Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

(ФГБОУ ВО «АмГУ»)

Факультет Энергетический

Кафедра Автоматизации производственных процессов и электротехники

Специальность 15.03.04 Автоматизация технологических процессов и происводств

КУРСОВОЙ ПРОЕКТ

Тема работы: Имитационное моделирование систем управления движением

По дисциплине: Автоматизация технологических процессов

Исполнитель
студент группы 641 – об _____ Т.А. Дорофеева

Руководитель
доцент, канд. техн. наук _____ А.Н. Рыбалев

Нормоконтроль
доцент, канд. техн. наук _____ А.Н. Рыбалев

Благовещенск 2020

Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

АМУРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

(ФГБОУ ВО «АмГУ»)

Факультет Энергетический

Кафедра автоматизации производственных процессов и электротехники

ЗАДАНИЕ

К курсовому проекту студента 641гр. Дорофеева Татьяна Александровна

1. Тема курсового проекта: Имитационное моделирование систем управления движением

2. Срок сдачи студентом законченной работы _____

3. Исходные данные к курсовому проекту:

1) ФГОС направление подготовки бакалавров 15.03.04 Автоматизация технологических процессов и производств

2) Учебный план направления подготовки бакалавров 15.03.04 Автоматизации технологических процессов и производств

4. Содержание курсовой работы (перечень подлежащих разработке вопросов):

1) Исследование и описание физических моделей;

2) Построение имитационных моделей;

3) Разработка программ управления и экранов визуализации.

5. Перечень материалов приложения (наличие чертежей, таблиц, графиков, схем, программных продуктов, иллюстративного материала и т.п.):

ПРИЛОЖЕНИЕ А – техническое задание

ПРИЛОЖЕНИЕ Б – код подпрограммы Kont_ri

ПРИЛОЖЕНИЕ В – код подпрограммы r_upr

ПРИЛОЖЕНИЕ Г – код подпрограммы a_upr

Лист 1 – Разработка 3D-модели в SolidWorks

Лист 2 – Экспорт 3D-модели в Simscape Multibody Link;

Лист 3 – Запуск OPC сервера и его настройка

6. Дата выдачи задания 10.10.2019

Руководитель курсовой работы Рыбалев Андрей Николаевич, доцент кафедры АПП и Э, канд. техн. наук.

Задание принял к исполнению (дата):_____

(подпись студента)

РЕФЕРАТ

Курсовой проект содержит 70 с., 41 рисунка, 1 таблицу, 5 источников.

АВТОМАТИЗИРОВАННАЯ СИСТЕМА УПРАВЛЕНИЯ ТЕХНОЛОГИЧЕСКИМ ПРОЦЕССОМ, СИСТЕМА ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ, СТАНДАРТ OPC.

Объекты для разработки проекта представлены в виде математических моделей, которые практически полностью реализуют необходимые свойства реальных объектов. Цель работы заключается в проектировании имитационных моделей систем управления АСУ ТП. Такая система включает в себя: модель объекта; управляющую программу; экран визуализации. Метод рассмотренный в курсовом проекте позволяет отказаться от физического объекта и программируемого логического контроллера, они реализованы в виде программ, разных классов и типов.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Разработка АСУ современных технологических процессов – сложная и ответственная задача, решение которой производится в несколько этапов: от составления математической модели до проектирования человеко-машинного интерфейса. В нашем курсовом проекте будет фигурировать имитационное управление систем движения. Имитационное моделирование — метод исследования объектов, основанный на том, что изучаемый объект заменяется имитирующим объектом. С имитирующим объектом проводят эксперименты (не прибегая к экспериментам на реальном объекте) и в результате получают информацию об изучаемом объекте. Имитирующий объект при этом являет из себя информационный объект. Цель имитационного моделирования — получение приближенных знаний о некотором параметре объекта, не производя непосредственное измерение его значений. Понятно, что это необходимо тогда и только тогда, когда измерение невозможно, или оно стоит дороже проведения имитации. При этом для изучения этого параметра мы можем пользоваться другими известными параметрами объекта и моделью его конструкции. Допуская, что модель конструкции достаточно точно описывает объект, предполагается, что полученные в ходе имитации статистические распределения значений параметра моделирующего объекта будут в той или иной степени совпадать с распределением значений параметра реального объекта.

Задачи которые нам нужно решить:

– имитационное моделирование технологического процесса в различных режимах работы при воздействиях, программно формируемых управляющей аппаратурой и средствами человеко-машинного интерфейса;

– отладка технологических программ;

– выбор наиболее удобных для пользователя средств визуализации технологического процесса и способов формирования управляющих воздействий.

Оперативный персонал задействует программный комплекс на этапе настройки АСУ ТП, а также в целях обучения.

Кроме того, разрабатываемая система, безусловно, будет весьма полезна в учебном процессе по образовательным программам, предусматривающие изучение дисциплин, связанных с проектированием АСУ ТП.

В рамках единого комплекса предлагается задействовать программные средства разных производителей и классов:

– система имитационного моделирования – для построения моделей технологического процесса;

– система класса PC-based controller – для программной реализации алгоритмов управления на языках программирования промышленных контроллеров;

– SCADA-система (supervisory control and data acquisition – система диспетчерского управления и сбора данных) – для визуализации технологических процессов и оперативного управления.

Перечисленные программные средства предназначены для исследования и разработки компонентов АСУ ТП, но, используемые по отдельности, не могут решать перечисленные выше задачи.

Для построения прототипа было решено использовать следующие программные продукты:

MathWorks® MATLAB®, Simulink® (среда имитационного моделирования);

3S-Smart Software® CODESYS® (PC-эмулятор ПЛК SP PLCWinN, OPC-сервер);

Выбор данных программ обусловлен опытом их применения в учебном процессе.

1 ТЕХНОЛОГИЯ ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ СИСТЕМ УПРАВЛЕНИЯ ДВИЖЕНИЕМ

Понятие «*компьютерного моделирования*» в сфере информационных технологий относительно ново и связано со становлением и выделением относительно традиционного моделирования с помощью ЭВМ (последние - это, как правило, функционально-ориентированные автоматизированные системы поддержки математического и других видов моделирования, реализуемые обычно как системы библиотечного типа) двух современных видов компьютерного моделирования : - структурно-функционального и имитационного. Компьютерное моделирование – эффективный метод решения задач анализа и синтеза сложных систем. Методологической основой компьютерного моделирования является системный анализ (в то время, как у моделирования на ЭВМ – те или иные разделы теории математических моделей). В мире информационных технологий имитационное моделирование переживает второе рождение. Интерес к этому виду компьютерного моделирования оживился в связи с существенным технологическим развитием систем моделирования, которые на сегодняшний день являются мощным аналитическим средством, вобравшим в себя весь арсенал новейших информационных технологий, включая развитые графические оболочки для целей конструирования моделей и интерпретации выходных результатов моделирования, мультимедийные средства и видео, поддерживающие анимацию в реальном масштабе времени, объектно-ориентированное программирование.

1.1 Концепция

Концепция состоит в том, что у нас имеется объект ,имитационную модель движения которого нужно создать. Для этого в программе SOLIDWORKS создаем 3D-модель каждого элемента этой системы и в последующем объединить их в сборку. После с помощью Simscape

Multiboby Link мы экспортируем 3D-модель объекта (в виде сборки ее элементов) в MatLab и получаем Simulink модель ,настраиваем ее и в приложении CoDeSys пишем для нее программу ,с помощью которой будем управлять движением имитационной модели объекта. Для того что бы заставить работать вместе Simulink модель и программу из CoDeSys, используем OPC сервер. Так же программу из CoDeSys нужно занести в ПЛК (в нашем случае мы используем PC-эмулятор ПЛК SP PLCWinN). Схема концепции изображена на рисунке 1.

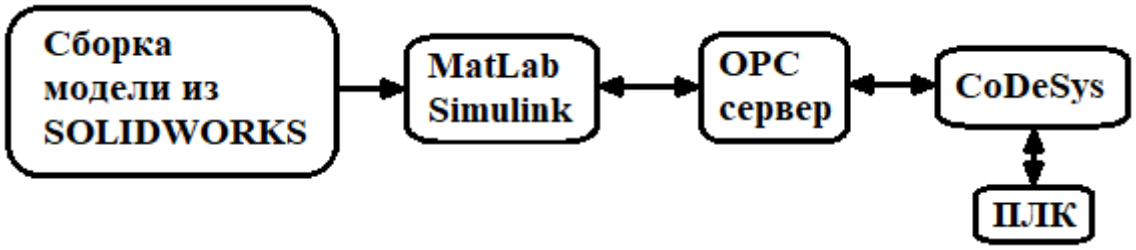


Рисунок 1 – Схема концепции

1.2 Разработка 3D-модели в SolidWorks

Несмотря на большие возможности Simscape Multibody Link, смоделировать сложную механическую систему очень сложно. Это связано с необходимостью определять тензоры моментов инерции элементов, учитывать координаты их расположения и т.д. Для упрощения моделирования в среде Simscape Multiboby Link, компания MATHWORKS разработала специализированный CAD-транслятор, обеспечивающий создание динамических моделей механизмов в среде Simscape Multibody Link на основе их твердотельных моделей разработанных в CAD-системах (напр., SolidWorks). При применении CAD-транслятора тензоры моментов инерции и присоединительные размеры передаются из CAD-системы в Simscape Multibody Link без изменений, при этом, работоспособность моделей проверяется в CAD-системе посредством установления правильных связей между деталями механизма. Такой подход значительно облегчает задачу и расширяет возможности имитационного моделирования мехатронных систем.

Solidworks — программный комплекс САПР предназначенный для автоматизации этапов подготовки производства. Основной задачей Solidworks является работа с 3D моделями, именно о них и будет идти речь.

Для примера представим, что у нас есть какое-то основание и на нем куб, который двигается по двум осям X и Y. Разделив модель объекта по частям , имеем два объекта куб (рисунок 2) и основание (рисунок 3) из которых состоит модель. Эти объекты объединяем в сборку(рисунок 4).

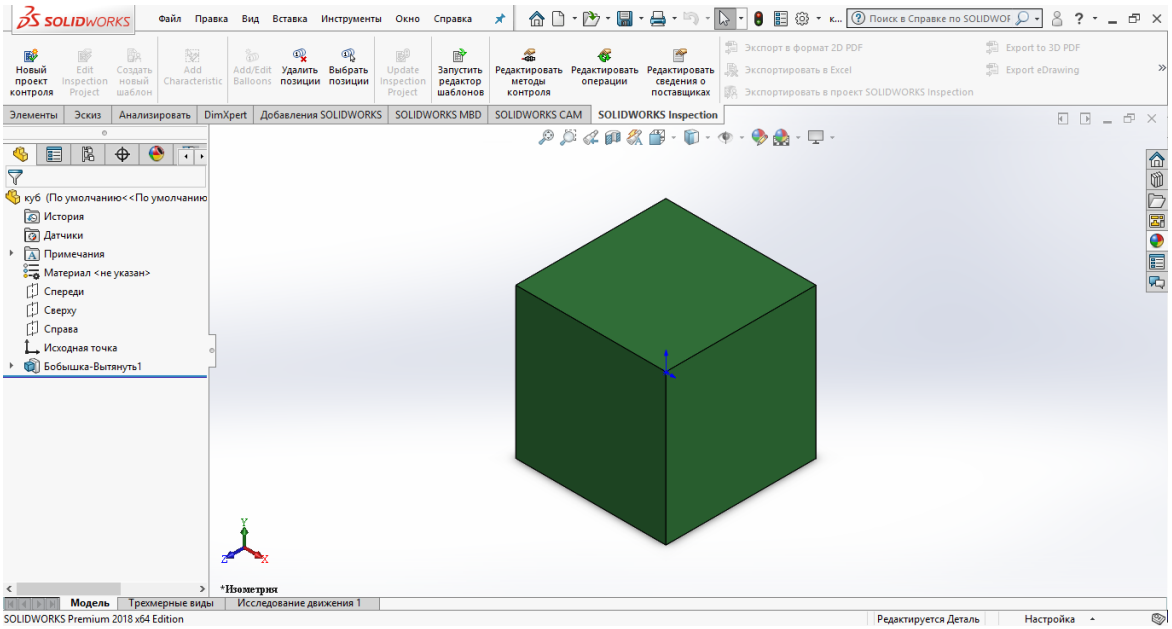


Рисунок 2 – 3D-модель куба

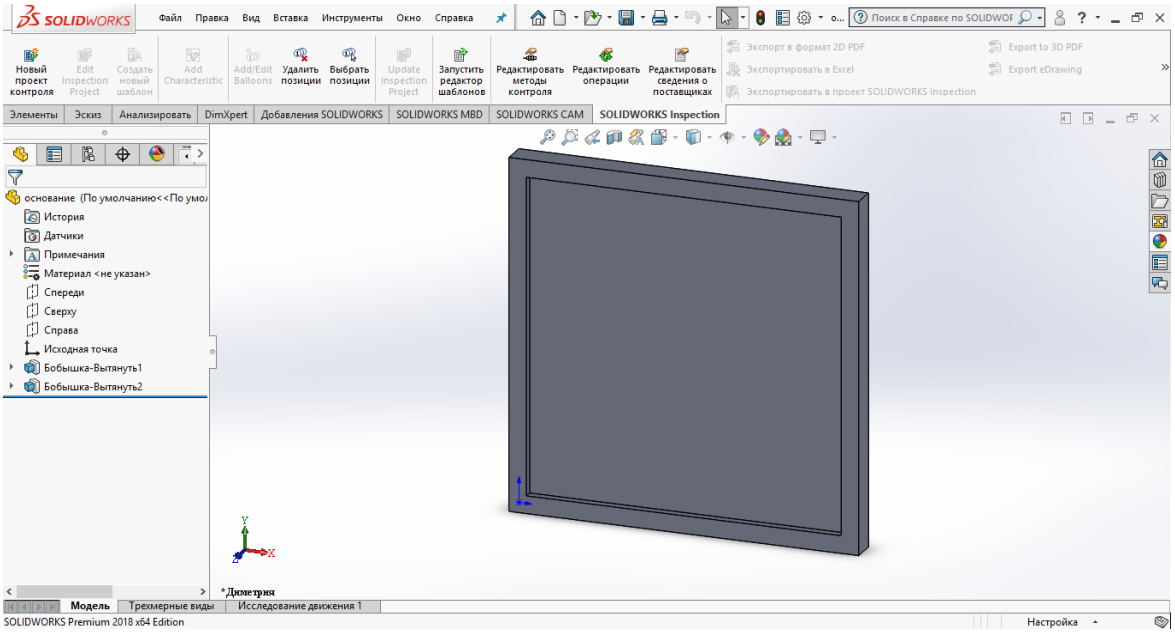


Рисунок 3 – 3D-модель основания

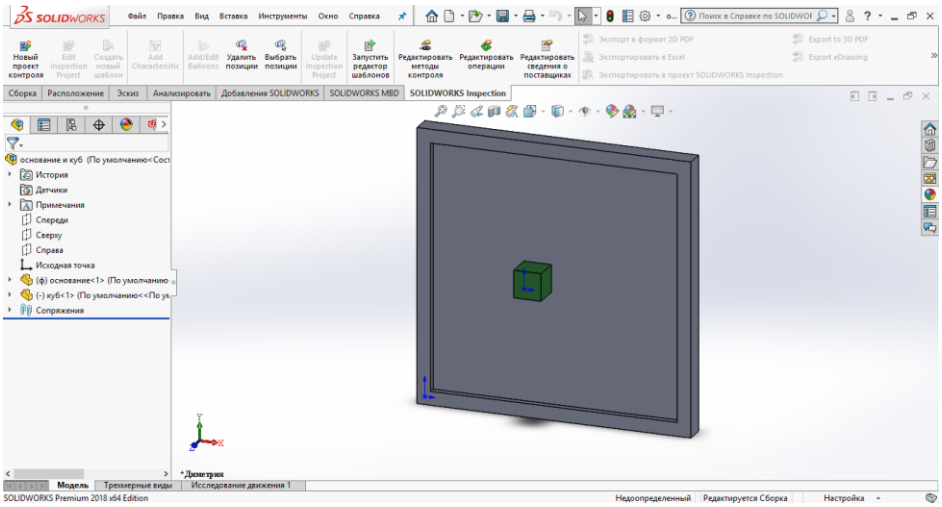


Рисунок 4 – 3D-модель сборки

1.3 Пакет Simscape Multibody Link

Simscape Multibody Link входит в программный комплекс имитационного моделирования Simulink системы MATLAB и позволяет моделировать сложные физические системы с помощью ненаправленных сигнальных графов. При этом, в инструментальной базе Simscape Multibody Link имеются блоки, посредствам которых можно организовать взаимодействие разработанной модели (передачу и прием сигналов) с другими компонентами библиотеки Simulink. Это дает возможность моделировать мехатронные и робототехнические системы, транспортные средства, летательные аппараты, производственное оборудование, до того, как будет начато их производство.

1) Необходимо с официального сайта MATHWORKS скопировать дистрибутив Simscape Multibody Link – CAD транслятор для системы SolidWorks. При этом версия дистрибутива должна соответствовать версии Matlab. Необходимо также скопировать установочный файл install_addon.m;

2) Запустить Matlab. Если используется операционная система Windows 7, то Matlab должен быть запущен с правами администратора. Для этого в свойствах файла (ярлыка на рабочем столе) MATLAB.exe на вкладке «Совместимость» необходимо выбрать опцию «Выполнять эту программу от имени администратора;

3) Из системы MatLab открыть установочный m-файл – install_addon.m. После этого запустится программа «Editor», предназначенная для создания и редактирования m-файлов среды Matlab и откроется диалоговое окно редактирования функции install_addom;

4) В главном меню «Editor» выбрать: Debug – Run Configuration for install_addon.m – Edit Run for install_addon.m;

5) В появившемся диалоговом окне «Edit M-File Configurations» прописать:

% Modify expression to add input arguments. % Example: % a = [1 2 3; 4 5 6]; % foo(a);

install_addon(указать полный путь архива, например: 'C:\Documents and Settings\student\Рабочий стол\Матлаб\smlink31.win32.zip')

6) Запустить M-File, путѐм нажатия кнопки Run;

7) Далее в окне Windows нажать кнопку Пуск – Выполнить – ввести cmd;

8) В командной оболочке ввести: matlab-regserver;

9) В окне консоли MatLab Command Window ввести: regmatlabserver;

10) В окне Command Window MatLab ввести: enableservice('AutomationServer',true). Ответ должен быть 1, есль ans =0, то необходимо ещѐ раз ввести enableservice('AutomationServer',true);

11) В окне Command Window MatLab ввести: smlink_linksw. В положительном результате должно появиться сообщение об успешной регистрации.

В Simscape Multibody Link есть блоки,находящиеся в библиотеке Simulink (рисунок 5), с помощью которых наша модель может работать .

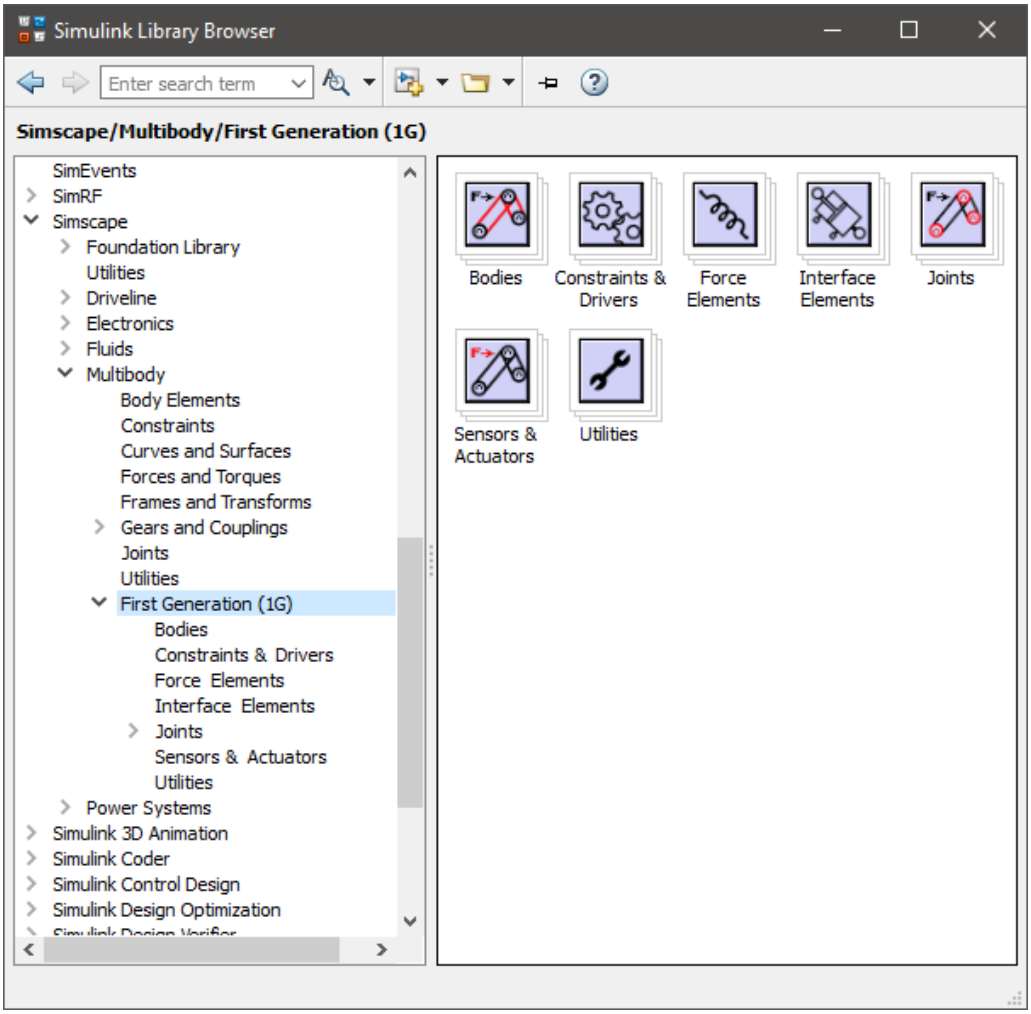
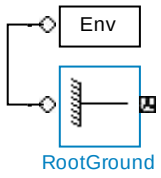
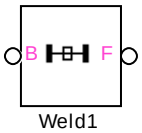
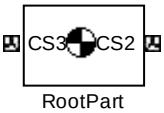
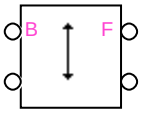
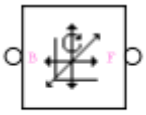
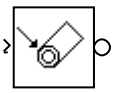
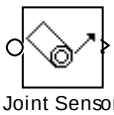


Рисунок 5 – Библиотека элементов Simulink

В таблице 1 описаны блоки Simscape Multibody Link.

Таблица 1 – Описание блоков Simscape Multibody Link подсистемы Robots

	Блок Ground, представляет собой неподвижную заземленную точку, подключенный к его входу блок, задает механические параметры для одного механизма.
	Блок Wade, представляет неподвижное соединение звеньев.

<table>
<tr><td> 
</td><td rowspan="2">Блок Body, представляет собой твердое настраиваемое тело. В свойствах блока указывается:
– Масса тела и момент инерции;
– Координаты центра тяжести(CG);
– Один или более систем координат тела(CSS);
– Дополнительная геометрия, цвет.</td></tr>
<tr><td> 
</td></tr>
<tr><td> 
</td><td>Блок Custom Joint, представляет общее пользовательское соединение с несколькими степенями свободы. Соединяет два тела с комбинацией призматических, вращающихся и/или сферических примитивов. Этот блок ограничен максимум шестью степенями свободы: до трех вращательных степеней свободы и до трех поступательных степеней свободы. Первый приметив, прикрепленный к базе (B). Последний приметив, прикрепленный к последователю (F).</td></tr>
<tr><td> 
</td><td>Соединение между двумя телами представляет относительные степени свободы, блок Joint Actuator приводит в поступательное движение, либо вращательное движение эти тела с точки зрения линейной позиции.</td></tr>
<tr><td> 
</td><td>Блок Join Sensor измеряет положение блока в пространстве.</td></tr>
</table>

1.4 Экспорт 3D-модели в Simscape Multibody Link

Процесс передачи модели из SolidWorks в Simscape Multibody Link осуществляется следующим образом, берем модель сборки из SolidWorks и экспортируем Simscape Multibody Link (рисунок 6).

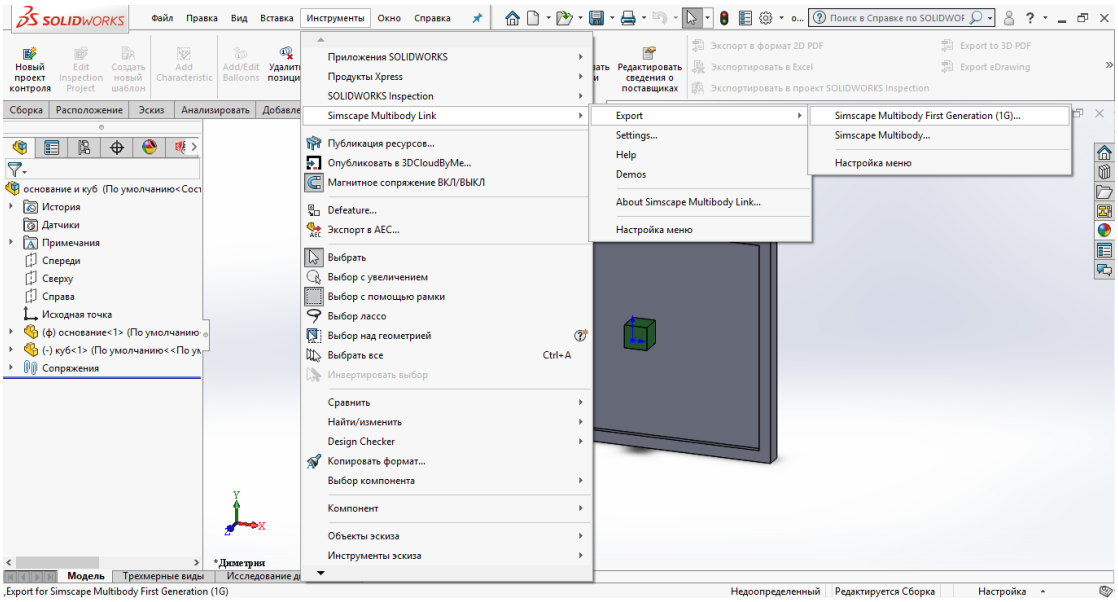


Рисунок 6 – Экспорт из SolidWorks в Simscape Multibody Link

Сохраняем файл в появившемся окне в формате .XML в удобное для вас место (рисунок 7).

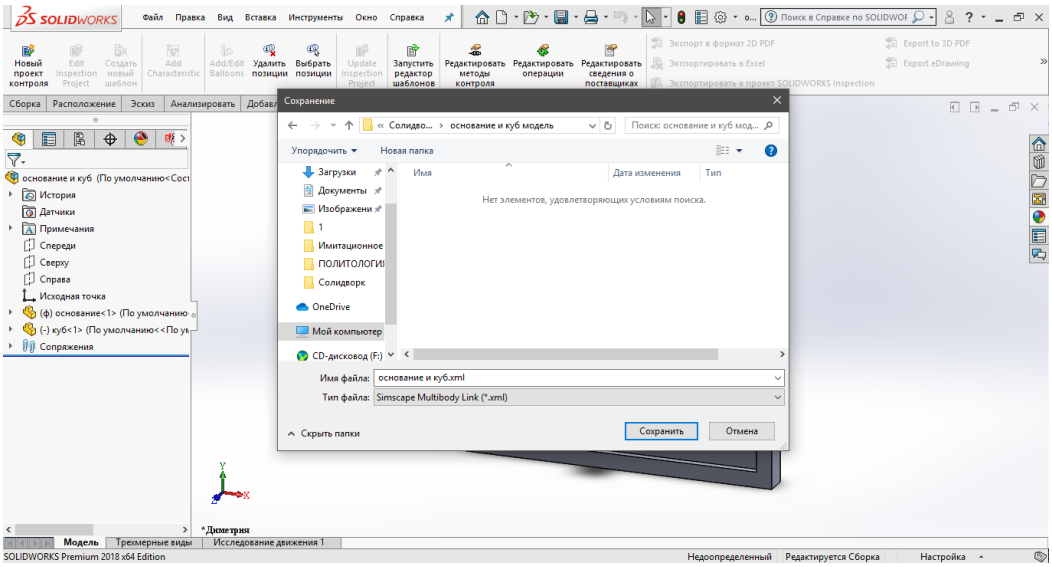


Рисунок 7 – Сохранение файла в .XML

В появившемся командном окне MatLab пишем команду mech_import (‘путь\имя файла.xml’)(Рисунок 8).

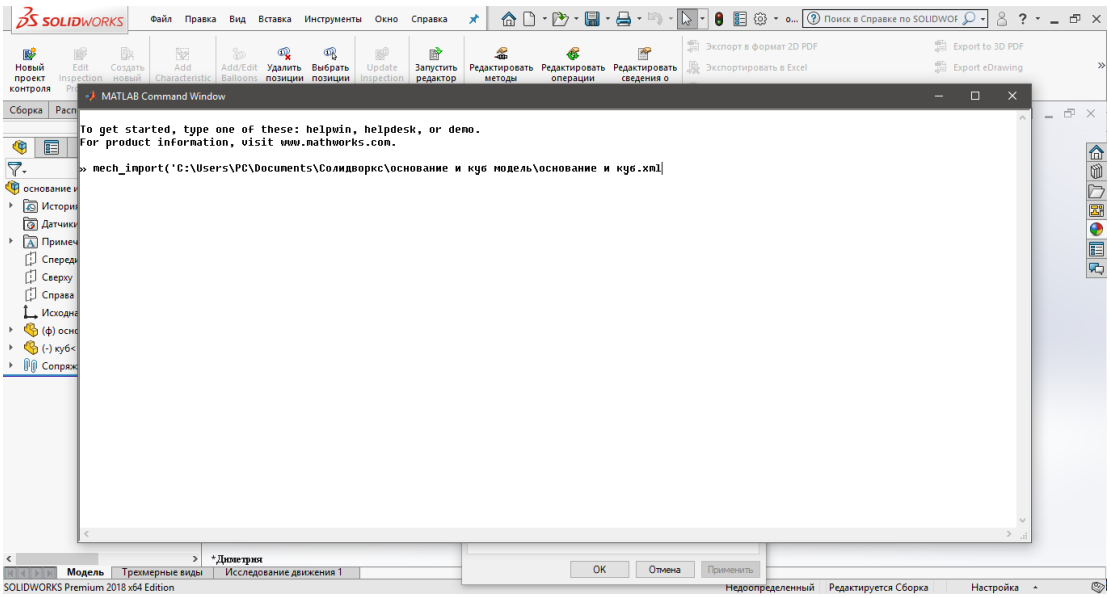


Рисунок 8 – командное окно MatLab

После того как ввели команду и нажали кнопку ввод ,создалась Simulink модель (рисунок 9) с которой можно в дальнейшем работать.

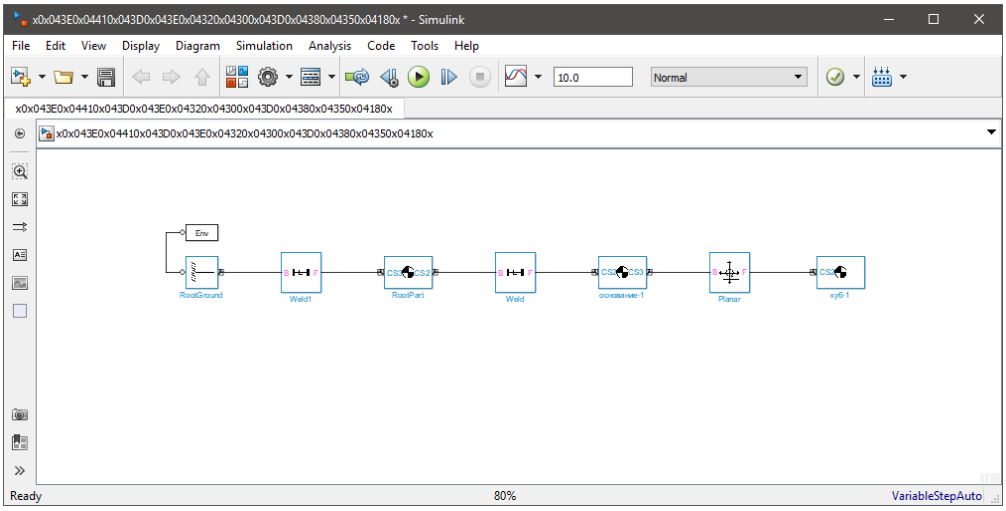


Рисунок 9 – Simulink модель

1.5 Разработка программы CodeSys для виртуального пакета

CodeSys — инструментальный программный комплекс промышленной автоматизации. Производится и распространяется компанией 3S-Smart Software Solutions GmbH

CodeSys – один из самых развитых инструментов для разработки программ ПЛК. CodeSys поддерживает 5 языков МЭК (IT , LD , FBD , SFC и ST) и еще один язык –CFC. То есть если разрабатывать программу в CodeSys

можно на любом из этих языков. Кроме того ,допускается комбинирование языков программирования в одной программе.

Прежде чем разрабатывать программу ,необходимо создать новый проект. Делается это с помощью главного меню файл – создать (рисунок 10)

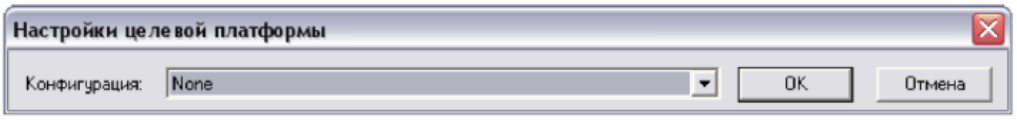


Рисунок 10 – Настройки целевой платформы

Проект является машинно-независимым, то есть программирование можно не привязывать к какой-либо конкретной модели контроллера. Если же вы хотите разрабатывать программу

для конкретной модели, то в окне настройки целевой платформы нужно выбрать эту модель в поле конфигурация. Однако вашего контроллера в списке может не оказаться. Тогда нужно будет найти целевой файл (target-file) и установить его, а потом создавать новый проект. В любом случае выбрать модель контроллера можно позже, уже после создания проекта.

Мы же всё это затеяли в учебных целях, поэтому оставим значение None и нажмём кнопку «ок», после чего появится окно.

Любой проект состоит из одного или нескольких модулей. В однозадачном проекте обязательно должен присутствовать главный программный модуль с именем PLC_PRG.

В этом окне мы устанавливаем следующие настройки:

1.Имя нового POU – PLC_PRG

2.Тип POU – программа

3.Язык реализации – ST. Язык реализации – это язык программирования, на котором будет написан программный модуль (POU) (рисунок 11).

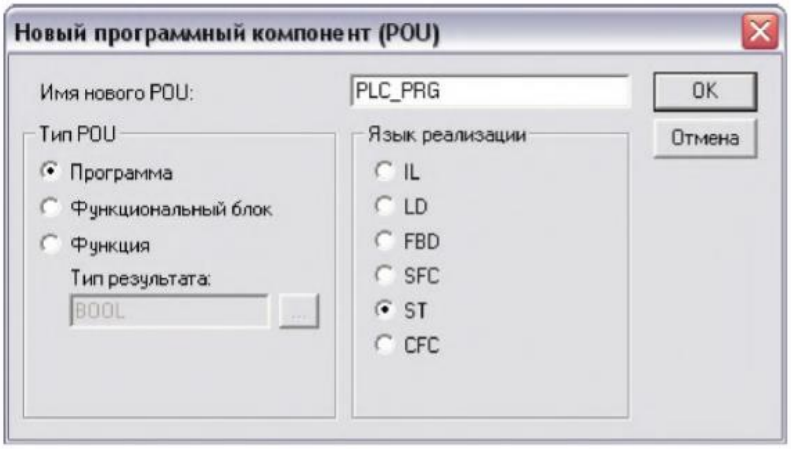


Рисунок 11 – Новый программный компонент

Нажимаем «ок». Всё, проект готов. Осталось только сохранить его. Как и во всех рассмотренных ранее случаях, сохраняем его в отдельном каталоге. Сохранение выполняется через меню файл– сохранить. Присвоим файлу имя TEST и сохраним проект.

После сохранения в выбранном каталоге появится файл TEST.PRO. Главное окно CoDeSys с новым проектом показано на рисунке 12.

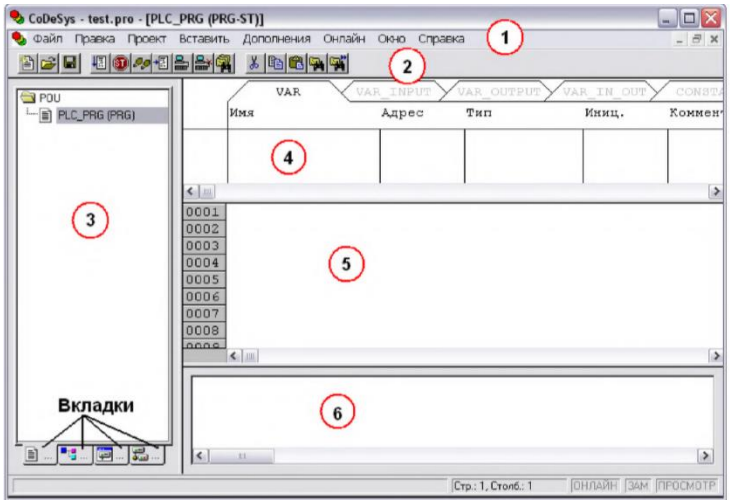


Рисунок 12 – Главное окно CoDeSys

1. Главное меню. Практически все команды можно выполнить через меню.

2. Панель инструментов. Дублирует основные команды меню.

3. Организатор объектов. Здесь отображаются и создаются все объекты проекта, такие как программные модули, визуализации и т.п. В нижней части

организатора объектов можно переключаться между его страницами (вкладками). Всего имеется четыре вкладки:

а. POU – список программных модулей

б. Типы данных

в. Визуализации

г. Ресурсы – глобальные переменные, конфигуратор задач, настройки целевой платформы и т.п.

4. Окно объявления переменных. В вашем случае оно будет выглядеть, скорее всего, по-другому, то есть переменные будут отображаться в виде списка. Чтобы переменные отображались в виде таблицы , необходимо изменить настройки программы следующим образом: в организаторе объектов перейти на

вкладку ресурсы, выбрать там раздел рабочая область, в появившемся окне выбрать категорию редактор, установить флажок напротив надписи объявления таблицей и нажать кнопку «ок».

5. Редактор исходного кода. Здесь и будем писать программу.

6. Окно сообщений. Здесь выводятся сообщения о ходе компиляции.

Для объявления глобальных переменных перейдём на вкладку РЕСУРСЫ, дважды щёлкнем

по разделу глобальные переменные, затем также двойным щелчком выберем раздел Global_Variables.

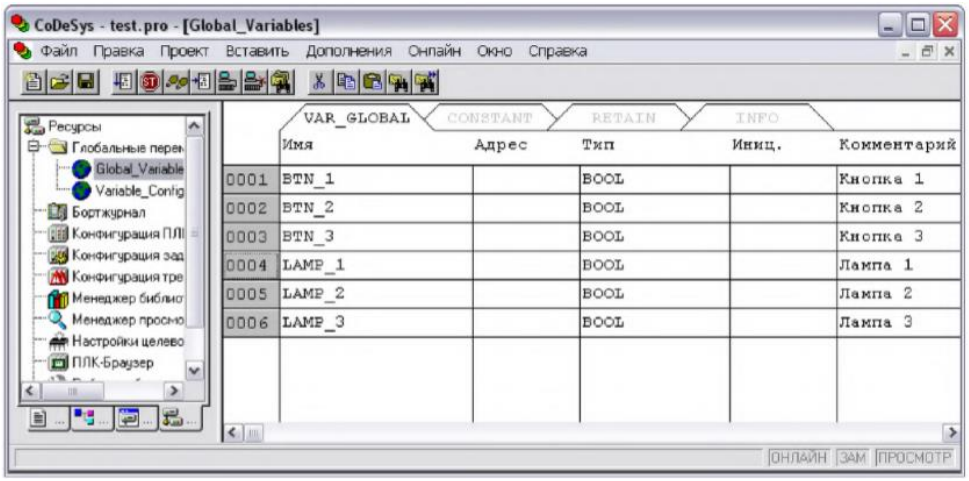


Рисунок 13 – Глобальные переменные

В верхней части таблицы расположены четыре вкладки (рисунок 13), имена которых соответствуюттипам объявляемыхданных:

1. VAR_GLOBAL – глобальные переменные

2. CONSTANT – глобальные константы

3. RETAIN – глобальные переменные, хранящиеся в ПЗУ контроллера. Если вы объявите переменную в этом разделе, то её значение будет сохранено даже после выключения

питания контроллера (правда, для этого нужно иметь контроллер).

4. INFO – информация о программе

Каждая вкладка таблицы (кроме INFO) имеет следующие столбцы:

1. Имя – идентификатор переменной

2. Адрес – адрес переменной. За исключением случаев связывания переменных с входами/выходами ПЛК адрес указывать нет необходимости. Принципы адресации в данной книге не рассматриваются.

3.Тип – тип данных, к которому принадлежит переменная

4.Инициализация – значение, которое будет присвоено переменной при старте программы ПЛК. Это не относится к переменным, хранящимся в ПЗУ, если только программа не работает в режиме эмуляции.

5. Комментарий – комментарий

Чтобы добавить в таблицу переменную, нужно щёлкнуть правой кнопкой по таблице и выбрать команду НОВОЕ ОБЪЯВЛЕНИЕ. В таблице появится новая строка со значениями по умолчанию. Чтобы изменить какое-либо значение, нужно щёлкнуть по нужной ячейке

таблицы и ввести с клавиатуры новое значение.

Все наши переменные будут иметь тип BOOL – логический тип, эквивалентный типу Boolean в Паскале. После ввода всех переменных, у вас должно получиться примерно так, как

показано на рисунке 13.

Теперь можно переходить к написанию исходного кода программы

1.6 Запуск OPC сервера и его настройка

Аббревиатура OPC (сейчас – это Open Platform Communications, взаимо-действие открытых платформ) ранее означала OLE for Process Control, OLE для управления процессами.

OPC использует технологию COM/DCOM для решения задачи обмена данными в системах промышленной автоматизации. OPC-клиент (любое приложение, например SCADA-система), вызывая определенные функции объекта OPC-сервера, подписывается на получение определенных данных с определенной частотой. OPC-сервер, получив каким-то образом эти данные (чаще всего,опросив оборудование), вызывает известные функции клиента и вручает ему данные. Таким образом, используются как прямые COM-вызовы (от клиента к серверу), так и обратные (callback, от сервера к клиенту).

OPC-серверы конкретных аппаратных устройств поставляются многими производителями аппаратуры. Связь сервера с аппаратурой может осуществляться через какой-либо физический интерфейс компьютера: последовательный порт, USB, Ethernet, плату расширения с выходом на промышленную сеть и т.д. Для приложения-клиента параметры физического подключения совершенно не интересны, поскольку при настройке соединения с сервером они не задействуются (они используются только припервоначальном конфигурировании самого сервера). Настройка клиента сводится к выбору нужного сервера из списка зарегистрированных в системе, соединения с ним и выбору переменных для чтения и записи из предоставляемого сервером списка переменных. Таким образом, OPC-сервер создает абстракцию аппаратуры, позволяя любому OPC-клиенту записывать и считывать данные с устройства.

Состав и схема взаимодействия программ-элементов имитационной системы показана на рисунке 14.

Рисунок 14 – Схема взаимодействия программ

Объект управления представлен своей имитационной Simulink-моделью. Говоря упрощенно, модель «пересчитывает» входные (управляющие) сигналы в выходные, несущие информацию о состоянии объекта. Для ввода-вывода используются специальные блоки из пакета OPC Toolbox, наличие которых в Simulink-диаграмме автоматически обеспечивает «работу» модели в реальном времени.

Частота пересчета модели, конечно, не совпадает с частотой обмена по протоколу OPC, – она существенно выше, что позволяет объекту изменять свое состояние и в период между обменами. Однако блоки препятствуют тому, чтобы модельное время изменялось быстрее, чем реальное, как это обычно имеет место в Simulink-моделях. Конечное время расчета в наших системах мы всегда будем устанавливать равным бесконечности (в Matlab за бесконечность «отвечает» константа inf – infinity, бесконечность). Таким образом, остановить расчет можно будет только вручную.

Управляющая программа выполняется на программном эмуляторе ПЛК SP PLC WinNT, входящем в состав пакета программ CoDeSys. Необходимость в «отдельном» программном эмуляторе состоит в следующем. Конечно, сама среда программирования CoDeSys дает возможность запуска открытых в ней проектов без подключения к ПЛК в режиме симуляции (Simulation Mode). Однако в данном режиме «общаться» с

запущенной программой, т.е. задавать входные и наблюдать выходные переменные, можно только из среды программирования. В этом случае следует забыть об использовании каких бы то ни было внешних программ и всю имитацию АСУ ТП внедрить непосредственно в проект CoDeSys. В принципе такой подход вполне имеет право на существование, поскольку CoDeSys предоставляет полный набор возможностей для его реализации: можно «оформить» программную модель объекта в виде одной или нескольких программных единиц (программ или функциональных блоков), а для имитации человеко-машинного интерфейса задействовать достаточно развитые средства визуализации самой системы CoDeSys. Но, на наш взгляд, такой «бюджетный» вариант имитационной системы все же проигрывает варианту с использованием разных программ.

Подробно последовательность действий, осуществляемых при подключении контроллеров системы CoDeSys через OPC-сервер к компьютеру, описана в . Ниже перечислены шаги, которые необходимо сделать для организации обмена с виртуальным контроллером SP PLCWinNT V2.4.

1. Создать программу (проект) в среде CoDeSys с целевой платформой 3S CoDeSys SP PLCWinNT V2.4. При настройке целевой платформы следует установить параметр Download symbol file вкладки General (рисунок 15).

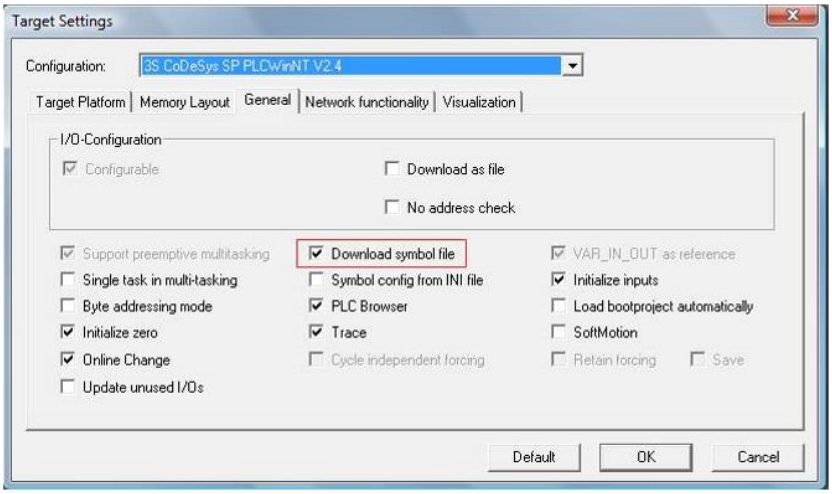


Рисунок 15 - Настройка целевой платформы

1. Объявить переменные для обмена по OPC. В общем случае такие переменные могут быть объявлены в любой из программ, однако для упрощения доступа лучше все их сделать глобальными.

3. Написать программу PLC_PRG. В программе должен быть, по меньшей мере, один оператор, поскольку без этого она не компилируется.

4. Сохранить проект под осмысленным именем в отдельную папку.

5. Запустить PLCWinNT (Пуск Все программы 3S Software CoDeSys SP PLCWinNT CoDeSys SP PLCWinNT V2.4)(рисунок 16) ,установить связь и загрузить программу в «контроллер».

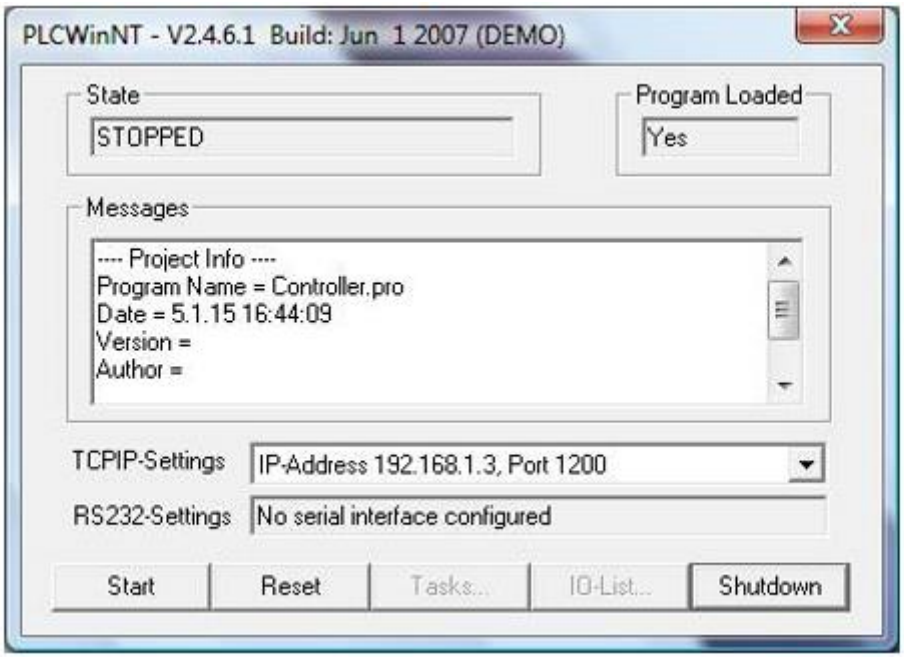


Рисунок 16 - Окно программы PLCWinNT

В случае, если связь с контроллером установить не удалось, следует настроить коммуникационные параметры в меню Online (рисунке 17).

6. Отключить CoDeSys от PLCWinNT и перейти в Опции (Options) в меню Project. Выбрать Symbol Configuration и установить галочку Dump symbol entries (рисунке 18).

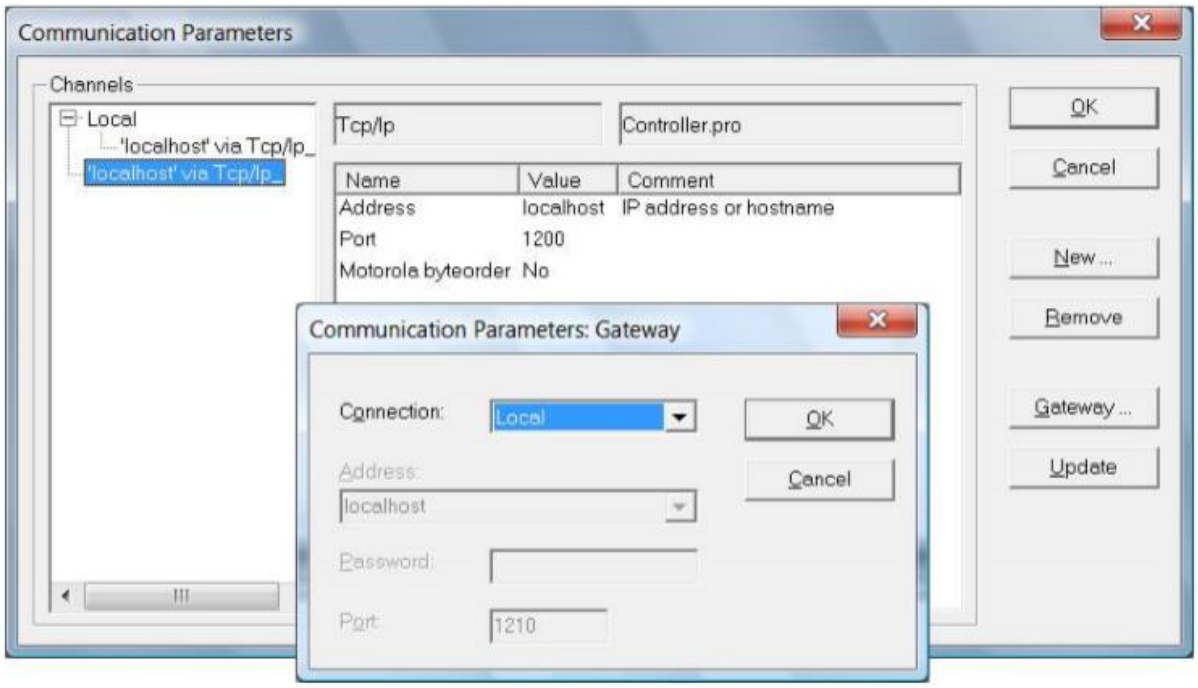


Рисунок 17 - Настройка коммутационных параметров

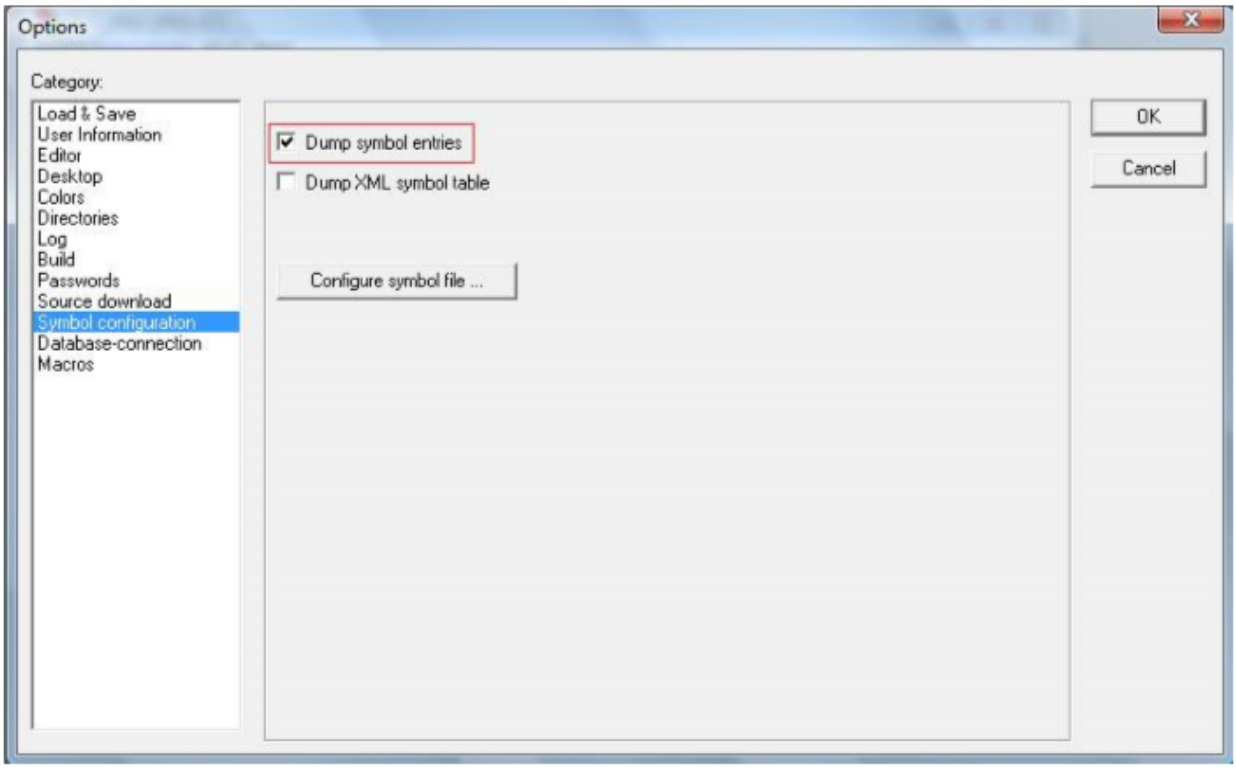


Рисунок 18 - Установка опций

7. Сконфигурировать «символьный файл», выбрав переменные для обмена по OPC, рисунок 19.

8. Настроить параметры OPC-сервера. Для этого необходимо запустить конфигуратор OPC-сервера (Пуск Все программы 3S Software

Communication CoDeSys OPC Configurator). В окне конфигуратора требуется добавить PLC (Append PLC) и настроить соединение (Connection). По существу достаточно выбрать из списка соединение, настроенное в проекте CoDeSys (рисунок 20)

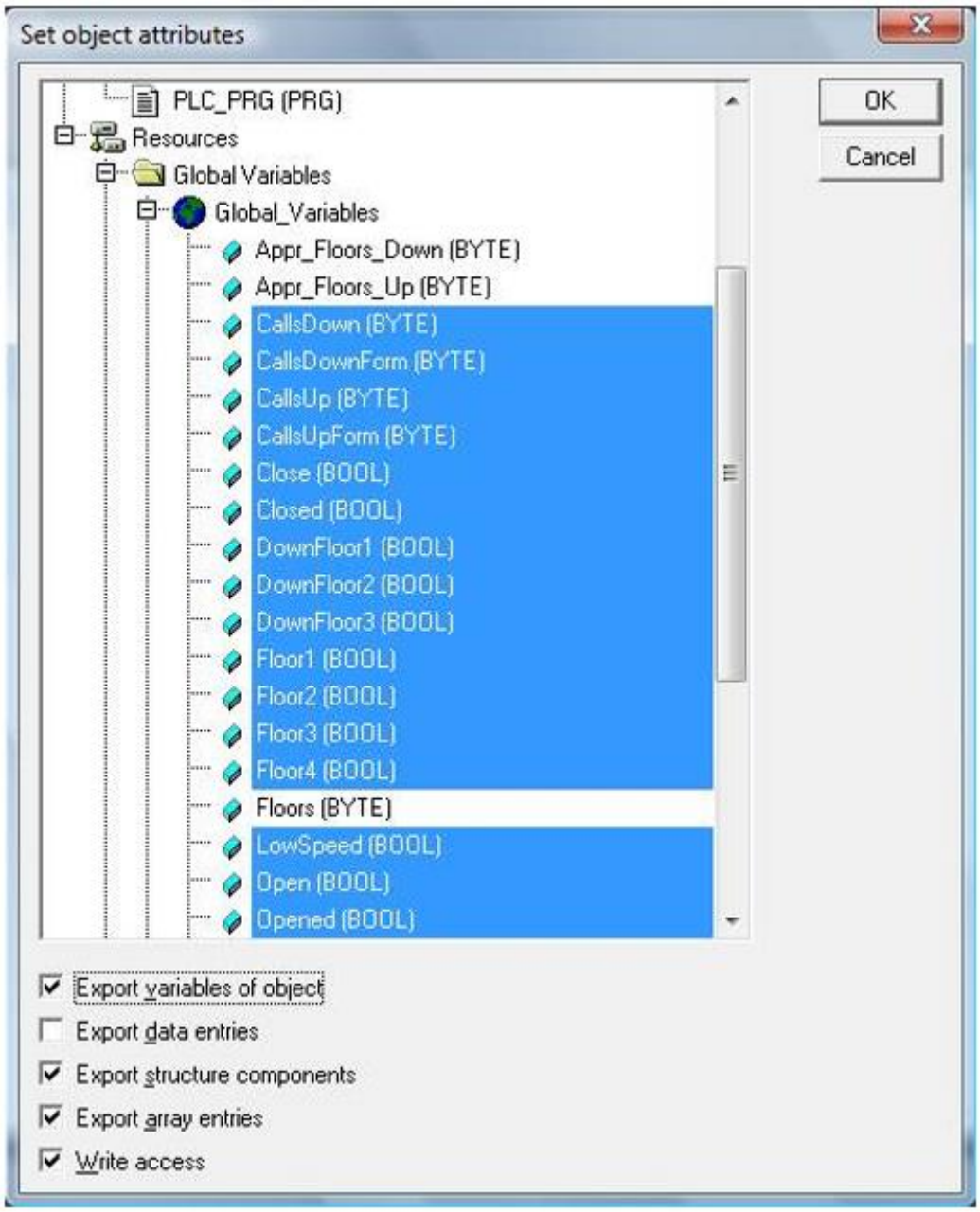


Рисунок 19 - Выбор переменных для обмена по OPC

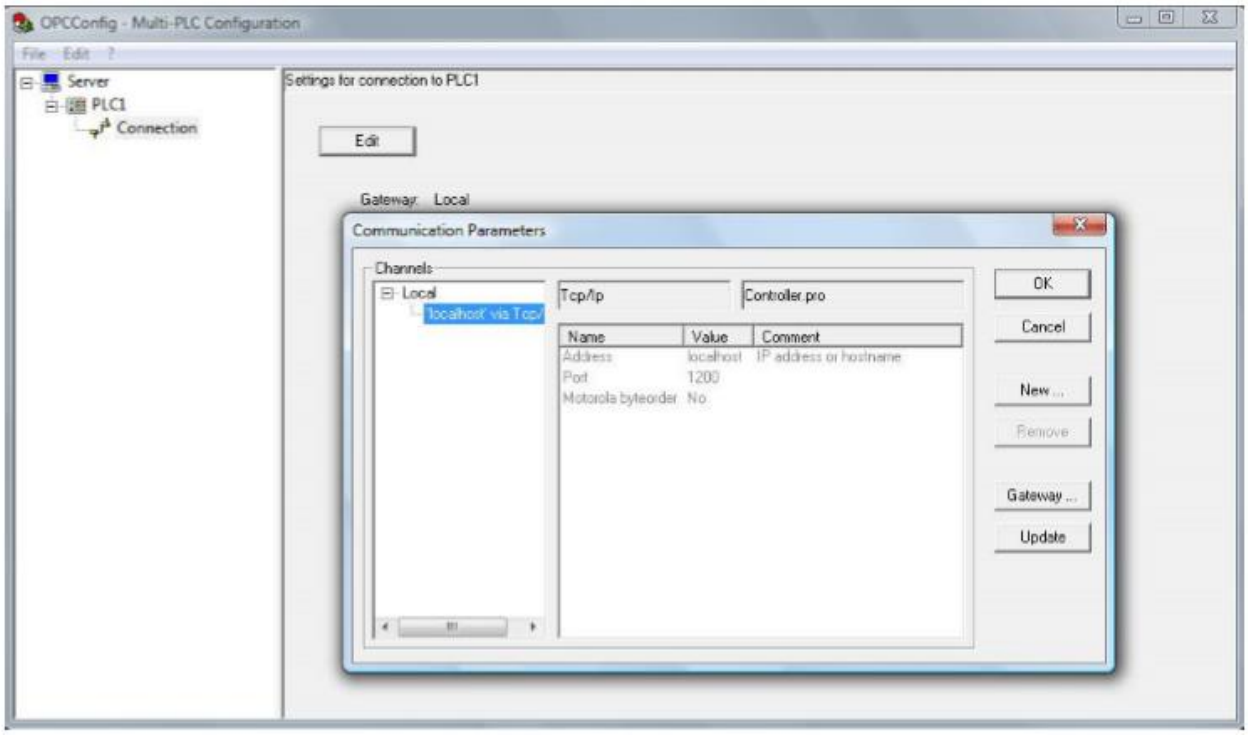


Рисунок 20 - Настройка OPC сервера

9. «Перестроить» программу (проект) CoDeSys, включив в нее все изменения, сделанные после загрузки в контроллер. Для этого требуется вызвать команду Clean all из меню Project, затем команду Rebuild all из того же меню. CoDeSys перекомпилирует программу и перезагрузит проект при следующем подключении к ПЛК.

10. Подключиться к PLCWinNT, согласившись на перезагрузку проекта. С помощью любой программы OPC-клиента, проверить доступность OPCсервера CoDeSys и переменных для обмена. Рекомендации по решению возможных проблем приведены в предыдущем пункте.

2 ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ СИСТЕМ УПРАВЛЕНИЯ ДВИЖЕНИЯ КОЗЛОВОГО КРАНА

Для курсового проекта будем моделировать козловой кран.

Модель козлового крана показана на рисунке 21.

Рисунок 21 – Физическая модель козлового крана

Козловой кран (КК) – многофункциональное оборудование для различных площадок и объектов, где грузоподъемные работы ведутся постоянно. Козловой кран — это агрегат мостового типа. Горизонтальная балка, по которой перемещается груз — конструктивно относится к мостовой группе подъемных механизмов. Основным отличием является то, что КК может сам передвигаться по рельсам. Усредненный срок эксплуатации козловых кранов — около 20 лет, наработка на отказ ≈ 3 000 циклов. Козловой кран состоит из:

1. ригель (ферма, балка) — пролетное строение. Мост легких и средних КК — одноблочный, тяжелых — двухблочный.

Балка бывает решетчатая или сплошностенная разного сечения. Она может иметь 1 или 2 консоли, или не иметь их совсем (консольные и бесконсольные краны);

2. опоры — жесткие, пространственные, гибкие или плоские. Если пролеты <25м, то все опоры — жесткие, внизу каждой зафиксированы ходовые тележки. При больших расстояниях между рельсами — одна нога фиксируется жестко, а 2-я — шарнирно. Каждая опора включает 1 или 2 стойки;

3. крановые (грузовые) тележки (или электроталь), оборудованные грузоподъемным механизмом, на который крепится груз. Грузовые тележки бывают монорельсовыми или двухрельсовыми. Двухрельсовые тележки передвигаются по обоим поясам — верхнему и нижнему, а однорельсовые — только по нижнему;

4. электрооборудование. Энергоснабжение КК — от внешней сети по троллеям или с помощью гибкого кабеля. Кабель накручивается на кабельный барабан, зафиксированный на одной из опор крана;

5. подкрановые рельсовые пути — наземные, рельсы Р43/Р24 (Р50/Р65) устанавливаются на деревянные (ж/б) шпалы на гравийной подушке. Рельсы должны быть с заземлением и тупиковыми упорами.Движение происходит благодаря двухребордным колесам, установленным на опорах. Размер колеи равен пролету крана. Помимо конструктивных элементов в устройство козлового крана входят механизмы передвижения всего крана и тележки, грузоподъемный механизм. Управлять подъемом и перемещением грузов КК можно с земли или из кабины.

Электрический способ — из кабины крановщика. С высоты четко видно груз любых размеров, кабина может быть оборудована бортовым компьютером для более точного размещения грузов.

2.1 3D-модель козлового крана

Изначально нам нужно создать трехмерную модель козлового крана. Разделаем по частям и поочередно изображаем. Первое, что нужно сделать, это создать подкрановые рельсовые пути, по которым кран будет перемещаться (рисунок 22).

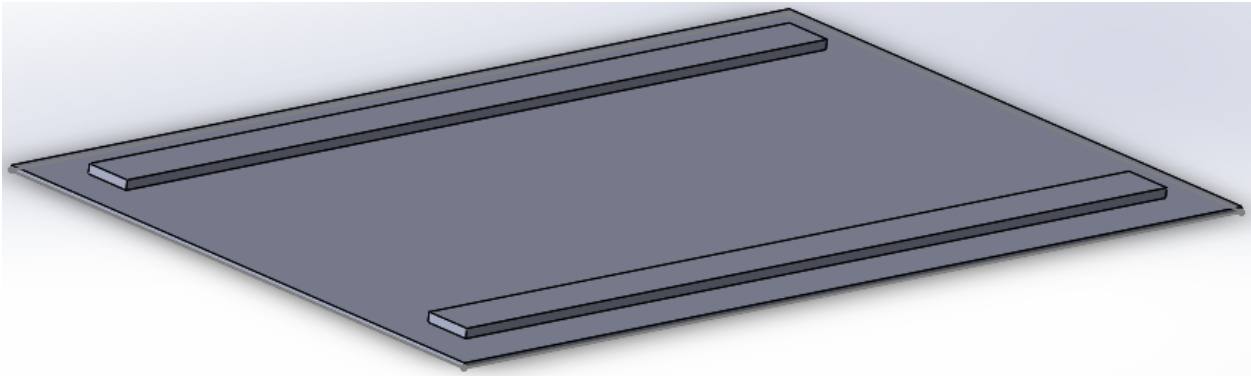


Рисунок 22 - подкрановые рельсовые пути.

Потом изображаются опоры (рисунок 23).

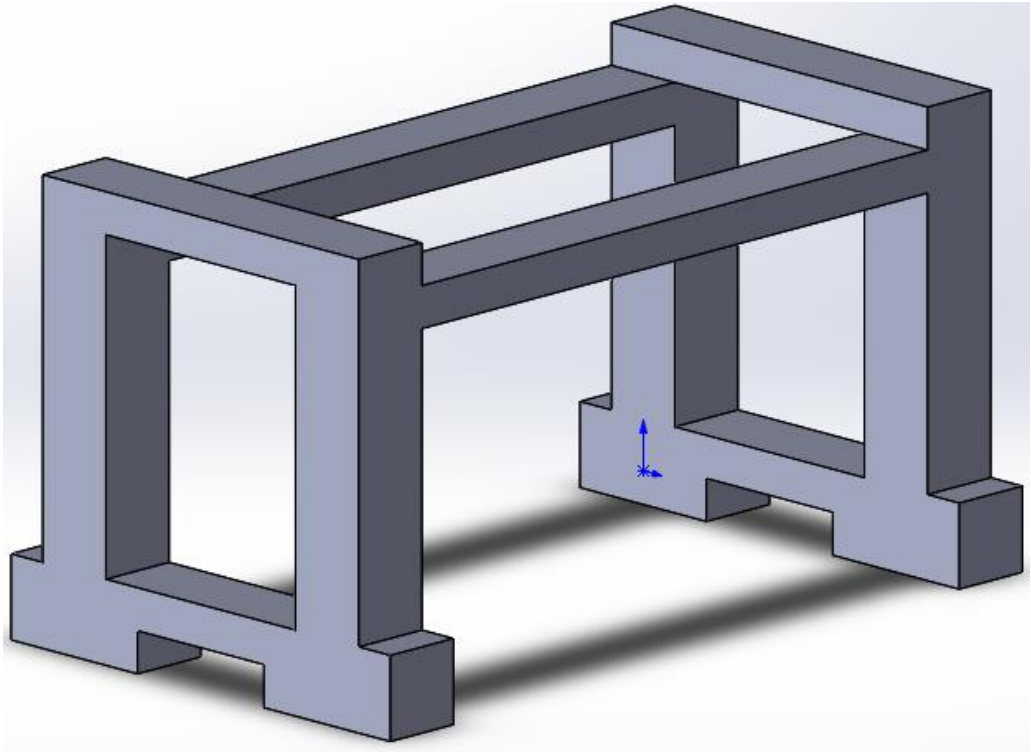


Рисунок 23 - опоры.

Далее мы делаем кабину и крюк и контейнер, изображенные на рисунке 24, рисунке 25 и рисунке 26.

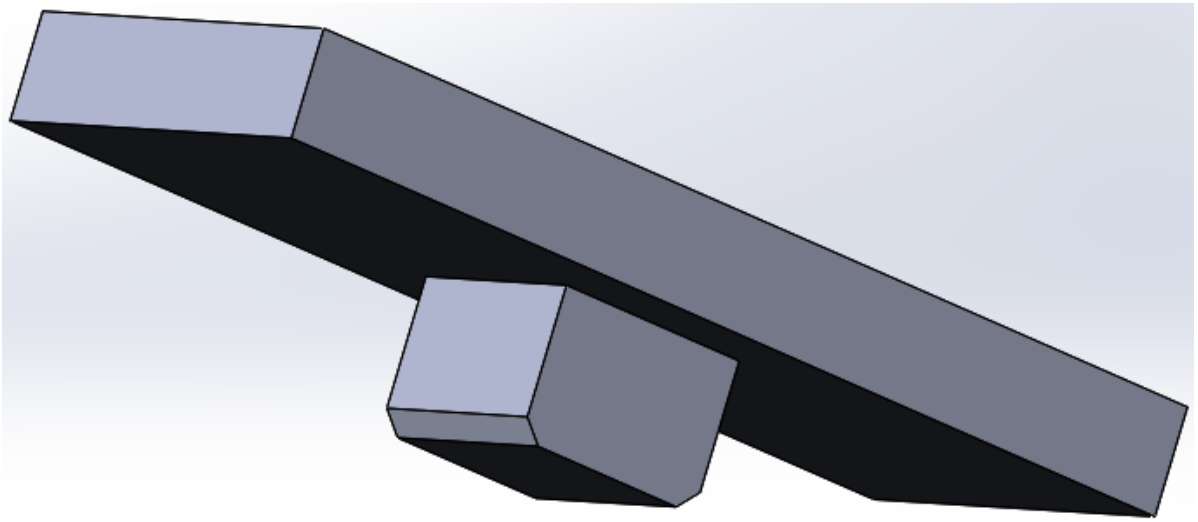


Рисунок 24 – кабина.

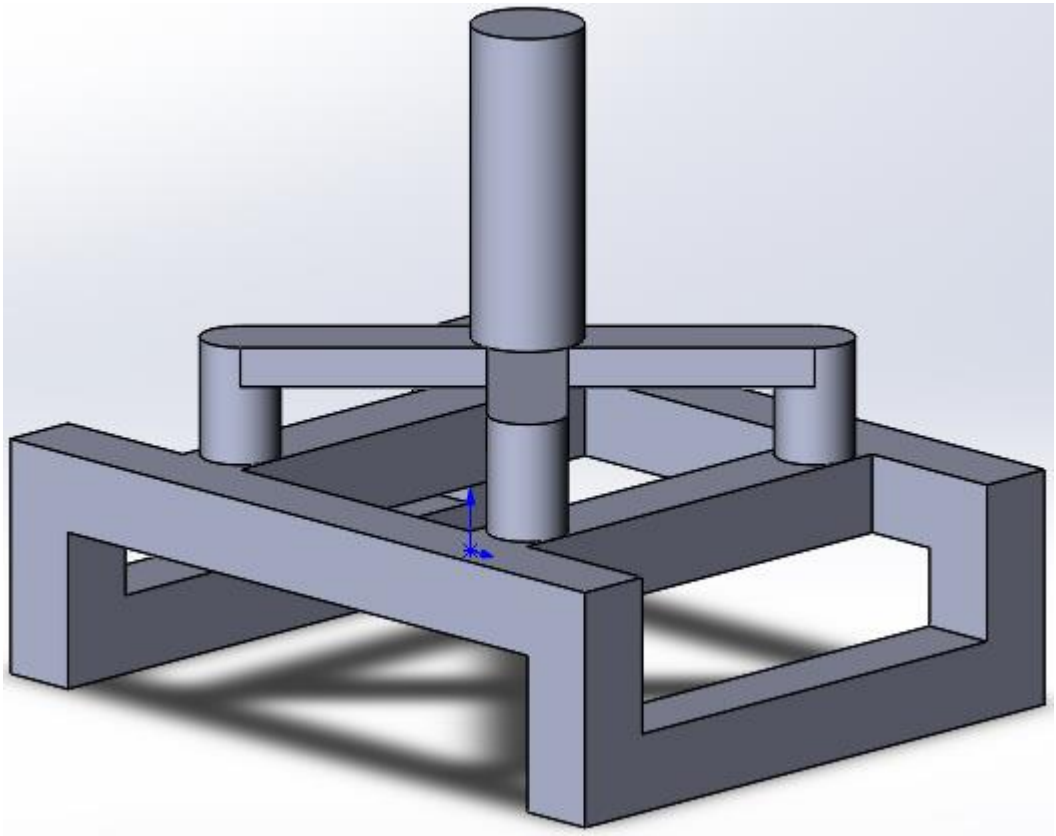


Рисунок 25 – крюк.

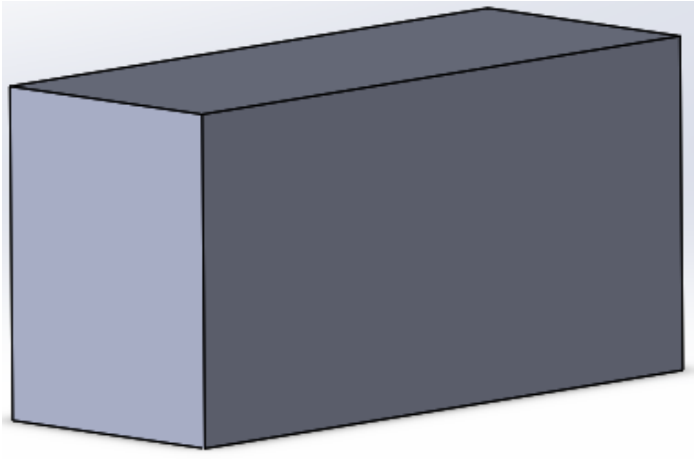


Рисунок 26 – контейнер.

Когда все части козлового крана готовы соединяем их в сборку и получаем 3D-модель крана.

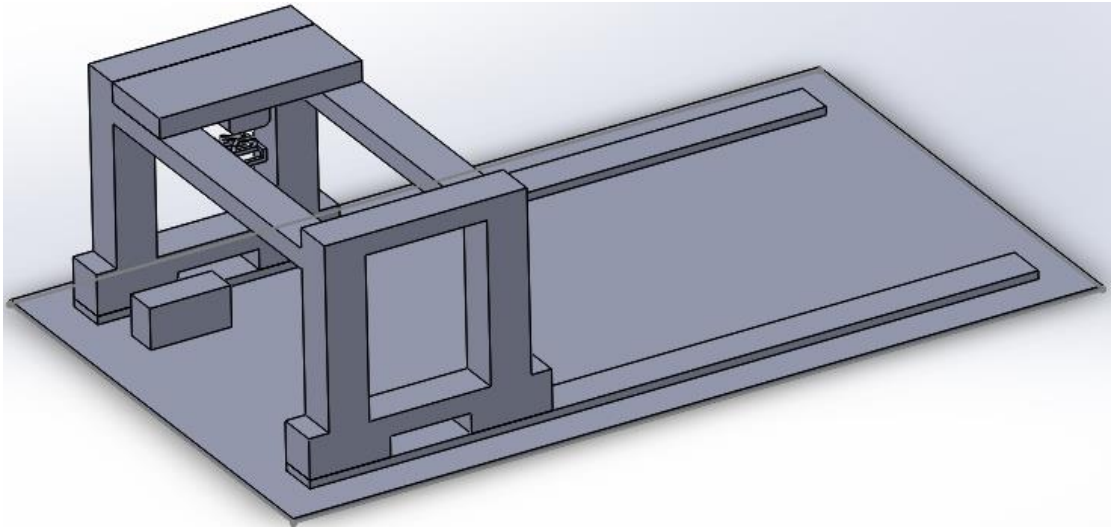


Рисунок 27 – 3D-модель козлового крана.

2.2 Создание Simulink модели

После переноса трехмерной модели из SolodWorks в Simulink, была получена модель которую нужно дорабатывать для того чтобы ей можно было задавать позиции объектам системы (тем самым управлять ею). Готовая система изображена на рисунке 28.

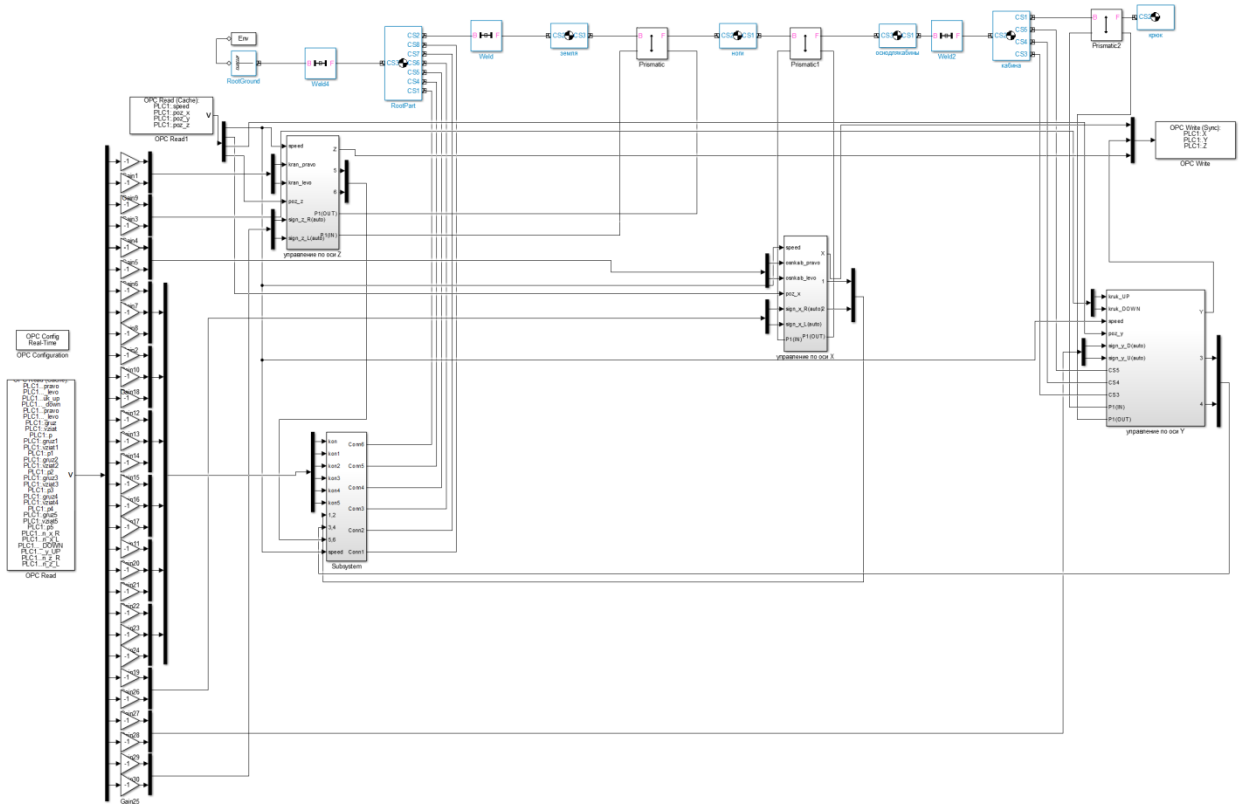


Рисунок 28 – Simulink модель

Саму систему можно поделить на четыре основные части, такие как: блок управление по оси X; блок управления по оси Y; блок управления по оси Z; блок задания положения контейнерам (Subsystem).

2.2.1 Основные блоки и их описание

Все блоки между собой практически идентичны. Некоторые блоки плотно взаимодействуют друг с другом, но при этом у каждого своя задача и объект управления.

Блок управление по оси X (рисунок 29) отвечает за передачу положения основанию кабины крана, которое совершает движение по оси X.

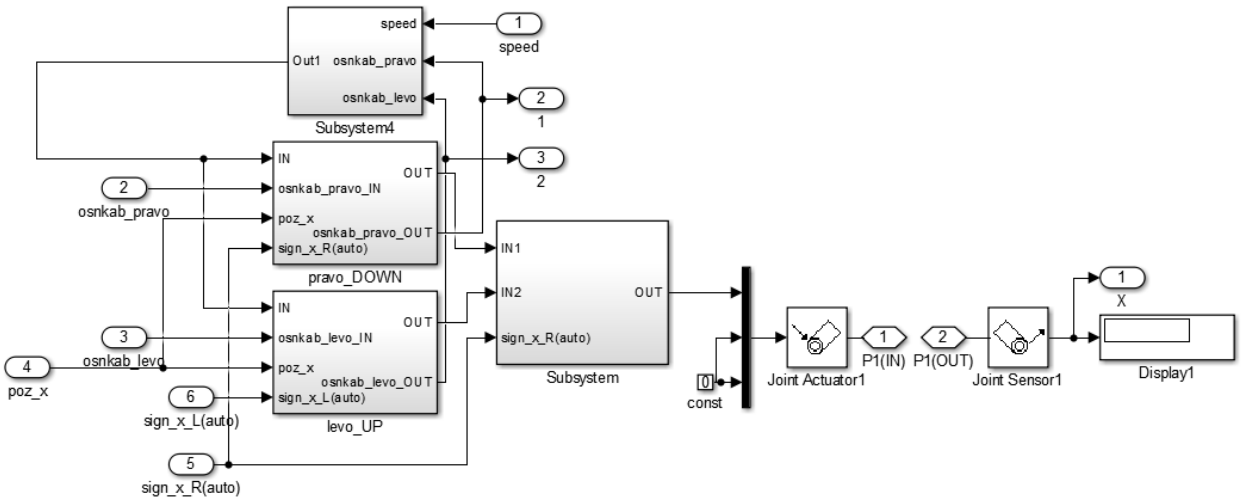


Рисунок 29 – Блок управление по оси X

Состоит он из следующих элементов: Subsystem4 (рисунок 24) – в данном блоке в основе лежит интегратор, в котором увеличивается или уменьшается число (в зависимости от того какой сигнал был подан, osnkab_pravo или osnkab_levo), тем самым это число будет служить позицией для основания кабины по оси X; pravo_DOWN (рисунок 30), levo_UP(рисунок 31) и Subsystem(рисунок 32) – служат для того чтобы можно было задать положение и занять именно его, независимо от скорости.

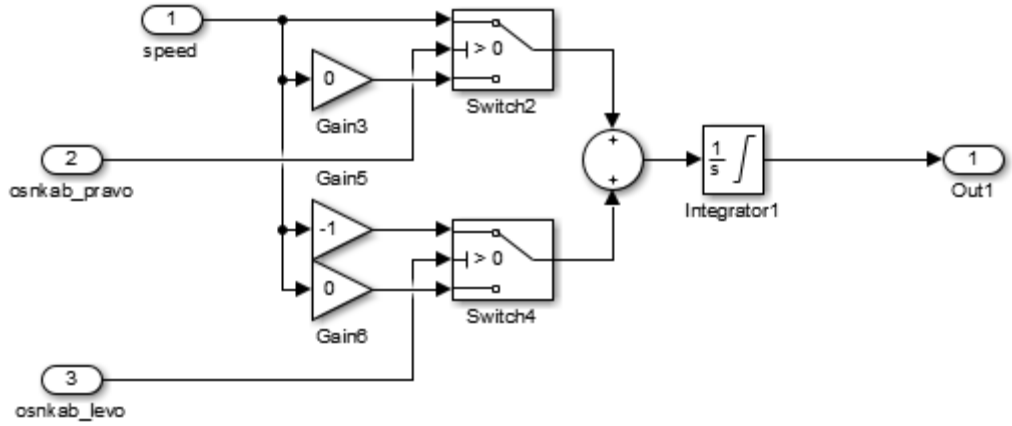


Рисунок 30 – Блок Subsystem4

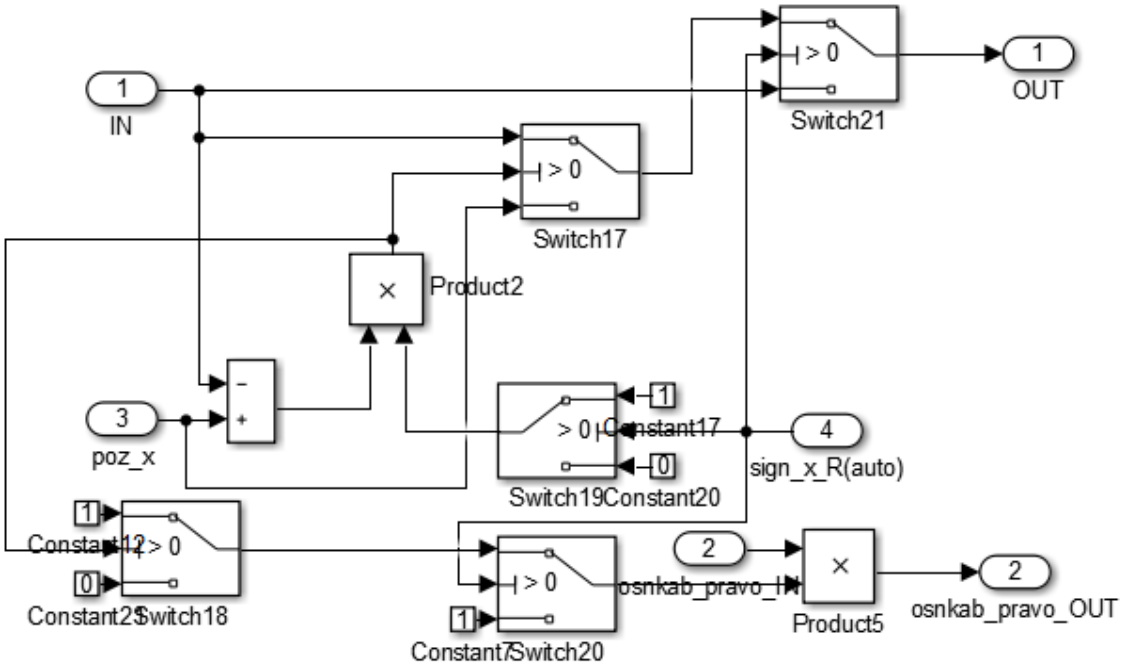


Рисунок 31 – Блок pravo_DOWN

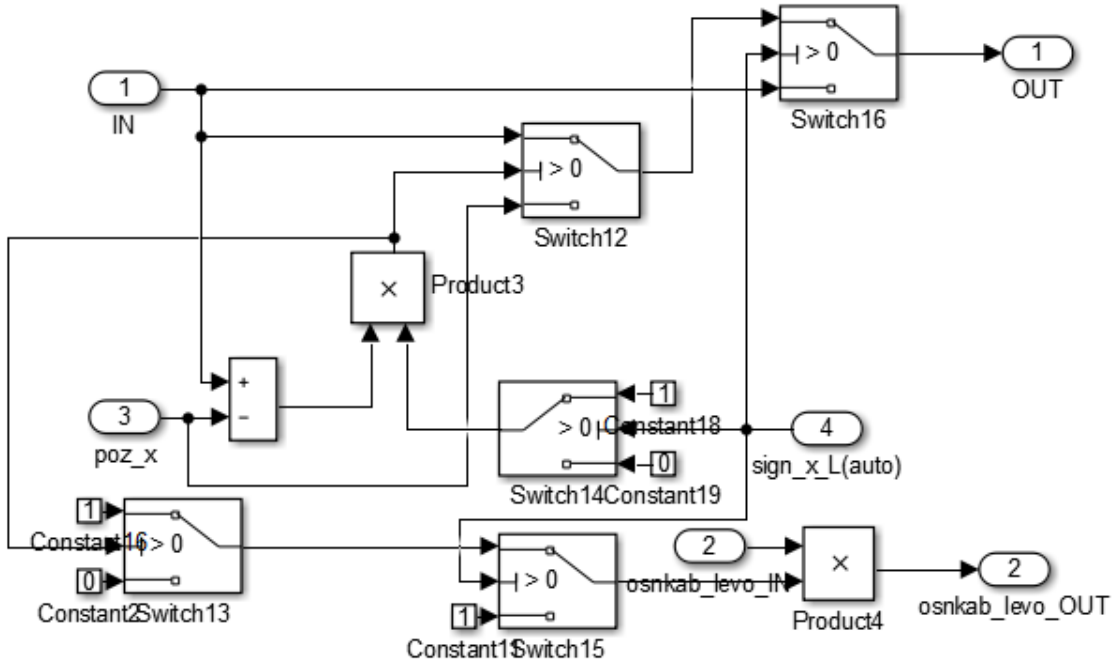


Рисунок 32 – Блок levo_UP

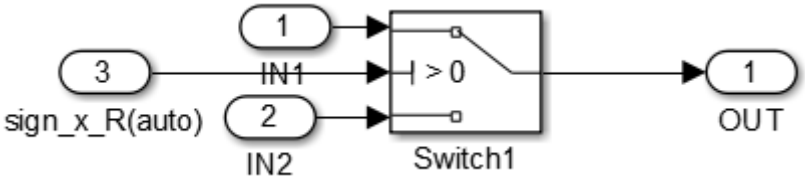


Рисунок 33 – Блок Subsystem

Аналогично устроены и остальные элементы системы, разница лишь в том, что блок управления по оси Y отвечает за крюк, блок управления по оси Z отвечает за основание крана, а блок задания положением контейнерам (Subsystem) отвечает за управление положением контейнеров в пространстве.

2.3 Создание управляющей программы

После того как разработали симулинк модель крана, переходим к разработке управляющей программы :

1. Описание визуализации панели управления движением модели.

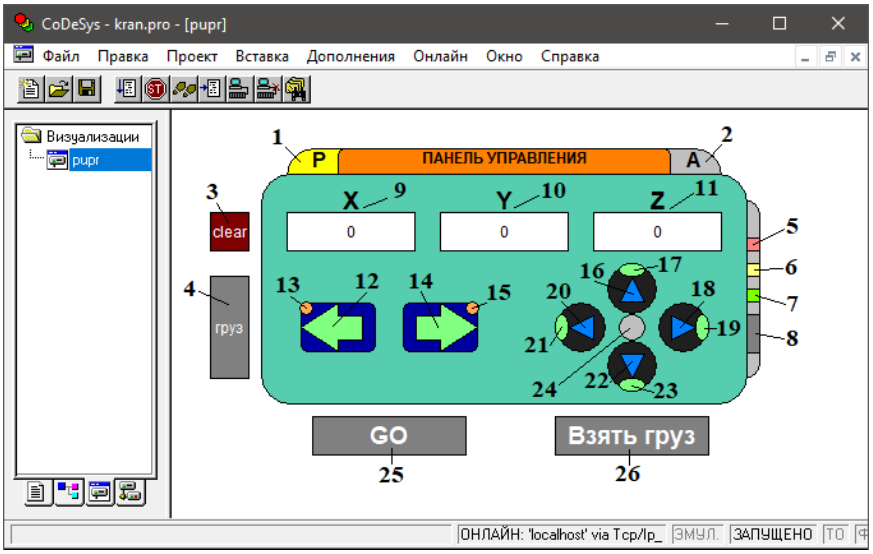


Рисунок 34 - панель ручного и автоматического управления

1 – кнопка ручного режима управления;

2 – кнопка автоматического режима управления;

3 – кнопка убрать все контейнеры;

4 – кнопка появления груза (по нажатию появляется груз, если есть идея получше, как назвать ее, то назови);

5 – кнопка включения высокой скорости движения модели;

6 – кнопка включения средней скорости движения модели;

7 – кнопка включения обычной скорости движения модели;

8 – кнопка задания начального положения;

9 – координаты крана по оси X;

10 – координаты крана по оси Y;

11 – координаты крана по оси Z;

12 – кнопка управления краном по оси Z (основание крана) влево;

13 – кнопка фиксации кнопки 12;

14 – кнопка управления краном по оси Z (основание крана) вправо;

15 – кнопка фиксации кнопки 14;

16 – кнопка управления краном по оси Y (крюк крана) вверх;

17 – кнопка фиксации кнопки 16;

18 – кнопка управления краном по оси Y (крюк крана) вниз;

19 – кнопка фиксации кнопки 18;

20 – кнопка управления краном по оси X (кабина крана) влево;

21 – кнопка фиксации кнопки 20;

22 – кнопка управления краном по оси X (кабина крана) вправо;

23 – кнопка фиксации кнопки 22;

24 – кнопка задания позиции для взятия груза;

25 – кнопка начала автоматической работы;

26 – кнопка захвата груза крюком;

2. Описание программы PLC_PRG(ST):

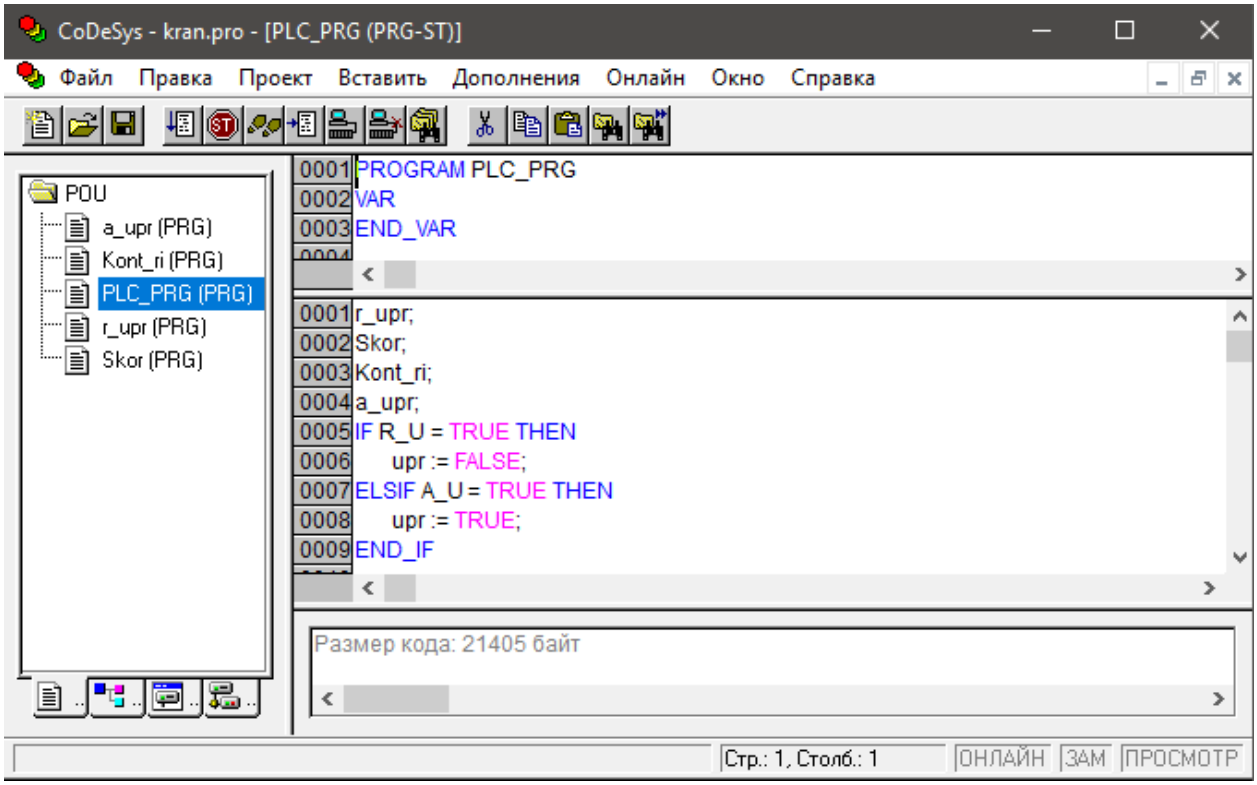


Рисунок 35 – основная программа PLC_PRG

В данной программе осуществляется запуск подпрограмм для работы системы и реализована работа кнопок для переключения между ручным и автоматическим режимом управления.

3. Описание подпрограммы Skor (ST):

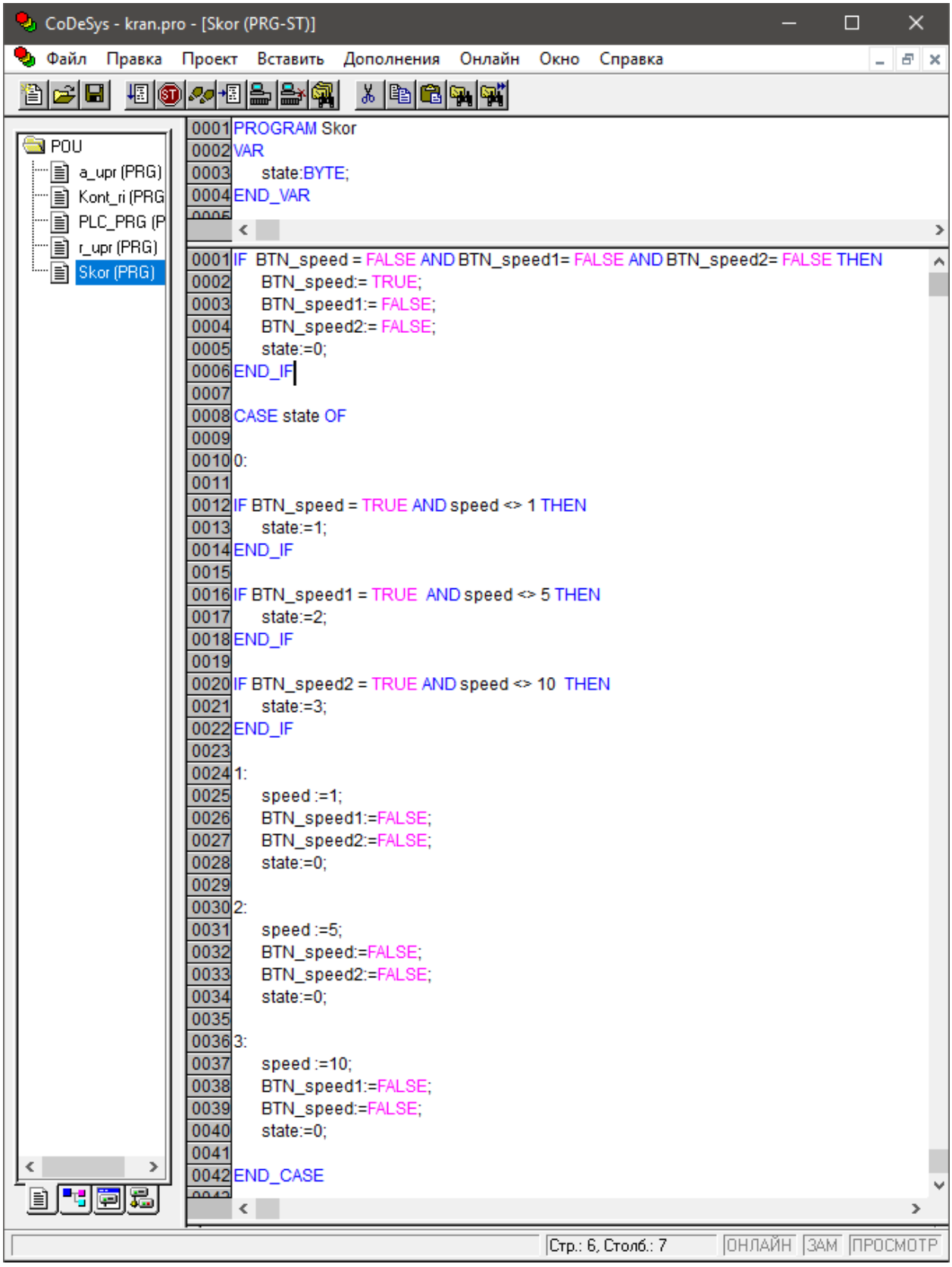


Рисунок 36 – код подпрограммы Skor

В данной подпрограмме реализован выбор скорости движения объектов системы. По главному условие если ни одна из кнопок не была нажата, то включается минимальный режим скорости. А дальше в скорость зависит от того какая кнопка была нажата (малая скорость, средняя или большая).

4. Описание подпрограммы Kont_ri (ST):

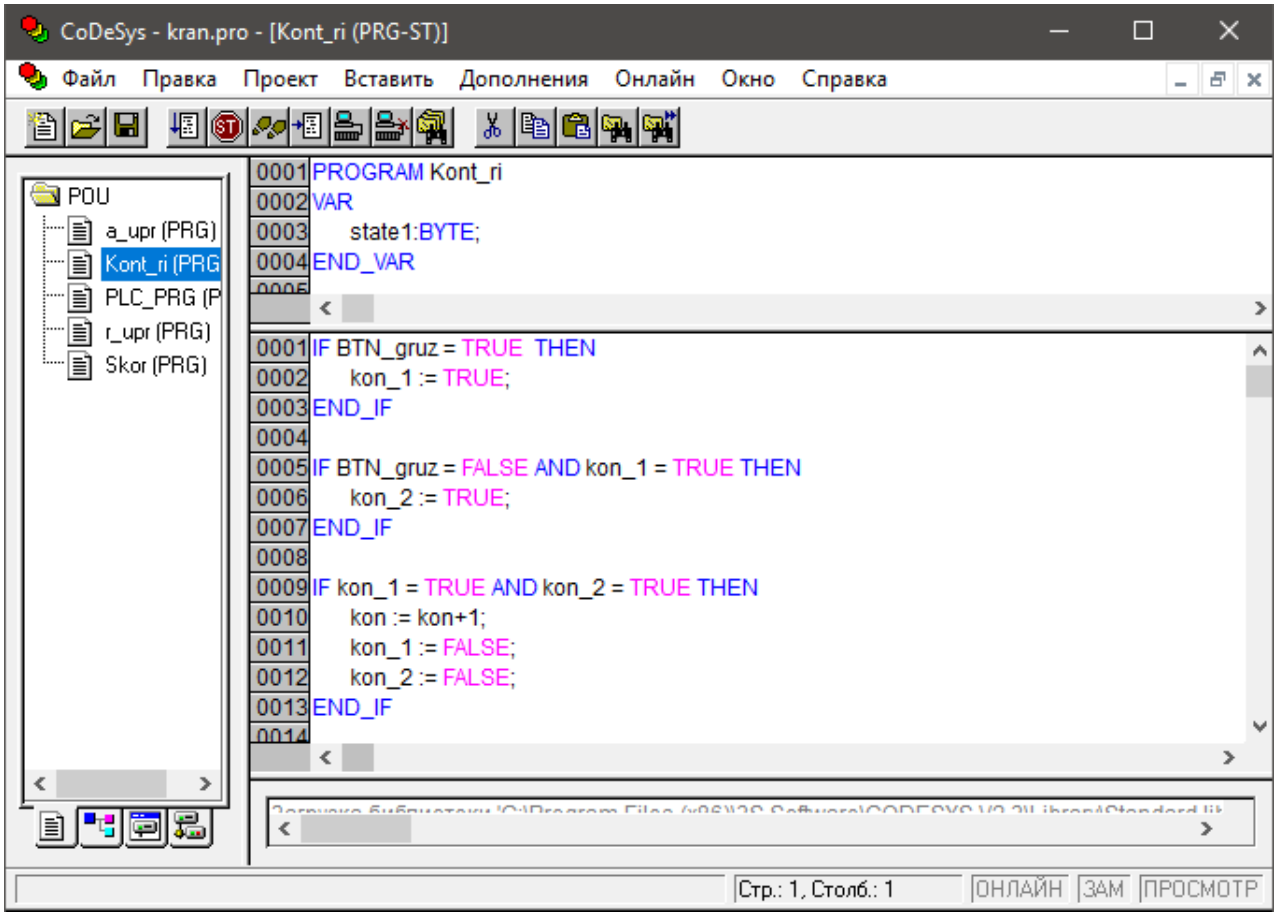


Рисунок 37 – подпрограммы Kont_ri (полный код подпрограммы в приложении Б)

Данная подпрограмма отвечает за появление груза и его взаимодействие с краном, а именно подбор контейнера. В нашем случае контейнеров шесть, каждый имеет свой номер. По условию контейнер можно подобрать только в том случае, когда крюк находится прямо над ним и касается его сверху.

5. Описание подпрограммы r_upr (ST):

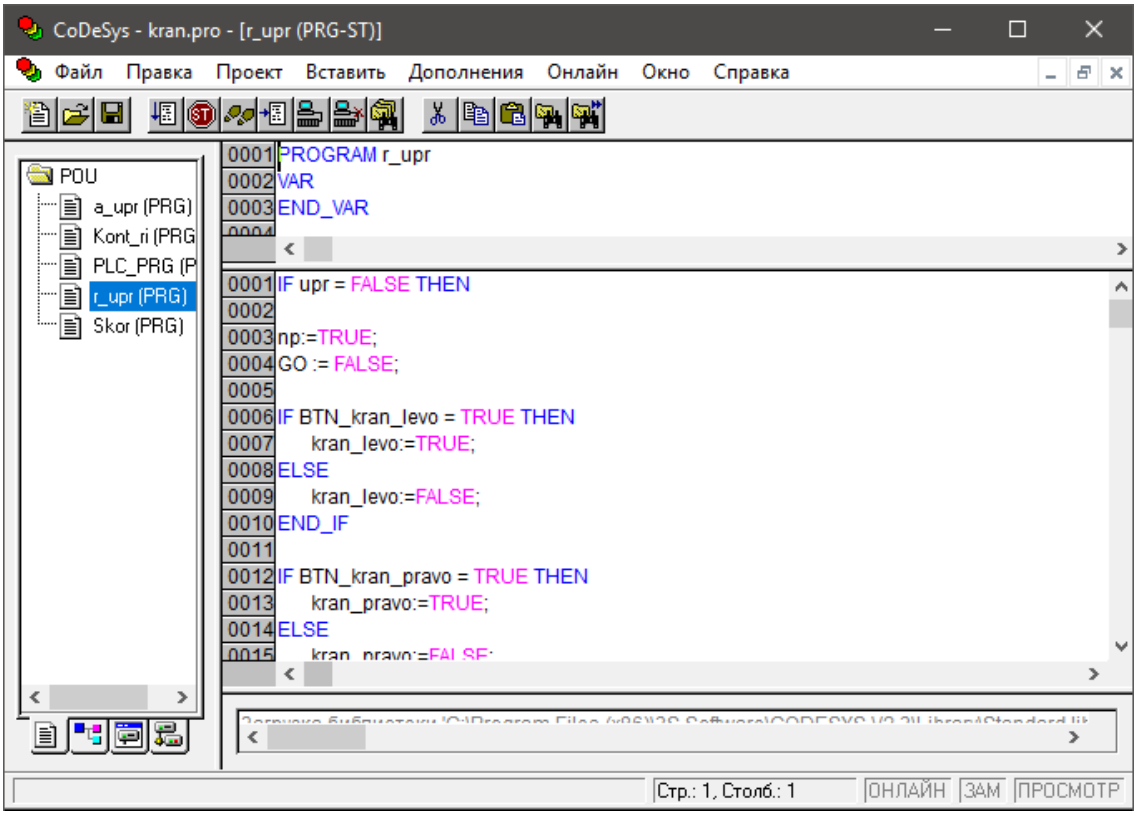


Рисунок 38 – подпрограмма r_upr (полный код подпрограммы в приложении В)

В данной подпрограмме реализовано ручное управление системой, если данный режим выбран. С ее помощью можно лично направить кран в то место, которое необходимо, а также подобрать груз и поставить его там, где необходимо. Так же эта подпрограмма содержит код для управления кнопками задания начального положения и задания положения для взятия груза.

6. Подпрограмма a_upr (ST):

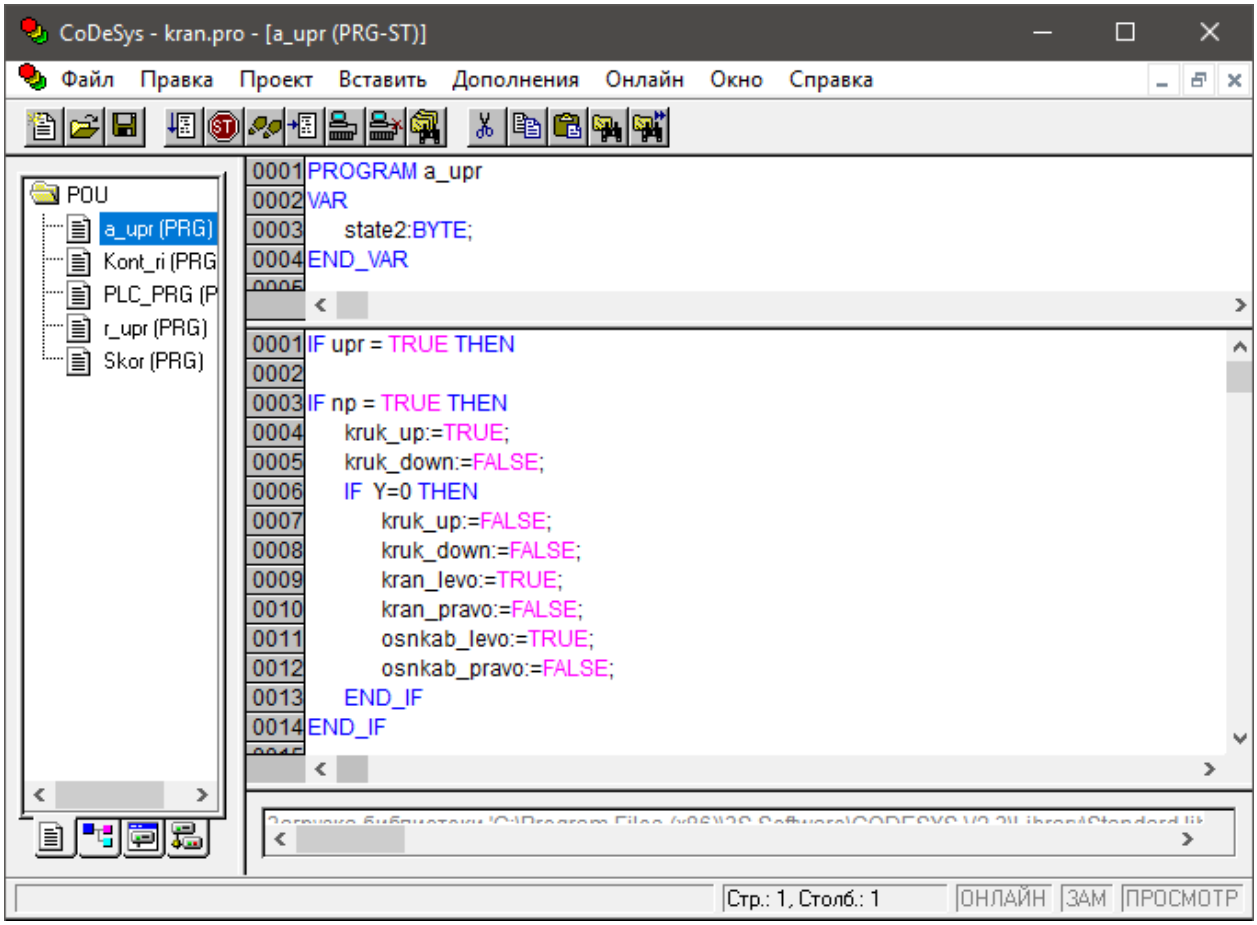


Рисунок 39 – подпрограммы a_upr (полный код подпрограммы в приложении Г)

Подпрограмма реализует автоматическое управление системой, если данный режим выбран. В автоматическом режиме кран самостоятельно расставляет все грузы по своим местам при нажатии на кнопку 25 (начало автоматической работы). После нажатия на кнопку (если имеется груз) кран занимает позицию для подъема контейнера, подцепляет его и переносит на определенное ему место, поставив груз на место, он отпускает его и возвращается в начальное положение, в котором ожидает нового груза и повторного нажатия на кнопку.

2.4 Запуск и проверка работы системы

В первую очередь открываем все программы, а именно MatLab и Simulink модель описанную пунктом выше, CoDeSys и его программу для управления движением системы, PC-эмулятор ПЛК SP PLCWinN.

Загружаем программу из CoDeSys в PC-эмулятор ПЛК и запускаем её. В Simulink во вкладке simulation stop time пишем inf для того чтобы наша модель работала без остановки, после запускаем модель и если все сделано правильно и не возникла проблема с подключением OPC – сервера, то откроется окно с трехмерной модель нашего крана (рисунок 40).

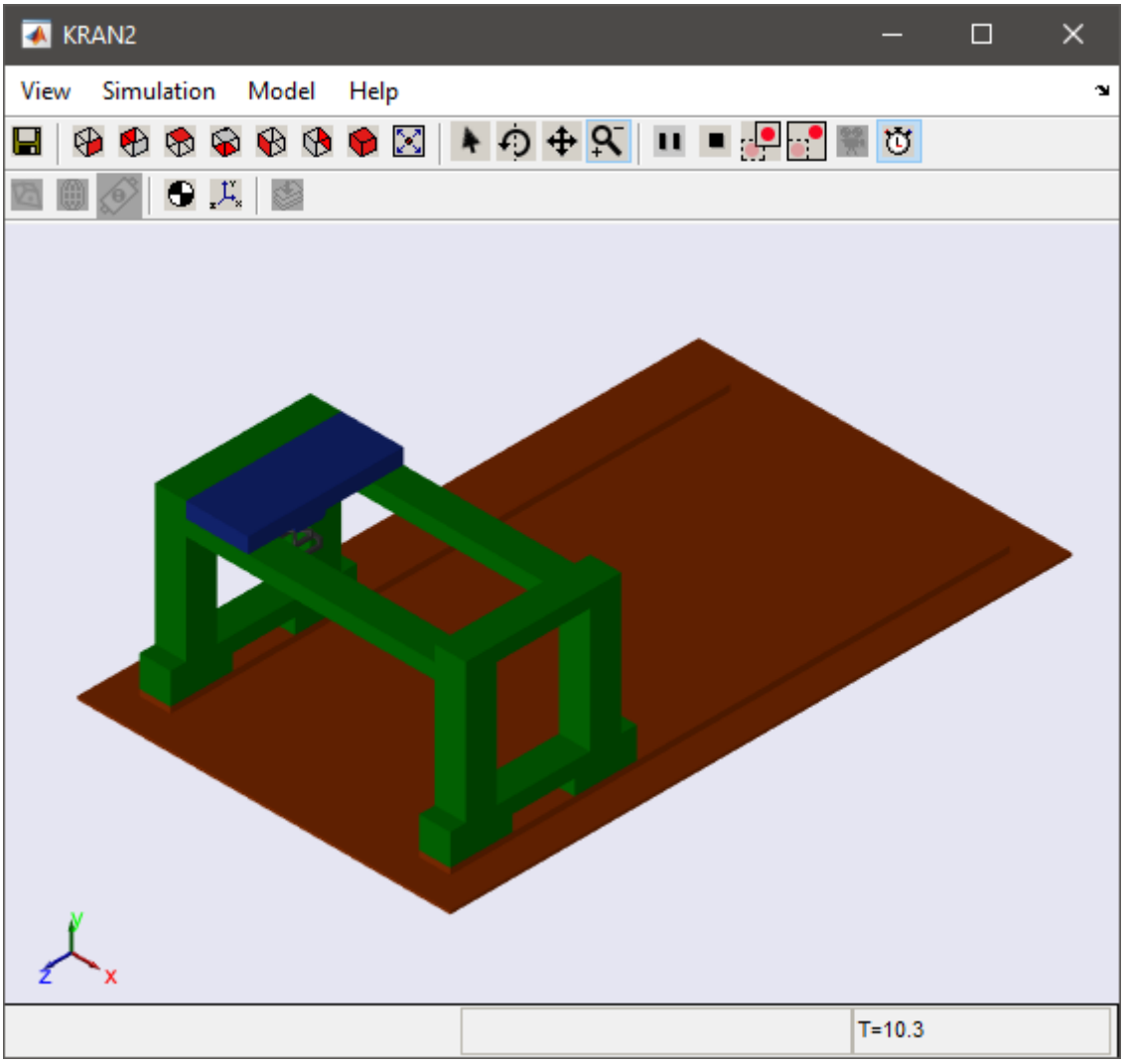


Рисунок 40 – Трехмерная модель козлового крана

Проверим управление движением системы с помощью визуализации в CoDeSys (рисунок 41).

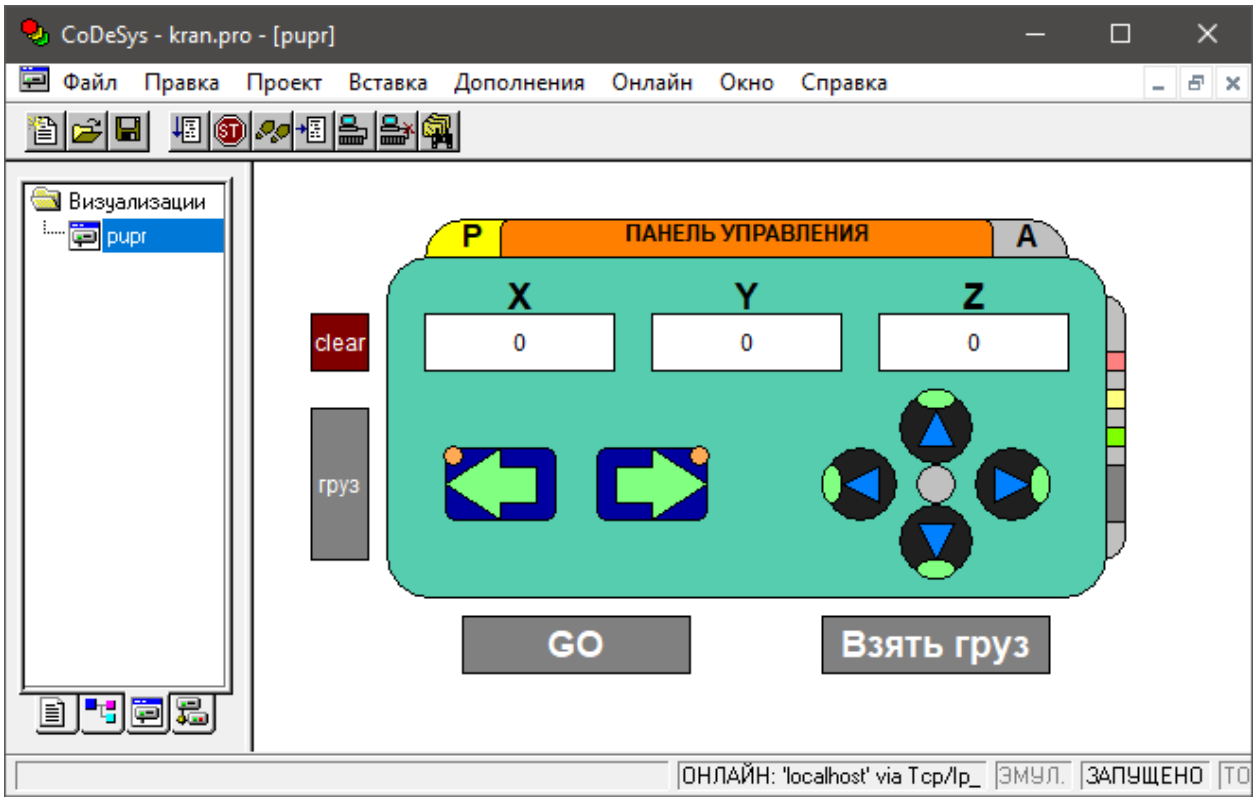


Рисунок 41 – Визуализация панель управления в CoDeSys

Если все кнопки и режимы работают, как задумывалось, то можно сделать вывод что имитационная модель системы управлением движения козлового крана работает правильно.

ЗАКЛЮЧЕНИЕ

В данном курсовом проекте подробно изучили имитационное моделирование систем управления движением. Была проведена работа в таких программах как MatLab, SolodWorks, Simulink и CoDeSys. В качестве примера сделали рабочую модель козлового крана, которая похожа на оригинал и в точности повторяет его движения. Имитационное моделирование можно использовать в разных сферах, что поможет не использовать какой ни будь стенд или оборудование,а сначала проверить программу на модели.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Рыбалев, А.Н. Разработка и эмулирование АСУ ТП с использованием программ разных производителей и типов / А.Н. Рыбалев, Ф.А. Николаец // Вестник Амурского государственного университета. – 2014. – Вып. 65: Сер. Естеств. и экон. науки. – С. 73–82.

2. Ануфриев, И.А. MATLAB7 Наиболее полное руководство [Текст] / И.А. Ануфриев, А.Б. Смирнов, Е.Н. Смирнова // Спб.: БХВ-Петербург, 2005. – 1104 с.

3. Руководство пользователя по программированию ПЛК в CoDeSys 2.3 [Электронный ресурс], –2006. – 158 с. Режим доступа: http://www.kipshop.ru/CoDeSys/steps/codesys_v23_ru.pdf, свободный. – Загл. с экрана.

4. Мусалимов В.М., Г.Б. Заморуев, И.И. Калапышина, А.Д. Перечесова, К.А. Нуждин. Моделирование мехатронных систем в среде MATLAB (Simulink / SimMechanics): учебное пособие для высших учебных заведений. – СПб: НИУ ИТМО, 2013. – 114 с.

5. Блинов О.В. Исследование механических систем в среде SimMechanics (MatLab) с использованием возможностей программ трехмерного моделирования / О.В. Блинов, В.Б. Кузнецов / Методические указания для лабораторного практикума по дисциплинам «Теория механических цепей», «Технология системного моделирования», «Моделирование систем», «Динамические модели сложных систем». Иваново, 2012. – 19 с.

ПРИЛОЖЕНИЕ А

Техническое задание на разработку

Техническое задание разрабатывается согласно требованиям ГОСТ 34.602-89 «Информационная технология».

Техническое задание содержит следующие разделы:

1) введение;

2) назначение и цели создания (развития) системы;

3) характеристика объектов автоматизации;

4) требования к системе;

5) состав и содержание работ по созданию системы;

6) порядок контроля и приемки системы;

7) требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие;

8) требования к документированию;

9) источники разработки.

1) Введение

Данный курсовой проект будет посвящен имитационному моделированию управления движения систем.

Объектом данной разработки является козловой кран.

Модель – это объект, который был создан искусственно с целью упрощенного представления о реальном объекте, процессе или явлении и отражает существенные стороны объекта, который изучается, с точки зрения цели моделирования.

Моделирование — исследование объектов познания на их моделях; построение и изучение моделей реально существующих объектов, процессов или явлений с целью получения объяснений этих явлений, а также для предсказания явлений, интересующих исследователя.

Объект, для которого создают его модель, называют оригиналом или прототипом. Модель не является абсолютной копией своего прототипа, а лишь отражает основные его качества и свойства, которые являются наиболее существенными для выбранной цели исследования. При создании модели всегда имеют место определенные допущения и гипотезы. С помощью системного подхода можно создавать полноценные модели. Исполнители: Алеко М.А , Дорофеева Т.А.

2) Назначение и цели создания (развития) системы

Имитационное моделирование управления систем движения нужно для того что бы изучить движение объекта, как он работает, из чего состоит. Разработка прототипа на основе которого будут создаваться реальные модели объектов, которые будут заменять реальные объекты.

3) Характеристика объектов автоматизации

Козловой кран (КК) – многофункциональное оборудование для различных площадок и объектов, где грузоподъемные работы ведутся постоянно. Козловой кран — это агрегат мостового типа. Горизонтальная балка, по которой перемещается груз — конструктивно относится к мостовой группе подъемных механизмов. Основным отличием является то, что КК может сам передвигаться по рельсам. Усредненный срок эксплуатации козловых кранов — около 20 лет, наработка на отказ ≈ 3 000 циклов.

4) Требование к системе

Нужно что бы моделируемая система была проста и понятна, максимально соответствовала движениям и вела себя, так как объект, который мы моделируем. Система должна реализовать ручное и автоматическое управление.

5) Состав и содержание работ по созданию системы

Разработка трехмерной модели в Solidworks ,перенос этой модели и разработка управляющей программы в Simulink, разработка программы визуализации, апробации, настройка.

6) Порядок контроля и приемки системы

Модель козлового крана, которая должна двигаться по трех осям, поднимать предметы. Модель должна двигаться при помощи команд, которые будут задаваться. Руководитель проекта раз в неделю проводит контроль системы , а приемка системы будет не посредственно на защите

7) Требования к составу и содержанию работ по подготовке объекта автоматизации к вводу системы в действие

Установка программного обеспечения (MatLab (с пакетом Simscape Multiboby Link),Codesys), настройка связи, к составу модели есть Solidworks.

8) Требования к документированию

Описание компонентов систем управления, инструкция к запуску модели, общее описание, описание режимов работ (ручного, автоматического).

9) Источники разработки

1. Рыбалев, А.Н. Разработка и эмулирование АСУ ТП с использованием программ разных производителей и типов / А.Н. Рыбалев, Ф.А. Николаец // Вестник Амурского государственного университета. – 2014. – Вып. 65: Сер. Естеств. и экон. науки. – С. 73–82.

2. Ануфриев, И.А. MATLAB7 Наиболее полное руководство [Текст] / И.А. Ануфриев, А.Б. Смирнов, Е.Н. Смирнова // Спб.: БХВ-Петербург, 2005. – 1104 с.

3. Руководство пользователя по программированию ПЛК в CoDeSys 2.3 [Электронный ресурс], –2006. – 158 с. Режим доступа:

http://www.kipshop.ru/CoDeSys/steps/codesys_v23_ru.pdf, свободный. – Загл. с экрана.

ПРИЛОЖЕНИЕ Б

код подпрограммы Kont_ri

```
IF BTN_gruz = TRUE  THEN
        kon_1 := TRUE;
END_IF

IF BTN_gruz = FALSE AND kon_1 = TRUE THEN
        kon_2 := TRUE;
END_IF

IF kon_1 = TRUE AND kon_2 = TRUE THEN
        kon := kon+1;
        kon_1 := FALSE;
        kon_2 := FALSE;
END_IF

IF kon = 7 THEN
        kon := 0;
END_IF

IF BTN_clear THEN
        gruz := FALSE;
        gruz1 := FALSE;
        gruz2 := FALSE;
        gruz3 := FALSE;
        gruz4 := FALSE;
        gruz5 := FALSE;
        p := FALSE;
        p1 := FALSE;
        p2 := FALSE;
        p3 := FALSE;
        p4 := FALSE;
        p5 := FALSE;
        kon := 0;
END_IF

IF kon = 0 THEN
        state1 := 0;
END_IF
IF kon = 1 THEN
        state1 := 1;
END_IF
IF kon = 2 THEN
        state1 := 2;
END_IF
IF kon = 3 THEN
        state1 := 3;
END_IF
IF kon = 4 THEN
        state1 := 4;
```

```
END_IF
IF kon = 5 THEN
        state1 := 5;
END_IF
IF kon = 6 THEN
        state1 := 6;
END_IF

CASE state1 OF

1:(*----------1----------*)
IF kon = 1 THEN
        gruz := TRUE;
END_IF

IF (kon = 1 AND v = TRUE AND BTN_vziat) OR (upr = TRUE AND vziat  = TRUE) THEN
        vziat := TRUE;
        p := TRUE;
ELSE
        vziat := FALSE;
END_IF

IF X = 15 AND Y = 14.5 AND Z = 0 AND kon = 1 THEN
        v := TRUE;
ELSIF vziat = FALSE AND kon = 1 AND NOT (X = 15 AND Y = 14.5 AND Z = 0) THEN
        v := FALSE;
END_IF



2:(*----------2----------*)
IF kon = 2 THEN
        gruz1 := TRUE;
END_IF

IF (kon = 2 AND v1 = TRUE AND BTN_vziat) OR (upr = TRUE AND vziat1  = TRUE) THEN
        vziat1 := TRUE;
        p1 := TRUE;
ELSE
        vziat1 := FALSE;
END_IF

IF X = 15 AND Y = 14.5 AND Z = 0 AND kon = 2 THEN
        v1 := TRUE;
ELSIF vziat1 = FALSE AND kon = 2 AND NOT (X = 15 AND Y = 14.5 AND Z = 0) THEN
        v1 := FALSE;
END_IF

3:(*----------3----------*)
IF kon = 3 THEN
        gruz2 := TRUE;
END_IF
```

```
IF (kon = 3 AND v2 = TRUE AND BTN_vziat) OR (upr = TRUE AND vziat2  = TRUE) THEN
        vziat2 := TRUE;
        p2 := TRUE;
ELSE
        vziat2 := FALSE;
END_IF

IF X = 15 AND Y = 14.5 AND Z = 0 AND kon = 3 THEN
        v2 := TRUE;
ELSIF vziat2 = FALSE AND kon = 3 AND NOT (X = 15 AND Y = 14.5 AND Z = 0) THEN
        v2 := FALSE;
END_IF

4:(*----------4----------*)
IF kon = 4 THEN
        gruz3 := TRUE;
END_IF

IF (kon = 4 AND v3 = TRUE AND BTN_vziat) OR (upr = TRUE AND vziat3  = TRUE) THEN
        vziat3 := TRUE;
        p3 := TRUE;
ELSE
        vziat3 := FALSE;
END_IF

IF X = 15 AND Y = 14.5 AND Z = 0 AND kon = 4 THEN
        v3 := TRUE;
ELSIF vziat3 = FALSE AND kon = 4 AND NOT (X = 15 AND Y = 14.5 AND Z = 0) THEN
        v3 := FALSE;
END_IF

5:(*----------5----------*)
IF kon = 5 THEN
        gruz4 := TRUE;
END_IF

IF (kon = 5 AND v4 = TRUE AND BTN_vziat) OR (upr = TRUE AND vziat4  = TRUE) THEN
        vziat4 := TRUE;
        p4 := TRUE;
ELSE
        vziat4 := FALSE;
END_IF

IF X = 15 AND Y = 14.5 AND Z = 0 AND kon = 5 THEN
        v4 := TRUE;
ELSIF vziat4 = FALSE AND kon = 5 AND NOT (X = 15 AND Y = 14.5 AND Z = 0) THEN
        v4 := FALSE;
END_IF

6:(*----------6----------*)
IF kon = 6 THEN
```

```
        gruz5 := TRUE;
END_IF

IF (kon = 6 AND v5 = TRUE AND BTN_vziat) OR (upr = TRUE AND vziat5  = TRUE) THEN
        vziat5 := TRUE;
        p5 := TRUE;
ELSE
        vziat5 := FALSE;
END_IF

IF X = 15 AND Y = 14.5 AND Z = 0 AND kon = 6 THEN
        v5 := TRUE;
ELSIF vziat5 = FALSE AND kon = 6 AND NOT (X = 15 AND Y = 14.5 AND Z = 0) THEN
        v5 := FALSE;
END_IF

END_CASE
```

ПРИЛОЖЕНИЕ В
код подпрограммы r_upr

```
IF upr = FALSE THEN

np:=TRUE;
GO := FALSE;

IF BTN_kran_levo = TRUE THEN
        kran_levo:=TRUE;
ELSE
        kran_levo:=FALSE;
END_IF

IF BTN_kran_pravo = TRUE THEN
        kran_pravo:=TRUE;
ELSE
        kran_pravo:=FALSE;
END_IF

IF BTN_osnkab_levo = TRUE THEN
        osnkab_levo:=TRUE;
ELSE
        osnkab_levo:=FALSE;
END_IF

IF BTN_osnkab_pravo = TRUE THEN
        osnkab_pravo:=TRUE;
ELSE
        osnkab_pravo:=FALSE;
END_IF


IF BTN_kruk_up = TRUE THEN
        kruk_up:=TRUE;
ELSE
        kruk_up:=FALSE;
END_IF

IF BTN_kruk_down = TRUE THEN
        kruk_down:=TRUE;
ELSE
        kruk_down:=FALSE;
END_IF

IF X=0 THEN
        BTN_osnkab_levo:=FALSE;
ELSIF X=30 THEN
        BTN_osnkab_pravo:=FALSE;
END_IF
```

```
IF Y=0 THEN
        BTN_kruk_up:=FALSE;
ELSIF Y=14.5 THEN
        BTN_kruk_down:=FALSE;
END_IF

IF Z=0 THEN
        BTN_kran_levo:=FALSE;
ELSIF Z=-60 THEN
        BTN_kran_pravo:=FALSE;
END_IF

(*||||||||||||||||||||||||||||||||||||||||||||||||*)

IF BTN_npol = TRUE THEN
        kruk_up:=TRUE;
        kruk_down:=FALSE;
        IF BTN_npol = TRUE AND Y=0 THEN
                kruk_up:=FALSE;
                kruk_down:=FALSE;
                kran_levo:=TRUE;
                kran_pravo:=FALSE;
                osnkab_levo:=TRUE;
                osnkab_pravo:=FALSE;
        END_IF
END_IF

IF  X=0 AND Y=0 AND Z=0 AND BTN_npol = TRUE THEN
                kruk_up:=FALSE;
                kruk_down:=FALSE;
                kran_levo:=FALSE;
                kran_pravo:=FALSE;
                osnkab_levo:=TRUE;
                osnkab_pravo:=FALSE;
                BTN_npol := FALSE;
END_IF
(*||||||||||||||||||||||||||||||||||||||||||||||||*)

IF BTN_pol = TRUE THEN
        poz_x := 15;
        poz_z := 0;
        poz_y :=14.5;
        kruk_up := TRUE;
        kruk_down:=FALSE;
        IF Y=0 AND BTN_pol =TRUE THEN
                kruk_up := FALSE;
                kruk_down:=FALSE;
                IF BTN_pol =TRUE AND poz_x - X > 0 THEN
                        osnkab_pravo := TRUE;
                        osnkab_levo := FALSE;
                        sign_x_R := TRUE;
                        sign_x_L := FALSE;
```

```
                ELSIF BTN_pol =TRUE AND poz_x - X < 0 THEN
                        osnkab_pravo := FALSE;
                        osnkab_levo := TRUE;
                        sign_x_L := TRUE;
                        sign_x_R := FALSE;
                ELSIF BTN_pol =TRUE AND poz_x - X = 0 THEN
                        osnkab_pravo := FALSE;
                        osnkab_levo := FALSE;
                        sign_x_L := FALSE;
                        sign_x_R := FALSE;
                END_IF

                IF BTN_pol =TRUE AND Z<> 0 THEN
                        kran_levo := TRUE;
                        kran_pravo := FALSE;
                        sign_z_R := TRUE;
                        sign_z_L := FALSE;
                ELSIF BTN_pol =TRUE AND Z = 0 THEN
                        kran_levo := FALSE;
                        kran_pravo := FALSE;
                        sign_z_R := FALSE;
                        sign_z_L := FALSE;
                END_IF
        END_IF
END_IF

IF BTN_pol = TRUE AND X = 15 AND Y=0 AND Z=0 THEN
        kran_levo := FALSE;
        kran_pravo := FALSE;
        sign_z_R := FALSE;
        sign_z_L := FALSE;
        osnkab_pravo := FALSE;
        osnkab_levo := FALSE;
        sign_x_L := FALSE;
        sign_x_R := FALSE;
        kruk_up := FALSE;
        kruk_down := FALSE;
        sign_y_UP := FALSE;
        sign_y_DOWN := FALSE;
        BTN_pol := FALSE;
END_IF

END_IF
```

ПРИЛОЖЕНИЕ Г
код подпрограммы a_upr

```
IF upr = TRUE THEN

IF np = TRUE THEN
        kruk_up:=TRUE;
        kruk_down:=FALSE;
        IF  Y=0 THEN
                kruk_up:=FALSE;
                kruk_down:=FALSE;
                kran_levo:=TRUE;
                kran_pravo:=FALSE;
                osnkab_levo:=TRUE;
                osnkab_pravo:=FALSE;
        END_IF
END_IF

IF  X=0 AND Y=0 AND Z=0  AND np = TRUE THEN
                kruk_up:=FALSE;
                kruk_down:=FALSE;
                kran_levo:=FALSE;
                kran_pravo:=FALSE;
                osnkab_levo:=FALSE;
                osnkab_pravo:=FALSE;
                np := FALSE;
END_IF

END_IF

IF kon = 1 AND GO = TRUE AND np = FALSE THEN
        state2 := 10;
END_IF

IF kon = 2 AND GO = TRUE AND np = FALSE THEN
        state2 := 20;
END_IF

IF kon = 3 AND GO = TRUE AND np = FALSE THEN
        state2 := 30;
END_IF

IF kon = 4 AND GO = TRUE AND np = FALSE THEN
        state2 := 40;
END_IF

IF kon = 5 AND GO = TRUE AND np = FALSE THEN
        state2 := 50;
END_IF
```

```
IF kon = 6 AND GO = TRUE AND np = FALSE THEN
        state2 := 60;
END_IF

CASE state2 OF

10:
        poz_x := 15;
        state2 :=11;

11:
        IF poz_x - X > 0 THEN
                osnkab_pravo := TRUE;
                osnkab_levo := FALSE;
                sign_x_R := TRUE;
                sign_x_L := FALSE;
        ELSIF poz_x - X = 0 THEN
                osnkab_pravo := FALSE;
                osnkab_levo := FALSE;
                sign_x_L := FALSE;
                sign_x_R := FALSE;
        END_IF

        IF X = 15 AND Y=0 AND Z=0 THEN
                state2 := 12;
        END_IF

12:
        IF Y<>14.5 THEN
                kruk_down := TRUE;
                kruk_up := FALSE;
        ELSE
                kruk_down := FALSE;
                kruk_up := FALSE;
                state2 := 13;
        END_IF

13:
        vziat := TRUE;
        kruk_down := FALSE;
        kruk_up := TRUE;
        IF Y = 0 THEN
                kruk_down := FALSE;
                kruk_up := FALSE;
                state2 := 14;
        END_IF

14:
        poz_x := 5;

        IF poz_x - X < 0 THEN
                osnkab_pravo := FALSE;
```

```
            osnkab_levo := TRUE;
            sign_x_R := FALSE;
            sign_x_L := TRUE;
        ELSIF poz_x - X = 0 THEN
            osnkab_pravo := FALSE;
            osnkab_levo := FALSE;
            sign_x_L := FALSE;
            sign_x_R := FALSE;
        END_IF

        kran_levo := FALSE;
        kran_pravo := TRUE;

        IF Z = - 60 THEN
            kran_levo := FALSE;
            kran_pravo := FALSE;
        END_IF

        IF X = 5 AND Y=0 AND Z=-60 THEN
            kruk_down := TRUE;
            kruk_up := FALSE;
        END_IF

        IF Y = 14.5 THEN
            kruk_down := FALSE;
            kruk_up := TRUE;
            vziat := FALSE;
            state2 := 15;
        END_IF

15:

        IF  Y=0 THEN
            kruk_up:=FALSE;
            kruk_down:=FALSE;
            kran_levo:=TRUE;
            kran_pravo:=FALSE;

            poz_x := 15;

            IF poz_x - X > 0 THEN
                osnkab_pravo := TRUE;
                osnkab_levo := FALSE;
                sign_x_R := TRUE;
                sign_x_L := FALSE;
            ELSIF poz_x - X = 0 THEN
                osnkab_pravo := FALSE;
                osnkab_levo := FALSE;
                sign_x_L := FALSE;
                sign_x_R := FALSE;
            END_IF
```

```
            IF X = 15 AND Y=0 AND Z=0 THEN
                  state2 := 0;
            END_IF
      END_IF

20:
      poz_x := 15;
      state2 :=21;

21:
      IF poz_x - X > 0 THEN
            osnkab_pravo := TRUE;
            osnkab_levo := FALSE;
            sign_x_R := TRUE;
            sign_x_L := FALSE;
      ELSIF poz_x - X = 0 THEN
            osnkab_pravo := FALSE;
            osnkab_levo := FALSE;
            sign_x_L := FALSE;
            sign_x_R := FALSE;
      END_IF

      IF X = 15 AND Y=0 AND Z=0 THEN
            state2 := 22;
      END_IF

22:
      IF Y<>14.5 THEN
            kruk_down := TRUE;
            kruk_up := FALSE;
      ELSE
            kruk_down := FALSE;
            kruk_up := FALSE;
            state2 := 23;
      END_IF

23:
      vziat1 := TRUE;
      kruk_down := FALSE;
      kruk_up := TRUE;
      IF Y = 0 THEN
            kruk_down := FALSE;
            kruk_up := FALSE;
            state2 := 24;
      END_IF

24:
      kran_levo := FALSE;
      kran_pravo := TRUE;

      IF Z = - 60 THEN
            kran_levo := FALSE;
```

```
            kran_pravo := FALSE;
        END_IF

        IF X = 15 AND Y=0 AND Z=-60 THEN
            kruk_down := TRUE;
            kruk_up := FALSE;
        END_IF

        IF Y = 14.5 THEN
            kruk_down := FALSE;
            kruk_up := TRUE;
            vziat1 := FALSE;
            state2 := 25;
        END_IF

25:

        IF  Y=0 THEN
            kruk_up:=FALSE;
            kruk_down:=FALSE;
            kran_levo:=TRUE;
            kran_pravo:=FALSE;

            poz_x := 15;

            IF poz_x - X > 0 THEN
                osnkab_pravo := TRUE;
                osnkab_levo := FALSE;
                sign_x_R := TRUE;
                sign_x_L := FALSE;
            ELSIF poz_x - X = 0 THEN
                osnkab_pravo := FALSE;
                osnkab_levo := FALSE;
                sign_x_L := FALSE;
                sign_x_R := FALSE;
            END_IF

            IF X = 15 AND Y=0 AND Z=0 THEN
                state2 := 0;
            END_IF
        END_IF

30:
        poz_x := 15;
        state2 :=31;

31:
        IF poz_x - X > 0 THEN
            osnkab_pravo := TRUE;
            osnkab_levo := FALSE;
            sign_x_R := TRUE;
            sign_x_L := FALSE;
```

```
        ELSIF poz_x - X = 0 THEN
                osnkab_pravo := FALSE;
                osnkab_levo := FALSE;
                sign_x_L := FALSE;
                sign_x_R := FALSE;
        END_IF

        IF X = 15 AND Y=0 AND Z=0 THEN
                state2 := 32;
        END_IF

32:
        IF Y<>14.5 THEN
                kruk_down := TRUE;
                kruk_up := FALSE;
        ELSE
                kruk_down := FALSE;
                kruk_up := FALSE;
                state2 := 33;
        END_IF

33:
        vziat2 := TRUE;
        kruk_down := FALSE;
        kruk_up := TRUE;
        IF Y = 0 THEN
                kruk_down := FALSE;
                kruk_up := FALSE;
                state2 := 34;
        END_IF

34:
        poz_x := 25;

        IF poz_x - X > 0 THEN
                osnkab_pravo := TRUE;
                osnkab_levo := FALSE;
                sign_x_R := TRUE;
                sign_x_L := FALSE;
        ELSIF poz_x - X = 0 THEN
                osnkab_pravo := FALSE;
                osnkab_levo := FALSE;
                sign_x_L := FALSE;
                sign_x_R := FALSE;
        END_IF

        kran_levo := FALSE;
        kran_pravo := TRUE;

        IF Z = - 60 THEN
                kran_levo := FALSE;
                kran_pravo := FALSE;
```

```
        END_IF

        IF X = 25 AND Y=0 AND Z=-60 THEN
                kruk_down := TRUE;
                kruk_up := FALSE;
        END_IF

        IF Y = 14.5 THEN
                kruk_down := FALSE;
                kruk_up := TRUE;
                vziat2 := FALSE;
                state2 := 35;
        END_IF

35:

        IF  Y=0 THEN
                kruk_up:=FALSE;
                kruk_down:=FALSE;
                kran_levo:=TRUE;
                kran_pravo:=FALSE;

                poz_x := 15;

                IF poz_x - X < 0 THEN
                        osnkab_pravo := FALSE;
                        osnkab_levo := TRUE;
                        sign_x_R := FALSE;
                        sign_x_L := TRUE;
                ELSIF poz_x - X = 0 THEN
                        osnkab_pravo := FALSE;
                        osnkab_levo := FALSE;
                        sign_x_L := FALSE;
                        sign_x_R := FALSE;
                END_IF

                IF X = 15 AND Y=0 AND Z=0 THEN
                        state2 := 0;
                END_IF
        END_IF

40:
        poz_x := 15;
        state2 :=41;

41:
        IF poz_x - X > 0 THEN
                osnkab_pravo := TRUE;
                osnkab_levo := FALSE;
                sign_x_R := TRUE;
                sign_x_L := FALSE;
        ELSIF poz_x - X = 0 THEN
```

```
            osnkab_pravo := FALSE;
            osnkab_levo := FALSE;
            sign_x_L := FALSE;
            sign_x_R := FALSE;
        END_IF

        IF X = 15 AND Y=0 AND Z=0 THEN
            state2 := 42;
        END_IF

42:
        IF Y<>14.5 THEN
            kruk_down := TRUE;
            kruk_up := FALSE;
        ELSE
            kruk_down := FALSE;
            kruk_up := FALSE;
            state2 := 43;
        END_IF

43:
        vziat3 := TRUE;
        kruk_down := FALSE;
        kruk_up := TRUE;
        IF Y = 0 THEN
            kruk_down := FALSE;
            kruk_up := FALSE;
            state2 := 44;
        END_IF

44:
        poz_x := 5;
        poz_z := 40;

        IF poz_x - X < 0 THEN
            osnkab_pravo := FALSE;
            osnkab_levo := TRUE;
            sign_x_R := FALSE;
            sign_x_L := TRUE;
        ELSIF poz_x - X = 0 THEN
            osnkab_pravo := FALSE;
            osnkab_levo := FALSE;
            sign_x_L := FALSE;
            sign_x_R := FALSE;
        END_IF

        IF  poz_z + Z > 0 THEN
                kran_levo := FALSE;
                kran_pravo := TRUE;
                sign_z_R := TRUE;
                sign_z_L := FALSE;
            ELSIF poz_z + Z = 0 THEN
```

```
                kran_levo := FALSE;
                kran_pravo := FALSE;
                sign_z_R := FALSE;
                sign_z_L := FALSE;
        END_IF

IF X = 5 AND Y=0 AND Z=-40 THEN
        kruk_down := TRUE;
        kruk_up := FALSE;
END_IF

IF Y = 14.5 THEN
        kruk_down := FALSE;
        kruk_up := TRUE;
        vziat3 := FALSE;
        state2 := 45;
END_IF

45:

IF  Y=0 THEN
        kruk_up:=FALSE;
        kruk_down:=FALSE;
        kran_levo:=TRUE;
        kran_pravo:=FALSE;

        poz_x := 15;

        IF poz_x - X > 0 THEN
                osnkab_pravo := TRUE;
                osnkab_levo := FALSE;
                sign_x_R := TRUE;
                sign_x_L := FALSE;
        ELSIF poz_x - X = 0 THEN
                osnkab_pravo := FALSE;
                osnkab_levo := FALSE;
                sign_x_L := FALSE;
                sign_x_R := FALSE;
        END_IF

        IF X = 15 AND Y=0 AND Z=0 THEN
                state2 := 0;
        END_IF
END_IF

50:
poz_x := 15;
state2 :=51;

51:
IF poz_x - X > 0 THEN
        osnkab_pravo := TRUE;
```

```
            osnkab_levo := FALSE;
            sign_x_R := TRUE;
            sign_x_L := FALSE;
        ELSIF poz_x - X = 0 THEN
            osnkab_pravo := FALSE;
            osnkab_levo := FALSE;
            sign_x_L := FALSE;
            sign_x_R := FALSE;
        END_IF

        IF X = 15 AND Y=0 AND Z=0 THEN
            state2 := 52;
        END_IF

52:
        IF Y<>14.5 THEN
            kruk_down := TRUE;
            kruk_up := FALSE;
        ELSE
            kruk_down := FALSE;
            kruk_up := FALSE;
            state2 := 53;
        END_IF

53:
        vziat4 := TRUE;
        kruk_down := FALSE;
        kruk_up := TRUE;
        IF Y = 0 THEN
            kruk_down := FALSE;
            kruk_up := FALSE;
            state2 := 54;
        END_IF

54:
        poz_z := 40;

        IF  poz_z + Z > 0 THEN
                kran_levo := FALSE;
                kran_pravo := TRUE;
                sign_z_R := TRUE;
                sign_z_L := FALSE;
            ELSIF poz_z + Z = 0 THEN
                kran_levo := FALSE;
                kran_pravo := FALSE;
                sign_z_R := FALSE;
                sign_z_L := FALSE;
            END_IF

        IF X = 15 AND Y=0 AND Z=-40 THEN
            kruk_down := TRUE;
            kruk_up := FALSE;
```

```
        END_IF

        IF Y = 14.5 THEN
                kruk_down := FALSE;
                kruk_up := TRUE;
                vziat4 := FALSE;
                state2 := 55;
        END_IF

55:

        IF  Y=0 THEN
                kruk_up:=FALSE;
                kruk_down:=FALSE;
                kran_levo:=TRUE;
                kran_pravo:=FALSE;

                poz_x := 15;

                IF poz_x - X > 0 THEN
                        osnkab_pravo := TRUE;
                        osnkab_levo := FALSE;
                        sign_x_R := TRUE;
                        sign_x_L := FALSE;
                ELSIF poz_x - X = 0 THEN
                        osnkab_pravo := FALSE;
                        osnkab_levo := FALSE;
                        sign_x_L := FALSE;
                        sign_x_R := FALSE;
                END_IF

                IF X = 15 AND Y=0 AND Z=0 THEN
                        state2 := 0;
                END_IF
        END_IF

60:
        poz_x := 15;
        state2 :=61;

61:
        IF poz_x - X > 0 THEN
                osnkab_pravo := TRUE;
                osnkab_levo := FALSE;
                sign_x_R := TRUE;
                sign_x_L := FALSE;
        ELSIF poz_x - X = 0 THEN
                osnkab_pravo := FALSE;
                osnkab_levo := FALSE;
                sign_x_L := FALSE;
                sign_x_R := FALSE;
        END_IF
```

```
        IF X = 15 AND Y=0 AND Z=0 THEN
                state2 := 62;
        END_IF

62:
        IF Y<>14.5 THEN
                kruk_down := TRUE;
                kruk_up := FALSE;
        ELSE
                kruk_down := FALSE;
                kruk_up := FALSE;
                state2 := 63;
        END_IF

63:
        vziat5 := TRUE;
        kruk_down := FALSE;
        kruk_up := TRUE;
        IF Y = 0 THEN
                kruk_down := FALSE;
                kruk_up := FALSE;
                state2 := 64;
        END_IF

64:
        poz_x := 25;
        poz_z := 40;

        IF poz_x - X > 0 THEN
                osnkab_pravo := TRUE;
                osnkab_levo := FALSE;
                sign_x_R := TRUE;
                sign_x_L := FALSE;
        ELSIF poz_x - X = 0 THEN
                osnkab_pravo := FALSE;
                osnkab_levo := FALSE;
                sign_x_L := FALSE;
                sign_x_R := FALSE;
        END_IF

        IF  poz_z + Z > 0 THEN
                        kran_levo := FALSE;
                        kran_pravo := TRUE;
                        sign_z_R := TRUE;
                        sign_z_L := FALSE;
                ELSIF poz_z + Z = 0 THEN
                        kran_levo := FALSE;
                        kran_pravo := FALSE;
                        sign_z_R := FALSE;
                        sign_z_L := FALSE;
                END_IF
```

```
        IF X = 25 AND Y=0 AND Z=-40 THEN
                kruk_down := TRUE;
                kruk_up := FALSE;
        END_IF

        IF Y = 14.5 THEN
                kruk_down := FALSE;
                kruk_up := TRUE;
                vziat5 := FALSE;
                state2 := 65;
        END_IF

65:

        IF  Y=0 THEN
                kruk_up:=FALSE;
                kruk_down:=FALSE;
                kran_levo:=TRUE;
                kran_pravo:=FALSE;

                poz_x := 15;

                IF poz_x - X < 0 THEN
                        osnkab_pravo := FALSE;
                        osnkab_levo := TRUE;
                        sign_x_R := FALSE;
                        sign_x_L := TRUE;
                ELSIF poz_x - X = 0 THEN
                        osnkab_pravo := FALSE;
                        osnkab_levo := FALSE;
                        sign_x_L := FALSE;
                        sign_x_R := FALSE;
                END_IF

                IF X = 15 AND Y=0 AND Z=0 THEN
                        state2 := 0;
                END_IF
        END_IF

END_CASE
```