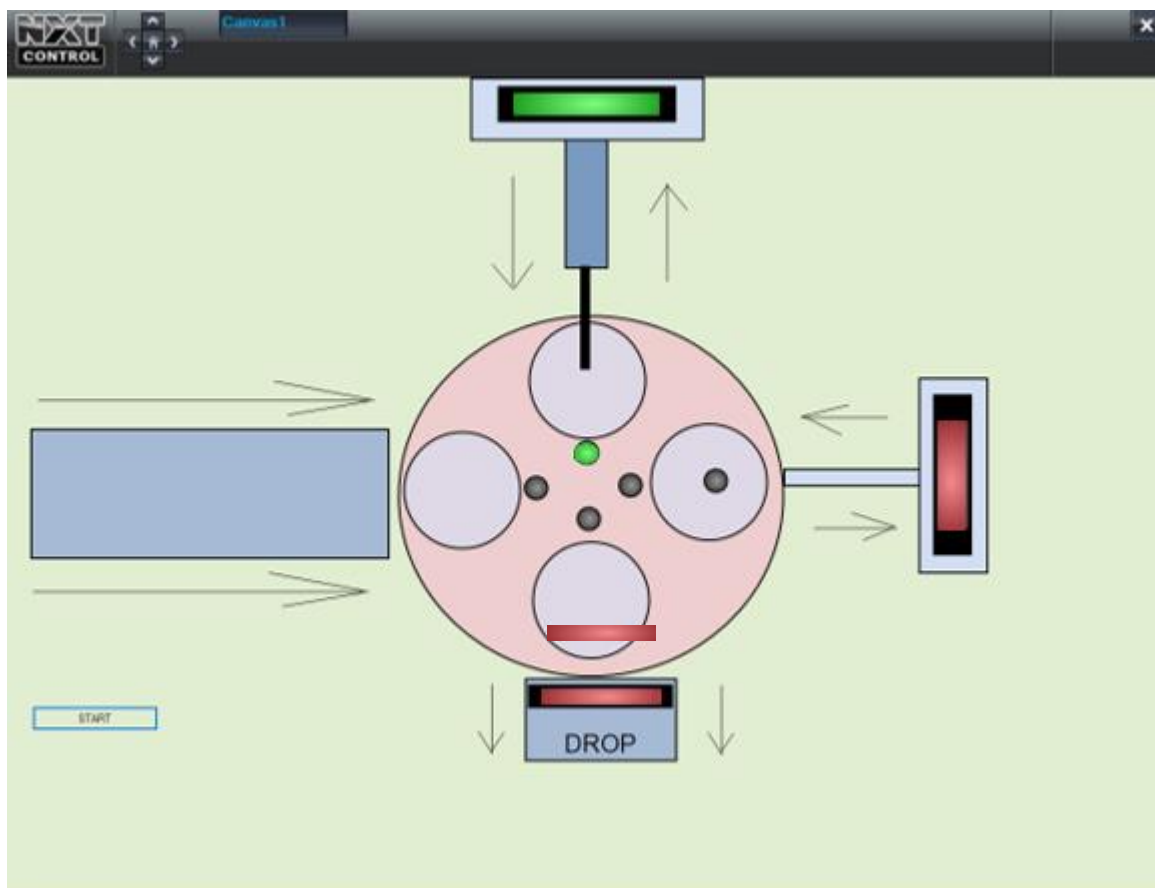


Рисунок 3.49 – Тестирование приложения
Следующим шагом будет проверка изготовленного отверстия

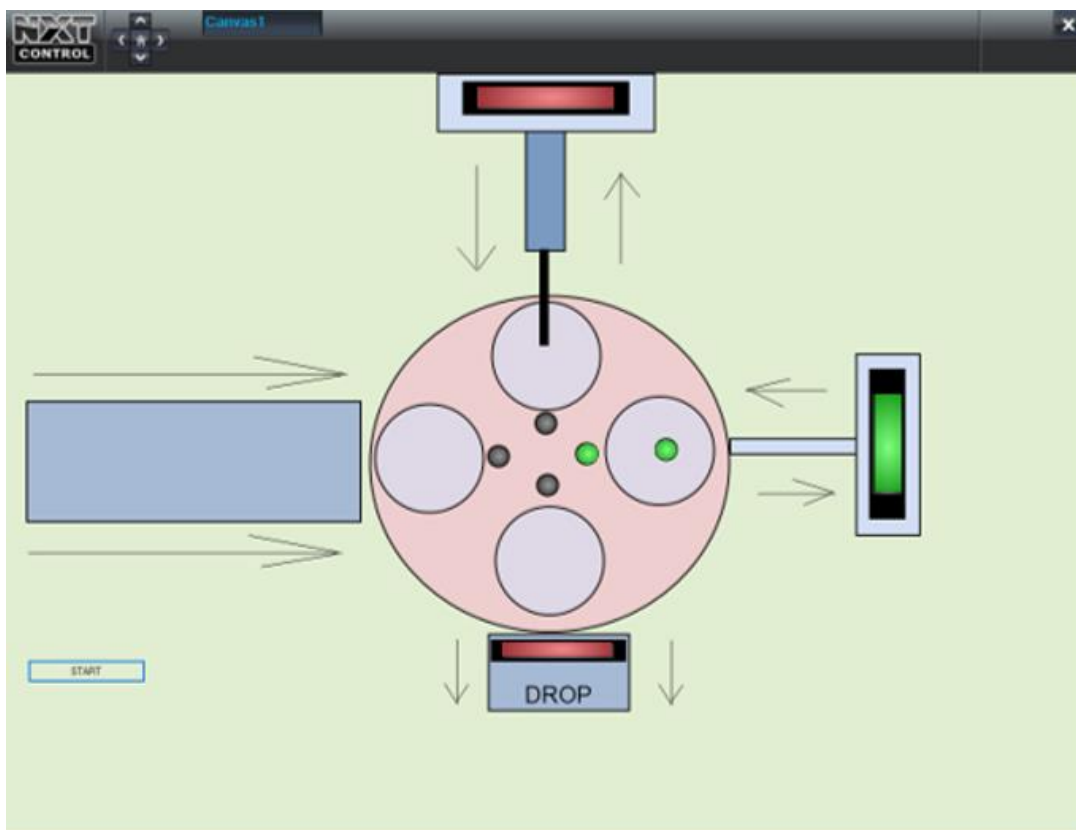


Рисунок 3.50 – Тестирование приложения

Последним шагом деталь передается на сброс для дальнейшего ее использования оператором.

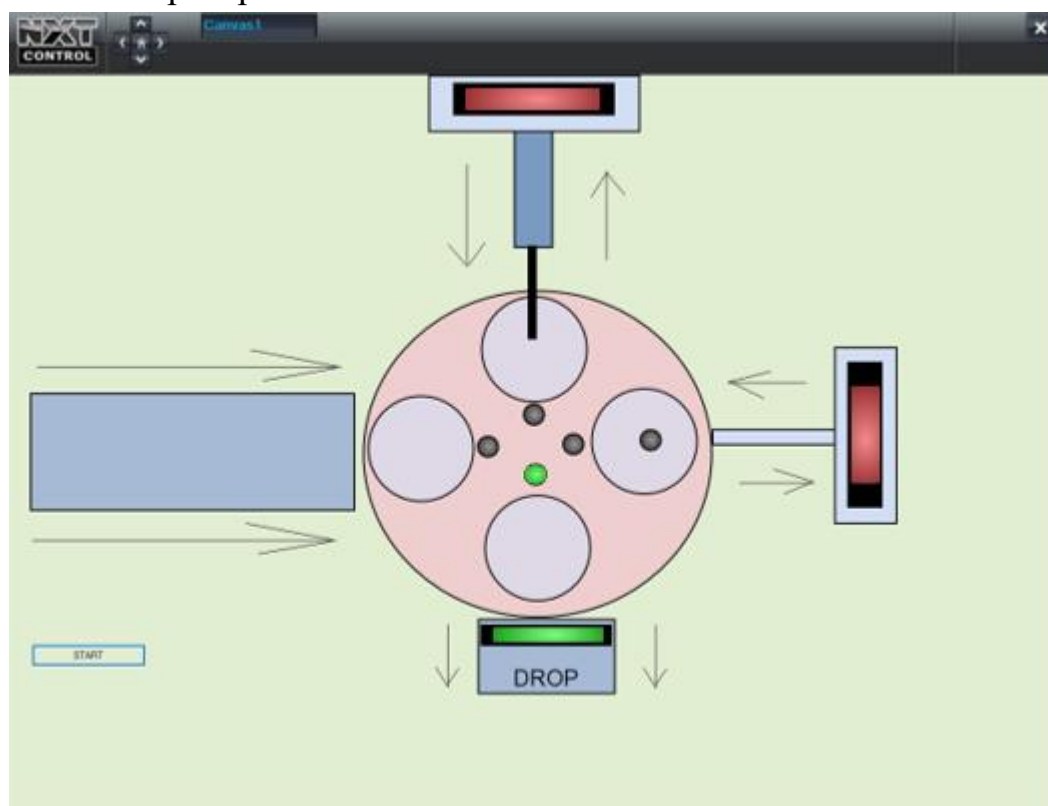


Рисунок 3.50 – Тестирование приложения

5 Освещение места работы инженера-разработчика

Под освещенностью понимают световую величину, равную отношению падающего на определенный участок поверхности светового потока к площади этого участка.

От правильного освещения зависит КПД работников, ведь свет влияет на все составляющие этого показателя:

- энергичность, выносливость, работоспособность;
- мотивацию работы;
- здоровье, хорошее физическое самочувствие.

Зрение людей при недостатке света неизбежно будет падать, возникнут хроническое переутомление и усталость, утратится заинтересованность в результатах работы. При избытке света глаза тоже устают, возникают головокружения, особенно при последующем попадании в помещение с менее яркими светильниками. Итогом становится раздражительность, плохое настроение, падение работоспособности, невнимательность сотрудников.

Согласно гигиеническому нормированию, основной показатель освещенности пространства измеряется в люксах (Лк). В некоторых нормативах единицей измерения выступает Люмен/квадратный метр площади. У офисных сотрудников, которые ежедневно работают за компьютером, постоянно повышенная зрительная нагрузка. Если свет в офисе будет неравномерным, станет падать на стол не с нужной стороны, возникнут проблемы со здоровьем. По нормам на столе за ПК световой поток обязан быть равным 300-500 Лк. Проведение необходимых замеров позволит быстро выявить отклонения в освещении.

Существуют разные классификации освещения. Так, по локализации оно бывает следующих типов:

Общее освещение. Предполагает равномерное освещение помещения без наличия темных или более светлых зон. Присутствие только такого освещения

обычно наблюдается в зонах, где рабочий процесс осуществляется неполное время.

Местное освещение. Локальная подсветка помогает дополнительно осветить определенные рабочие зоны: компьютерный или школьный стол, технику и станки. Оно предполагает установку различных осветительных приборов в непосредственной близости от рабочего места.

Применение исключительно локального освещения согласно нормам недопустимо, поскольку в помещении неизбежно будут присутствовать перепады света — от глубокого затемнения до яркого. Это вызовет проблемы со зрением работников. По источникам света освещение также делится на несколько видов.

Естественное освещение создается силами природы: прямыми солнечными лучами, а также диффузным (отраженным) светом небосвода. Отсутствие естественного света неблагоприятно для человека, ведь именно к нему глаза лучше всего приспособлены. Такой свет зависит от времени года и периода суток, в этом его основной недостаток. Но качество и объем поступающего естественного света зависит и от конструкции здания, количества и размера окон.

Без искусственного освещения в сумеречное время суток, в пасмурный день или зимой, когда рано темнеет, нормальный рабочий процесс невозможен. В качестве дополнительных источников света выступают лампы, светильники, торшеры, бра и прочие электроприборы. Обычно в офисы и на производство приобретают галогенные и светодиодные лампы. Обычные лампочки накаливания сейчас применяются редко, поскольку они тратят много электроэнергии, быстро выходят из строя. Чаще всего освещение бывает смешанным, когда естественное сочетается с искусственным. Последнее также подразделяется на следующие виды.

Рабочее. Обычное освещение, которое ежедневно применяется сотрудниками, помогает обеспечению рабочего процесса.

Аварийное. Включается только при аварии, в экстренной ситуации, когда основное освещение отключается.

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		72

Эвакуационное. Применяется для подсветки путей эвакуации людей при ЧС, обычно является не таким мощным, как рабочее.

Охранное. Используется охранным персоналом, присутствует не на всех предприятиях, а по необходимости. Не нормируется по интенсивности.

Дежурное. Остается включенным даже по окончании рабочего процесса (например, небольшое освещение коридоров в больших зданиях).

Чтобы грамотно нормировать освещение, нужно строго следовать СанПиН и иным нормативным документам. Все требования можно условно поделить на касающиеся собственно освещенности и относящиеся к качеству света. Общие для всех отраслей требования:

- освещенность достаточна для каждого человека;
- работник может регулировать освещение — направлять свет в нужную сторону, менять его интенсивность;
- кроме искусственного света обязательно присутствует естественный;
- все светильники на потолке, стенах должны быть приглушенного цвета, лампочки не вызывают зрительного дискомфорта.

Долгая, ежедневная работа с компьютером часто вызывает снижение зрения у сотрудников. Такие проблемы можно исключить или замедлить, соблюдая нормы и рекомендации:

- световой поток — 300 Лк;
- яркость элементов в осветительных приборах менее 200 кд/кв.м.;
- коэффициент естественного освещения (КЕО) — от 1,2%;
- расположение окна сбоку от места работы;
- наличие индивидуальных приборов освещения в комплексе с общими, если последних не достаточно;
- расположение локальных источников света справа от экрана;
- отсутствие бликов на мониторе;
- равномерное распределение луча света по столу и монитору.

Для выявления коэффициента освещенности нужно применить такую формулу: $КО = \frac{\text{Световой поток (Ватт)}}{\text{площадь помещения (кв.м.)}}$
 Пропорционально повышению площади снижается КПД потока света.

При оформлении интерьера в офисных помещениях важно учитывать цветовую гамму стен, ведь она по-разному воздействует на человека. Лучше всего выбирать пастельные тона, а также зеленоватые, желтые оттенки, приятные для глаз. Плохо воздействует на глаза свойство ярких поверхностей ухудшать контрастность и нарушать зрительный комфорт. Утомление глаз вызывают и колебания света, они сильно снижают производительность труда, поэтому тоже недопустимы.

Заключение

Согласно заданию, была разработана визуальная модель тестирующей и обрабатывающей станций установки FESTO на основе функциональных блоков IEC 61499. Была изучена и использована среда разработки NXTStudio 2.0.

Были разработаны визуальная модель тестирующей и обрабатывающей станций и их алгоритм работы в составе производственной установки FESTO.

Данную бакалаврскую работу можно использовать для обучения студентов работе со средой разработки NXTStudio 2.0.

Почти все поставленные задачи на дипломное проектирование можно считать выполненными.

Не решенной осталась задача с анимацией визуальной модели производственной системы. Я, как разработчик, столкнулся с проблемой банальной нехватки теоретических и практических навыков в этом вопросе. К сожалению, я не успел восполнить этот пробел в знания за время выполнения выпускной квалификационной работы.

Я считаю, что данное направление разработки очень перспективное и может очень сильно помочь промышленности при достаточных затратах сил и ресурсов на его развитие.

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		75

Список использованных источников

1. Модели функциональных блоков IEC 61499, их проверка трансформации в проектировании распределенных систем управления: моногр. / В. Н. Дубинин, В. В. Вяткин ; под ред. проф. Н. П. Вашкевича. – Пенза : Изд-во ПГУ, 2012 – 348 с.

2. V. Vyatkin - IEC 61499 FUNCTION BLOCKS FOR EMBEDDED AND DISTRIBUTED CONTROL SYSTEMS DESIGN with NxtStudio.// Luleå University of Technology, Sweden, 2012 г.

3. file:///FESTO_modelling_site/festo/descr_control_ru.htm#_2._Назначение_и

4. <https://www.nxtcontrol.com/en/>

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		76

ПРИЛОЖЕНИЕ А

(обязательное)

Микроконтроллерный измеритель расхода топлива

Презентация

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		77

ПГУ, ФВТ
Направление подготовки: 09.03.01
«Информатика и вычислительная техника»
Профиль подготовки:
«Вычислительные машины, комплексы,
системы и сети»

Визуальная имитационная модель производственной системы на основе функциональных блоков

Автор работы: студент группы 16ВВ1 Кукушкин А. Ю.

Руководитель: доцент каф. ВТ ПГУ Дубинин В. Н.

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		78

Цель работы

Целью данной выпускной квалификационной работы является разработка визуальной модели тестирующей и обрабатывающей станций производственной установки FESTO.

ПС FESTO предназначена для изготовления некоторых законченных деталей из поступающих на ее вход заготовок. Она состоит из трех станций: распределительной (Distribution Station), тестирующей (Testing Station) и обрабатывающей (Processing Station).

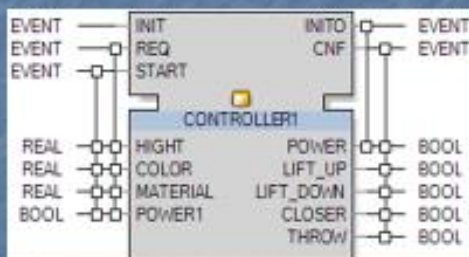
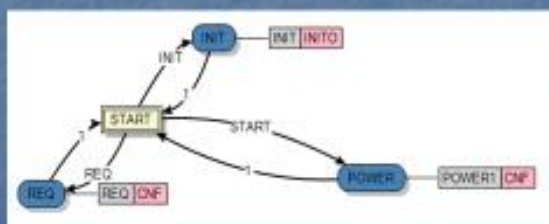
В ходе выполнения ВКР были рассмотрены следующие вопросы, подлежащие разработке и решены следующие задачи проектирования:

- анализ требований к разрабатываемой системе;
- разработка и реализация управляющих алгоритмов для тестирующей станции;
- разработка и реализация управляющих алгоритмов для обрабатывающей станции;
- отладка и тестирование системы ФБ тестирующей станции;
- отладка и тестирование системы ФБ обрабатывающей станции;
- разработка визуальной модели.

Производственная установка FESTO

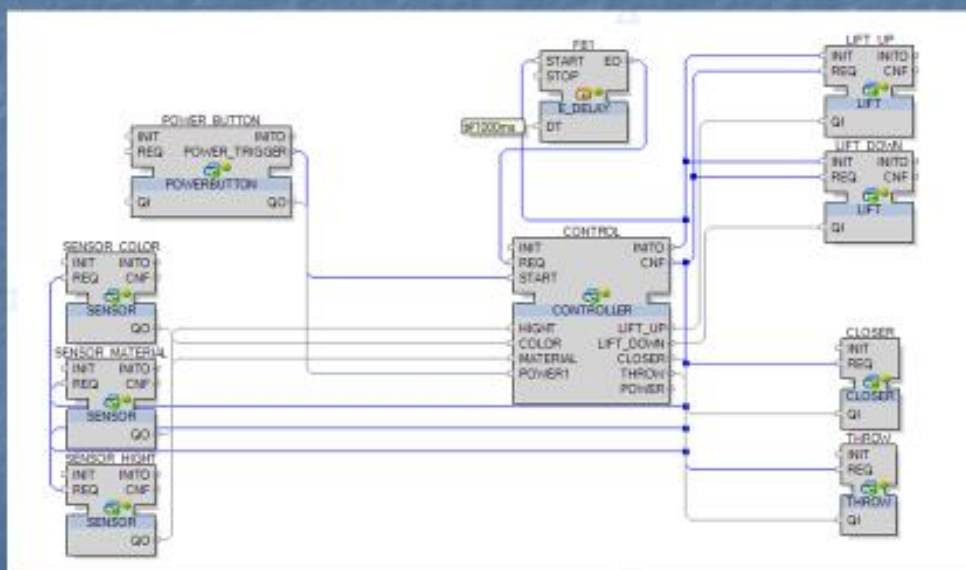


Разработка и реализация управляющих алгоритмов для тестирующей станции

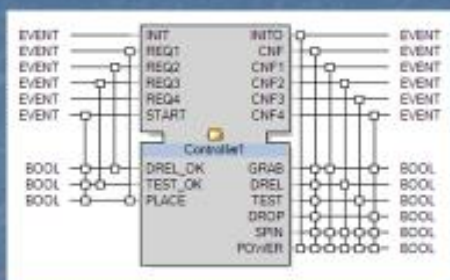


Interface : SCC Algorithms	
Name	Type
EventInputs	
INIT	
REQ	
START	
EventOutputs	
INITO	
CNF	
InputVars	
HIGHT	REAL
COLOR	REAL
MATERIAL	REAL
POWER1	BOOL
OutputVars	
POWER	BOOL
LIFT_UP	BOOL
LIFT_DOWN	BOOL
CLOSER	BOOL
THROW	BOOL
Sockets	
Plugs	
InternalVars	
POWER_INTERVAL	BOOL

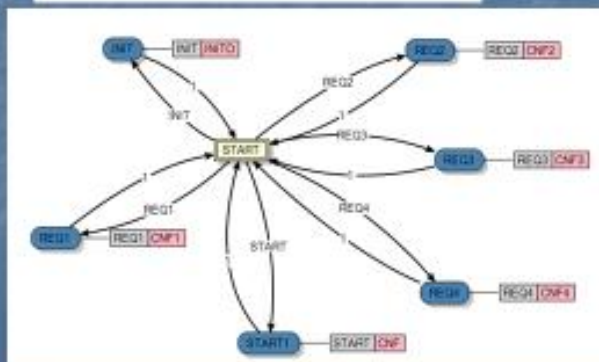
Связи между ФБ в тестирующей станции



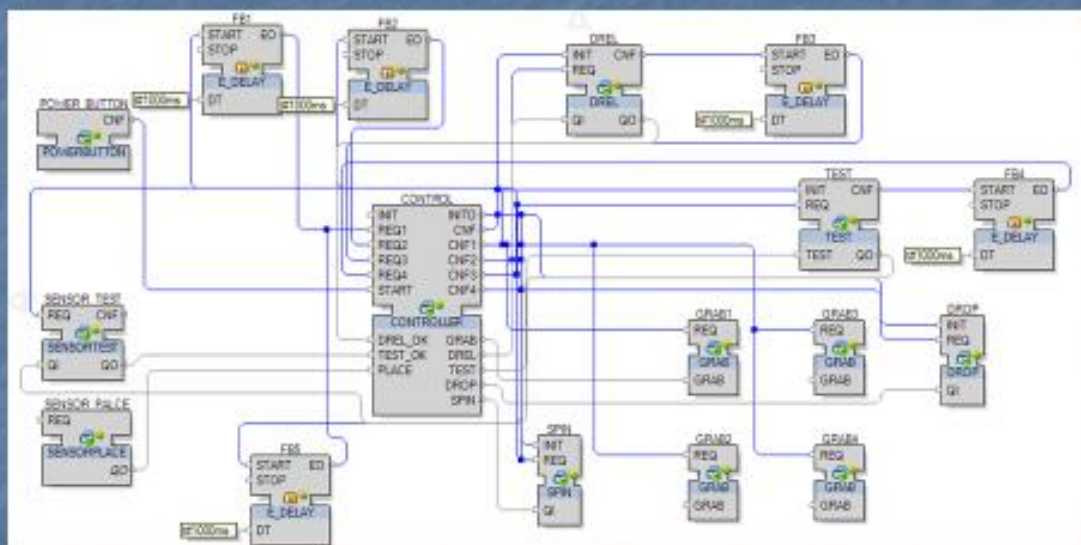
Разработка и реализация управляющих алгоритмов для обрабатывающей станции



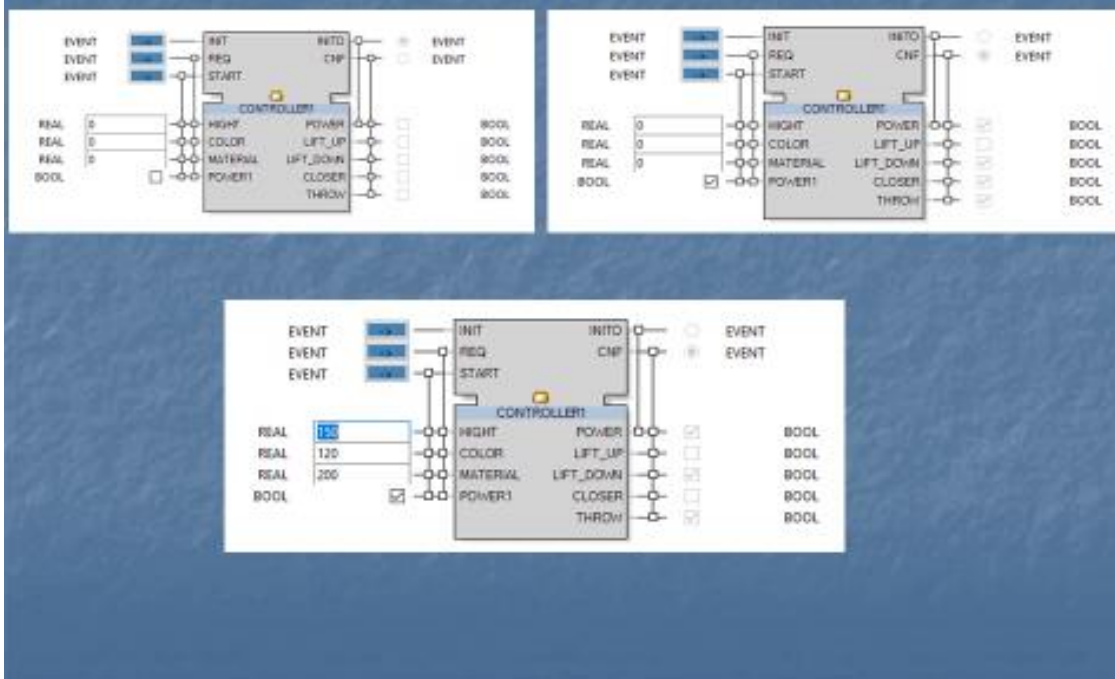
Name	Type	Area	Initial v.	SPM
EventInputs				
REQ1				
REQ2				
REQ3				
REQ4				
START				
EventOutputs				
REQ1				
REQ2				
REQ3				
REQ4				
START				
Inputs				
DREL_OK	BOOL			
TEST_OK	BOOL			
PLACE	BOOL			
Outputs				
POWER	BOOL			
DREL	BOOL			
TEST	BOOL			
DROP	BOOL			
SPIN	BOOL			
POWER	BOOL			



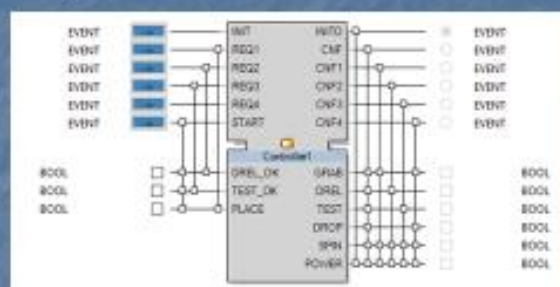
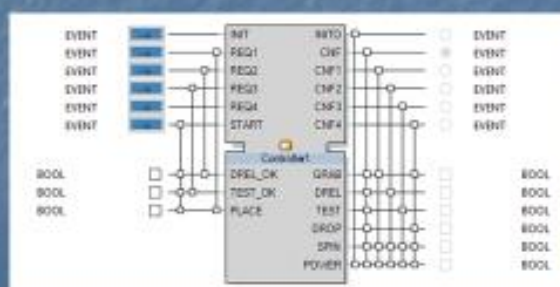
Связи между ФБ в обрабатывающей станции



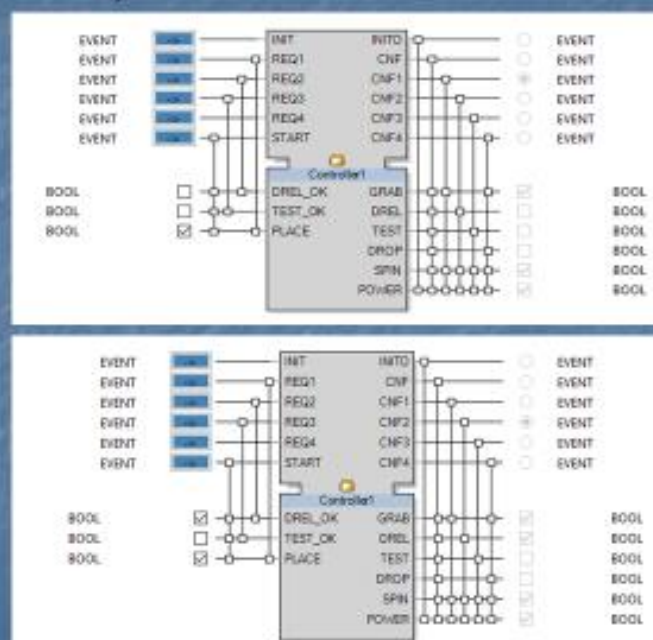
Отладка и тестирование системы ФБ тестирующей станции



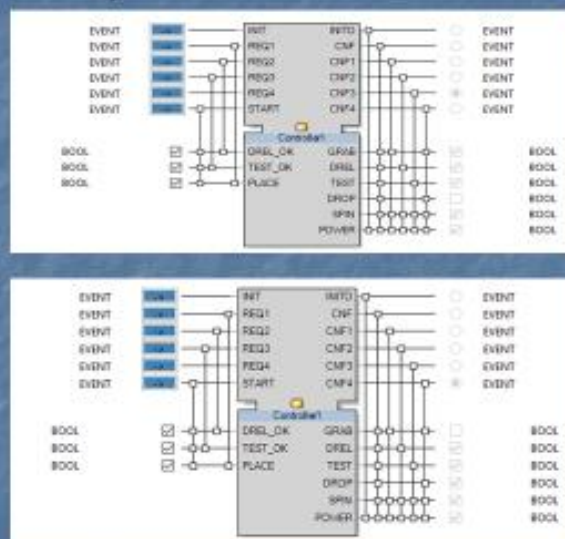
Отладка и тестирование системы ФБ обрабатывающей станции



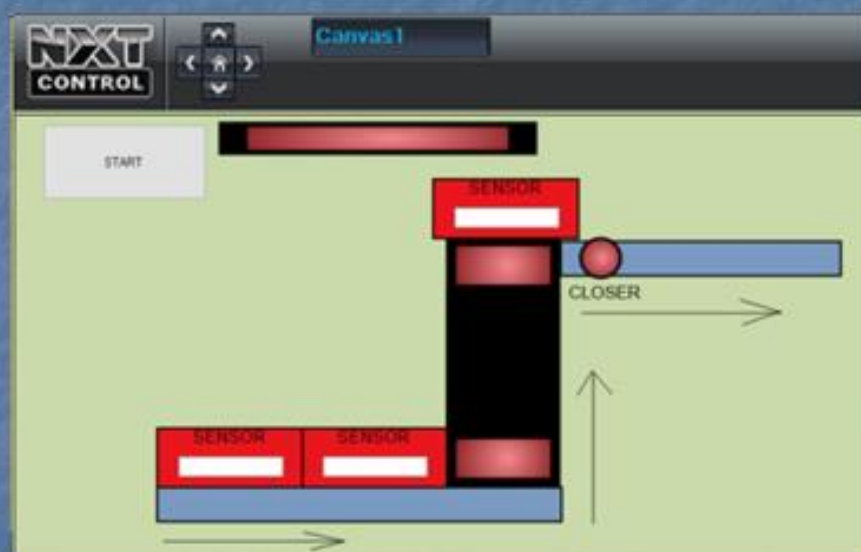
Отладка и тестирование системы ФБ обрабатывающей станции



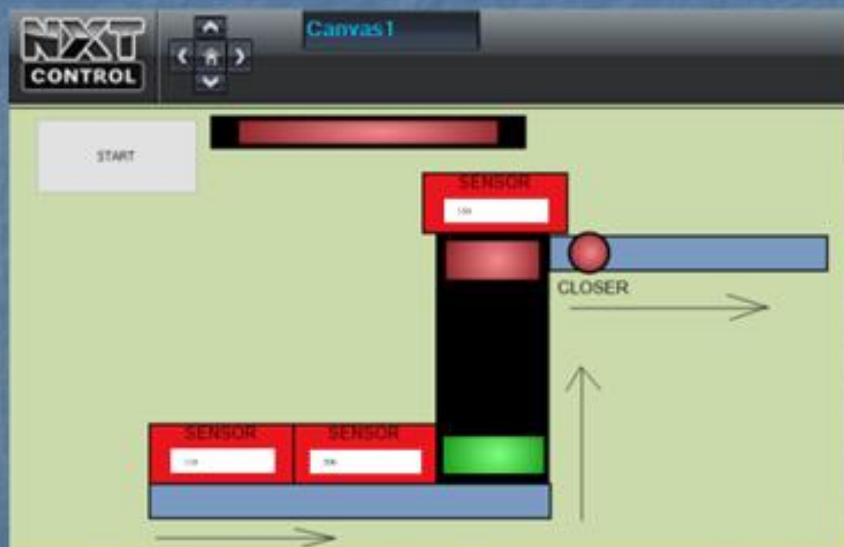
Отладка и тестирование системы ФБ обрабатывающей станции



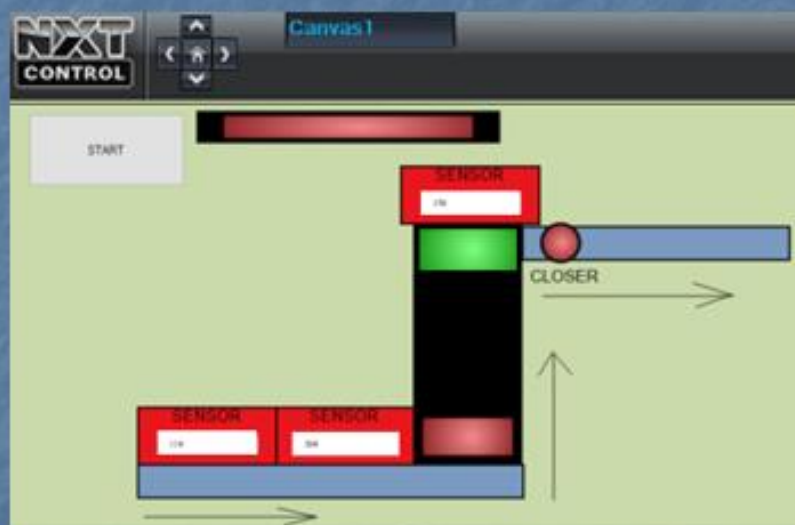
Разработка визуальной модели тестирующей станции



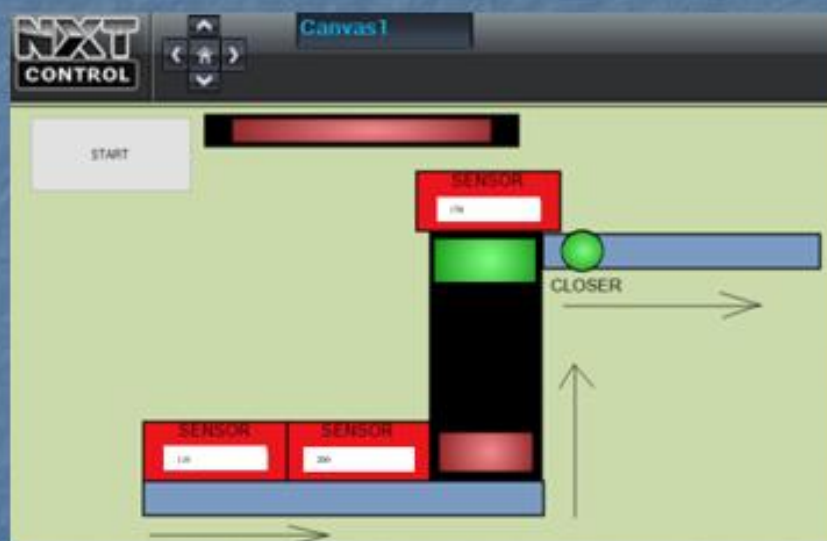
Разработка визуальной модели тестирующей станции



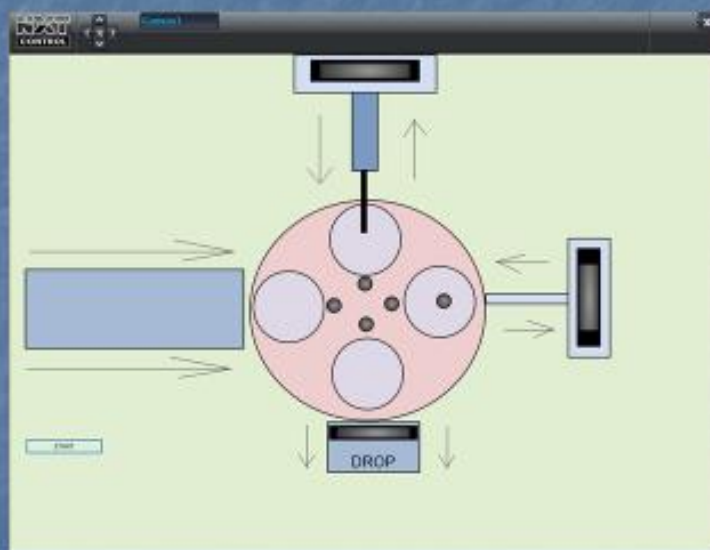
Разработка визуальной модели тестирующей станции



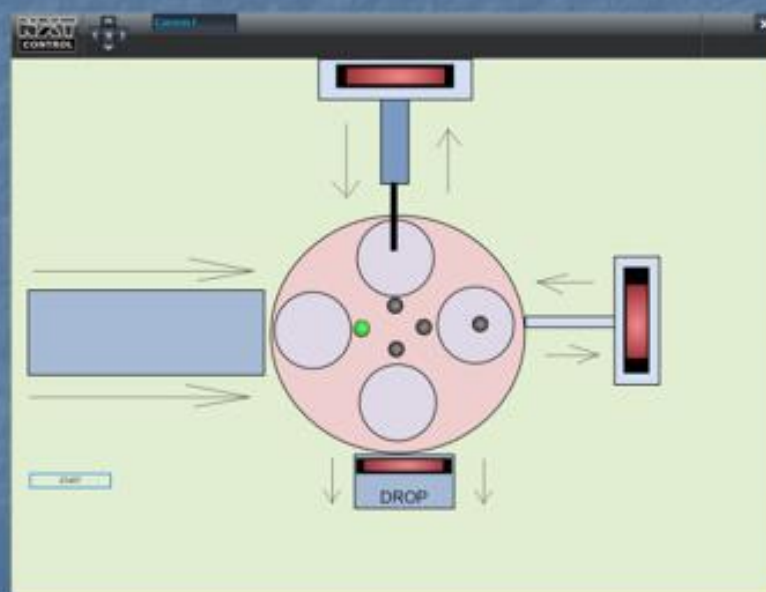
Разработка визуальной модели тестирующей станции



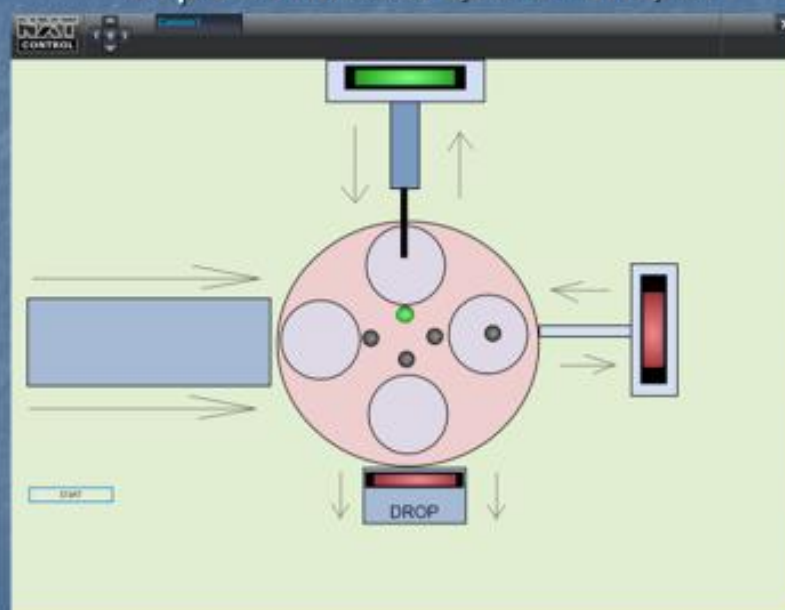
Разработка визуальной модели обрабатывающей станции



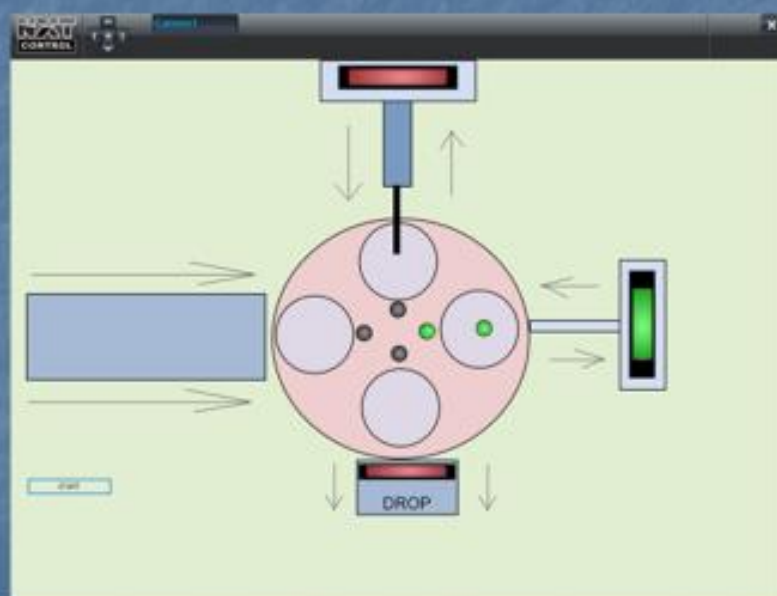
Разработка визуальной модели обрабатывающей станции



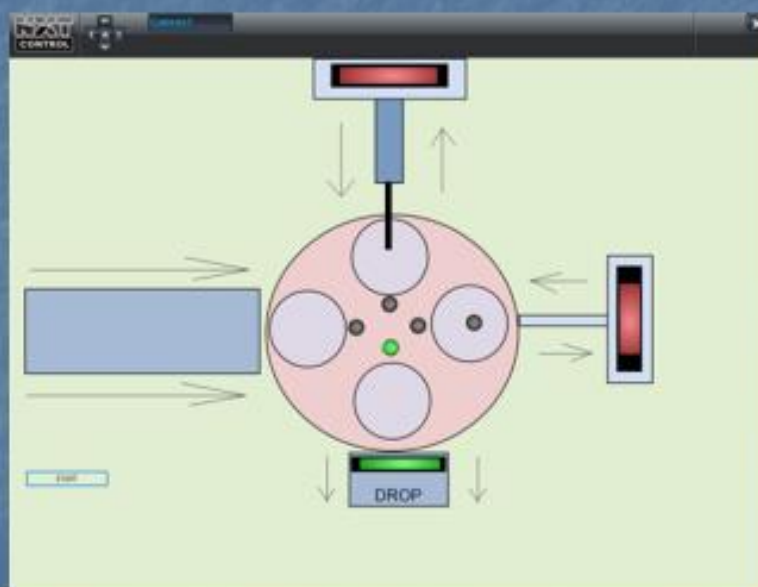
Разработка визуальной модели обрабатывающей станции



Разработка визуальной модели обрабатывающей станции



Разработка визуальной модели обрабатывающей станции



Заключение

- Согласно заданию, была разработана визуальная модель тестирующей и обрабатывающей станций установки FESTO на основе функциональных блоков IEC 61499. Была изучена и использована среда разработки NXTStudio 2.0.
- Были разработаны визуальная модель тестирующей и обрабатывающей станций и их алгоритм работы в составе производственной установки FESTO.
- Данную бакалаврскую работу можно использовать для обучения студентов работе со средой разработки NXTStudio 2.0.
- В разделе БЖД рассмотрены вопросы организации освещения места работы инженера-разработчика

Спасибо за внимание!

					ПГУ 2.09.03.01.23.001 ПЗ	Лист
						100
Изм.	Лист	№ докум.	Подпись	Дата		