



@CODELIBRARY_IT

ISBN 978-5-89971-662-1



9 785899 716621

АРМАВИРСКИЙ ГОСУДАРСТВЕННЫЙ ПЕДАГОГИЧЕСКИЙ УНИВЕРСИТЕТ

fax: 8 (86137) 3 27 39, e-mail: rits_agpu@mail.ru, сайт: rits.agpu.net

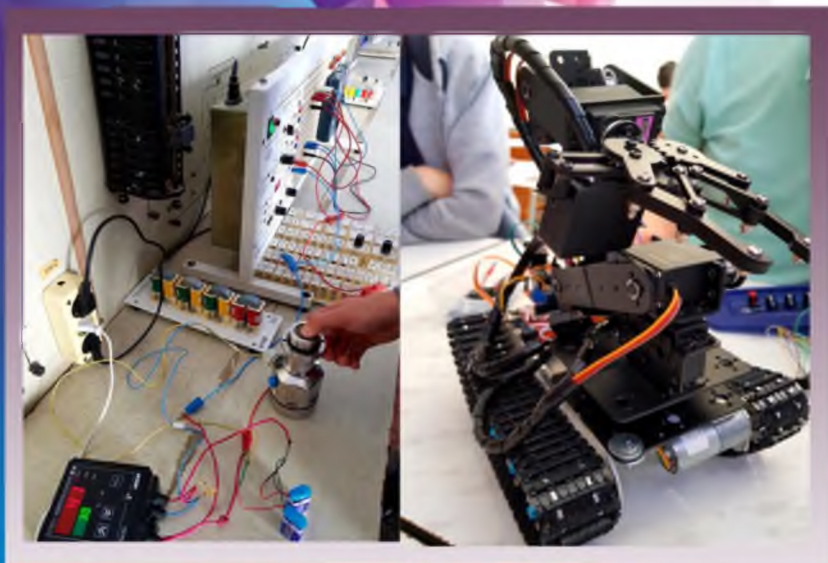
Редакционно-издательский отдел ФГБОУ ВО «Армави́рский государственный педагогический университет»

© АГПУ, 352900, Армавир, ул. Ефремова, 35

Моногаров С. И.

ПРОГРАММИРОВАНИЕ МЕХАТРОННЫХ СИСТЕМ. CODESYS И ARDUINO

Учебное пособие



МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ОБРАЗОВАНИЯ
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»
АРМАВИРСКИЙ МЕХАНИКО-ТЕХНОЛОГИЧЕСКИЙ ИНСТИТУТ
КАФЕДРА ВНУТРИЗАВОДСКОГО ЭЛЕКТРООБОРУДОВАНИЯ И АВТОМАТИКИ

ПРОГРАММИРОВАНИЕ МЕХАТРОННЫХ СИСТЕМ. CODESYS И ARDUINO

*Учебное пособие по изучению дисциплины
«Основы микропроцессорной техники»
для студентов всех форм обучения направления
13.03.02 – Электроэнергетика и электротехника*

Армавир
2019

УДК-004.896
ББК-32.816
П 78

Рецензенты:

Т.Ю. Белозерская – кандидат технических наук, доцент кафедры ВЭА
Армавирского механико-технологического института (филиал)
ФГБОУ ВО «Кубанский государственный технологический университет»

С.К. Давыдов – доцент кафедры ВЭА
Армавирского механико-технологического института (филиал)
ФГБОУ ВО «Кубанский государственный технологический университет»

П 78 Программирование мехатронных систем. CODESYS и Arduino :
учебное пособие по изучению дисциплины «Основы микропроцессорной
техники» для студентов всех форм обучения направления 13.03.02 –
Электроэнергетика и электротехника / авт.-сост. С. И. Моногаров. –
Армавир : РИО АГПУ, 2019. – 152 с.

ISBN 978-5-89971-662-1

В данном учебном пособии обучение предлагается на конкретных
примерах на двух платформах: CODESYS и Arduino.

Рекомендованные к выполнению прикладные программы, как и всё
учебное пособие, могут быть использованы как для студентов, на лаборатор-
ных и практических занятиях, так и на курсах повышения квалификации.

Ил. 54. Библиогр.: 47 назв.

УДК-004.896
ББК-32.816

ISBN 978-5-89971-662-1

© ФГБОУ ВО «Кубанский государственный
технологический университет», 2019
© Оформление. ФГБОУ ВО
«Армавирский государственный
педагогический университет», 2019

СОДЕРЖАНИЕ

Введение.....	5
РАЗДЕЛ 1. Теоретические сведения	6
1.1 Микропроцессоры	6
1.2 Триггер	9
1.3 Базовые логические элементы	11
РАЗДЕЛ 2. Изучение программной среды CODESYS	15
2.1 Первый проект в CODESYS	15
2.2 Основные функциональные блоки в CODESYS	21
2.3 Создание программы реверсивного пуска двигателя в CODESYS	25
2.4 Модернизация токарно-винторезного станка.....	29
2.5 Вариации решений задач по автоматизации производственных процессов в программном комплексе CODESYS	33
2.6 Автоматизация автономной системы подачи воды на животно- водческий комплекс	40
2.7 Автономная система подачи воды с применением пид регулиро- вания, реализованная на ПЛК154	47
РАЗДЕЛ 3. Изучение Arduino	53
3.1 Установка и работа с программным обеспечением	53
3.2 Применение команд Arduino	60
3.3 Частичное замещение элементов схемы реверсивного пускателя на платформе Arduino	65
3.4 Реверс электродвигателя на платформе Arduino	70
3.5 Схема парктроники, реализованная на Arduino	75
3.6 Выводы по платформе Arduino	80
Заключение.....	83

Список использованных источников и литературы.....	84
Приложения.....	91

ВВЕДЕНИЕ

В промышленности все явственнее наблюдается тенденция к объединению различных контролирующих устройств под управлением одного общего устройства, осуществляющего полный контроль и управление всеми механизмами производственных установок.

Новейшие образцы производственных установок часто оборудованы микропроцессорной системой, позволяющей осуществлять одновременный контроль и управление параметрами процесса. Некоторые системы контроля могут быть объединены в систему общего управления производством.

В данном учебном пособии предлагается ознакомление с автоматизацией производственных процессов на конкретных примерах. Обучение предлагается на двух платформах: CODESYS и Arduino. Выбор на них пришелся не случайно. Программные среды бесплатны и доступны для любого желающего. Также имеется у обоих проектов отличная техническая поддержка и множество последователей.

В пособии предлагается сделать первые шаги в создании прикладных программ, устанавливаемых на программируемые логические контроллеры (ПЛК) в CODESYS. Рекомендованные к выполнению прикладные программы, как и все учебное пособие, могут быть использованы на лабораторных и практических занятиях, как для студентов, так и на курсах повышения квалификации.

В приложении даны лабораторные работы для изучения электрических аппаратов и основ релейно-контакторной логики.

РАЗДЕЛ 1. ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1 Микропроцессоры

Появились в середине 30-х годов 19 века /1/.

В 1938 году под руководством Цузе был создан микропроцессор на электромагнитных реле

В 1955 году – транзисторные процессоры, а в начале 60-х появление интегральных микросхем.

В 1971 году Intel выпустила микропроцессор 4004.

Данные и программа их обработки поступают от внешних источников информации; результаты обработки тоже так или иначе воспринимаются внешними устройствами. Кроме того, без ЗУ самому микропроцессору негде хранить программу и некуда посылать результаты промежуточных вычислений. Все вместе: микропроцессор, содержащий АЛУ и УУ, память, необходимые устройства ввода-вывода представляют собой микропроцессорное устройство (МПУ), способное автономно принимать, обрабатывать и передавать информацию. В зависимости от того, какие задачи ставятся перед МПУ, в его состав включают различные УВВ, наделяют памятью того или иного объема, наконец, применяют более или менее мощный микропроцессор. Если МПУ представляет собой персональный компьютер, то в числе УВВ должны быть обязательно устройства связи с пользователем: обычно это клавиатура и дисплей. Такое МПУ требует достаточно большого объема памяти для данных и программ и достаточно мощный (по числу команд и быстродействию) микропроцессор. Если же МПУ управляет каким-либо оборудованием, например, по заданной программе переключает сигналы светофора, ему нужны только соответствующие коммутаторы в качестве УВВ, минимум памяти для несложной программы и самый простейший микропроцессор. Обычно

такие МПУ называются микропроцессорными контроллерами, или просто микроконтроллерами /1/.

Рассмотрим применение микропроцессора в устройстве и работе счётчика «Меркурий 200». Конструктивно счётчик состоит из следующих узлов /2/:

- корпуса (основания корпуса, крышки корпуса, крышки зажимов);
- контактной колодки с датчиком тока (шунт);
- печатной платы модуля электронного;
- толкателей кнопок управления индикацией на корпусе счётчика.

Печатная плата модуля электронного представляет собой плату с электронными компонентами, которая устанавливается в основании корпуса на упоры и закрепляется защёлками. Печатная плата подключается к контактной колодке с помощью проводов.

Структурная схема счётчика приведена на рисунке 1.

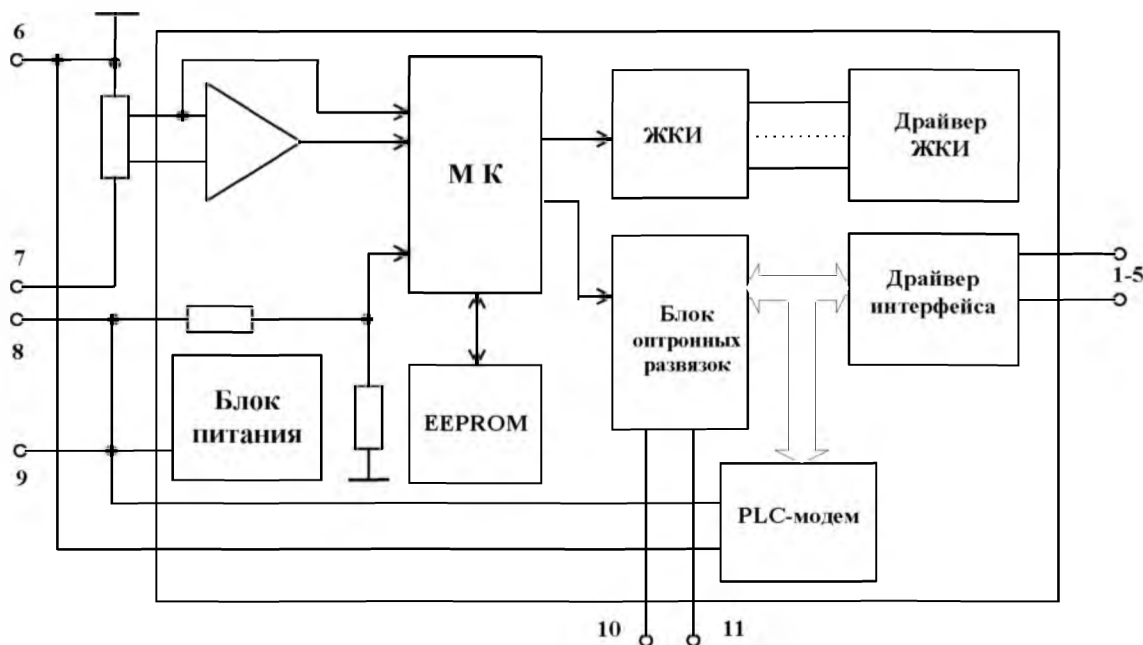


Рисунок 1 - Структурная схема счётчика

На печатной плате находятся:

- микросхема - усилитель сигналов;

- блок питания;
- микроконтроллер (МК);
- энергонезависимое запоминающее устройство;
- элемент резервного питания;
- микросхема драйвера интерфейса;
- PLC-модем (для «Меркурий 200.04» и «Меркурий 200.05»)
- элементы оптронных развязок.

В качестве датчика тока в счётчике используется шунт. В качестве датчика напряжения в счётчике используются резистивный делитель.

Сигналы с датчика тока поступают на вход микросхемы – усилителя сигналов, сигналы с датчика напряжения поступают на аналоговый вход микроконтроллера.

Микроконтроллер производит обработку аналоговых сигналов, поступающих с датчика напряжения и микросхемы-усилителя сигналов, обрабатывает полученные сигналы и посылает полученный результат на жидкокристаллический индикатор для отображения.

МК управляет всеми узлами счётчика и реализует измерительные алгоритмы в соответствии со специализированной программой, помещенной во внутреннюю память программ. Управление узлами счётчика производится через программные интерфейсы, реализованные на портах ввода/вывода МК:

- UART для RS-485 или CAN;
- двухпроводный для PLC;
- I²C интерфейс для связи с энергонезависимой памятью.

МК периодически определяет текущую тарифную зону, формирует импульсы телеметрии, ведет учет энергии и времени, обрабатывает поступившие команды по интерфейсу или модему и, при необходимости, формирует ответ. Кроме данных об учтённой электроэнергии в ОЗУ МК хранятся калибровочные коэффициенты, тарифное расписание, серийный номер, версия программного обеспечения счётчика т.д. Калибровочные коэффициенты заносятся в память на предприятии-изготовителе и защищаются удалением пе-

ремычки разрешения записи. Без вскрытия счётчика и установки перемычки нельзя изменить калибровочные коэффициенты на стадии эксплуатации счётчика.

При отсутствии напряжения питания МК переводится в режим пониженного потребления с питанием от литиевой батареи с напряжением 3 В. Каждую секунду МК переходит в нормальный режим для непрерывного подсчёта времени.

МК синхронизирован внешним кварцевым резонатором, работающим на частоте 32,768 кГц. Установка и коррекция точности хода часов производится программным способом.

МК управляет работой устройства индикации с целью отображения измеренных данных. Индикация может изменяться посредством кнопки управления индикацией.

Для организации связи с внешним управляющим компьютером используется микросхема драйвера интерфейса. Информационные интерфейсные сигналы от МК через опторазвязку поступают на микросхему драйвера интерфейса, работающей на скорости от 600 до 9600 Бод. Сигналы от микросхемы драйвера интерфейса поступают на контакты 2, 3 счётчика. В состав УУ входит микросхема энергонезависимой памяти (EEPROM). Микросхема предназначена для периодического сохранения данных МК. В случае возникновения аварийного режима (“зависание” МК или падение напряжения литиевой батареи) МК восстанавливает данные из EEPROM.

Опрос состояния кнопок управления производится МК на программном уровне.

1.2 Триггер

Триггером называют логическую схему с положительной обратной связью, имеющую два устойчивых состояния, способную скачком переходить из

одного состояния в другое под воздействием внешних сигналов. Эти состояния триггера называются единичным и нулевым и соответствуют логическому уровню на прямом выходе триггера. Перевод триггера в единичное состояние называют установкой (включением), для чего воздействуют на вход S (от Set - установка). При воздействии на вход R (от Reset - сброс) триггер обнуляется (сбрасывается).

Простейший RS-триггер получается, если включить два инвертирующих элемента, например, 2ИЛИ - НЕ, как показано на рисунке 2.

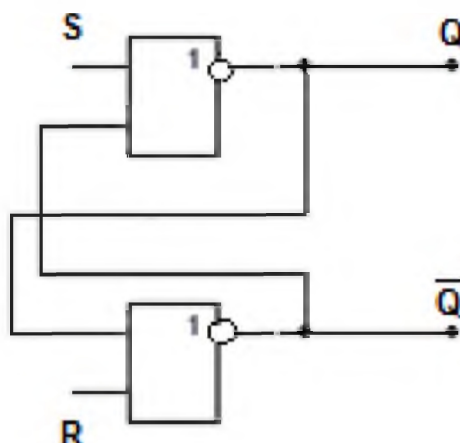


Рисунок 2 - RS-триггер на элементах 2ИЛИ-НЕ

Пока на обоих управляющих входах R и S уровни сигналов не активны, триггер находится в каком либо одном из двух устойчивых состояний. В таблице 1 приведены все возможные сочетания входных сигналов и их воздействие на выход Q в n+1 такте с учетом собственного состояния триггера в предыдущем n-ом такте.

Таблица 1 Таблица состояний RS-триггера

R	S	Q_n	Q_{n+1}	Режим	R	S	Q_n	Q_{n+1}	Режим
0	0	0	0	Хранение	1	0	0	0	Сброс
0	0	1	1	Хранение	1	0	1	0	Сброс
0	1	0	1	Установка	1	1	0	X	Запрещено
0	1	1	1	Установка	1	1	1	X	Запрещено

Ситуация $S=R=1$ считается запрещенной из-за своей неопределенности после снятия входных сигналов. Это отражено в двух последних строках таблицы.

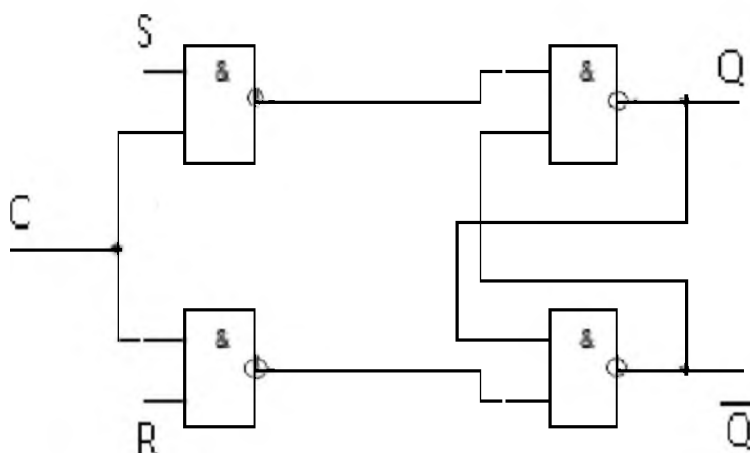


Рисунок 3 - Синхронный RS-триггер

Изменение состояний выходов триггера возможно только при совпадении сигналов управления и синхронизации.

1.3 Базовые логические элементы

В цифровых устройствах широко применяются элементы, которые выполняют определенные логические операции. Такие элементы называются логическими элементами (ЛЭ).

Для описания логических операций используется математический аппарат, получивший название алгебры логики или булевой алгебры. Алгебра логики изучает взаимосвязь между простыми высказываниями, образующими сложные высказывания. С точки зрения алгебры логики простое высказывание может иметь только два значения - истинное или ложное. Одно из этих значений принимается за 1, второе - за 0. То есть простое высказывание является двоичной переменной.

Основными логическими операциями являются:

Операция логического отрицания НЕ (инверсия) преобразует истинное высказывание в ложное или наоборот. Символически операция логического отрицания обозначается в виде черточки над аргументом: $y = \bar{x}$. Такое выражение читается: «у равен НЕ x».

Операция логического отрицания имеет два исхода в зависимости от значения аргумента: а) если $x=0$, то $y=1$ и б) если $x=1$, то $y=0$.

Операция логического сложения ИЛИ (дизъюнкция) используется для образования сложного высказывания из простых. При этом сложное высказывание будет истинным, если истинно хотя бы одно из простых высказываний, и ложным, если ложны все простые высказывания. Символически операция ИЛИ обозначается выражениями:

$$y = x_1 + x_2 + x_3 + \dots \text{ или } y = x_1 \vee x_2 \vee x_3 \vee \dots$$

Читается: «у равен x_1 или x_2 или $x_3 \dots$ »

Результаты логической операции ИЛИ для всех возможных комбинаций двух аргументов приведены в таблице 2.

Операция логического умножения И (конъюнкция) тоже используется для образования сложного высказывания из простых, но при этом сложное высказывание считается истинным тогда и только тогда, когда истинны все простые высказывания.

Символически операция И обозначается выражениями:

$$y = x_1 \cdot x_2 \cdot x_3 \cdot \dots \text{ или } y = x_1 \wedge x_2 \wedge x_3 \wedge \dots$$

Читается: «у равен x_1 и x_2 и $x_3 \dots$ »

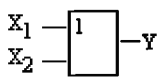
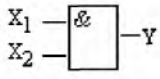
Логические элементы работают с двоичным кодированием информации, которое характеризуется двумя уровнями напряжения двоичной переменной, обозначаемыми цифрами 1 (высокий) и 0 (низкий уровень). В зависимости от уровня напряжения, при котором воспринимается или вырабатывается информация, различают прямые и инверсные входы и выходы ЛЭ. Прямым считается такой вход (или выход), на котором двоичная переменная имеет значение 1, когда уровень напряжения на этом входе (выходе) соответ-

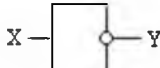
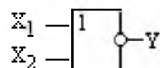
ствует состоянию, принятому за 1. Если двоичная переменная на входе (выходе) имеет значение 1 при уровне напряжения на нем, соответствующем состоянию, принятому за 0, то такой вход (выход) называется инверсным.

Логические элементы обычно выполняются на ИМС, в которых используется положительная (позитивная) логика, когда логической единице соответствует высокий потенциальный уровень, а логическому нулю — низкий потенциальный уровень. Если логической единице соответствует низкий потенциальный уровень, а логическому нулю — высокий, такую логику называют отрицательной, или негативной. Очевидно, что входы и выходы, являющиеся прямыми в положительной логике, будут инверсными в отрицательной логике, и наоборот.

На принципиальных схемах ЛЭ изображаются в виде прямоугольника (таблица 2). В верхней части прямоугольника указывается символ функции: 1 - для логической функции ИЛИ и & - для логической функции И. Входы и выходы изображаются линиями, проведенными перпендикулярно к боковым сторонам прямоугольника (входы - с левой стороны, выходы - с правой). Если вход или выход является инверсным, то в месте пересечения изображающей его линии со стороной прямоугольника ставится кружок. В таблице 2 приведены буквенные и графические обозначения основных логических элементов, их таблицы истинности и примеры технической реализации.

Таблица 2 Простейшие логические элементы.

Тип операций	Графическое обозначение элемента	Таблица истинности		
ИЛИ		X1	X2	Y
		1	0	1
		0	1	1
		1	1	1
		0	0	0
И		X1	X2	Y
		1	1	1
		1	0	0
		0	1	0
		0	0	0

НЕ		<table><tr><td>X</td><td>Y</td></tr><tr><td>1</td><td>0</td></tr><tr><td>□</td><td>1</td></tr></table>	X	Y	1	0	□	1												
X	Y																			
1	0																			
□	1																			
И-НЕ		<table><tr><td>X1</td><td>X</td><td>Y</td></tr><tr><td></td><td>□</td><td></td></tr><tr><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>0</td><td>0</td><td>1</td></tr></table>	X1	X	Y		□		1	1	0	1	0	1	0	1	1	0	0	1
X1	X	Y																		
	□																			
1	1	0																		
1	0	1																		
0	1	1																		
0	0	1																		
ИЛИ-НЕ		<table><tr><td>X1</td><td>X2</td><td>Y</td></tr><tr><td>1</td><td>1</td><td>0</td></tr><tr><td>1</td><td>□</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>0</td><td>0</td><td>1</td></tr></table>	X1	X2	Y	1	1	0	1	□	0	0	1	0	0	0	1			
X1	X2	Y																		
1	1	0																		
1	□	0																		
0	1	0																		
0	0	1																		

Вопросы для самопроверки:

1. Из чего состоит процессор?
2. Назовите недостаток одноплатного контроллера.
3. По каким направлениям идет применение микропроцессорной техники в электроаппаратостроении?
4. Поясните структурную схему устройства пуска асинхронного двигателя.

РАЗДЕЛ 2. ИЗУЧЕНИЕ ПРОГРАММНОЙ СРЕДЫ CODESYS

2.1 Первый проект в CODESYS

CODESYS – это программное обеспечение, позволяющее разрабатывать прикладные программы, устанавливающиеся на программированные логические контроллеры (ПЛК) /15/. Рассмотрим поэтапно инструкцию по установке данного программного комплекса.

1. Скачиваем файл установки программы:

2. Для скачивания приложения переходим на официальный сайт OWEN – **www.owen.ru**. Затем переходим по вкладкам:

3. «*Продукция / Программное обеспечение / Среда программирования CODESYS / CODESYS v.3*». На данной странице находим русифицированную сборку программы и скачиваем (см. рис. 4).

4. Устанавливаем программное обеспечение:

Запускаем файл установки программы. В открывшемся окне поочередно нажимаем на кнопки:

«*Next – Yes – Next – Yes – Next – Next – Next – Next*»

После сделанных нажатий программа начинает установку. После нажимаем на кнопку «*Finish*».

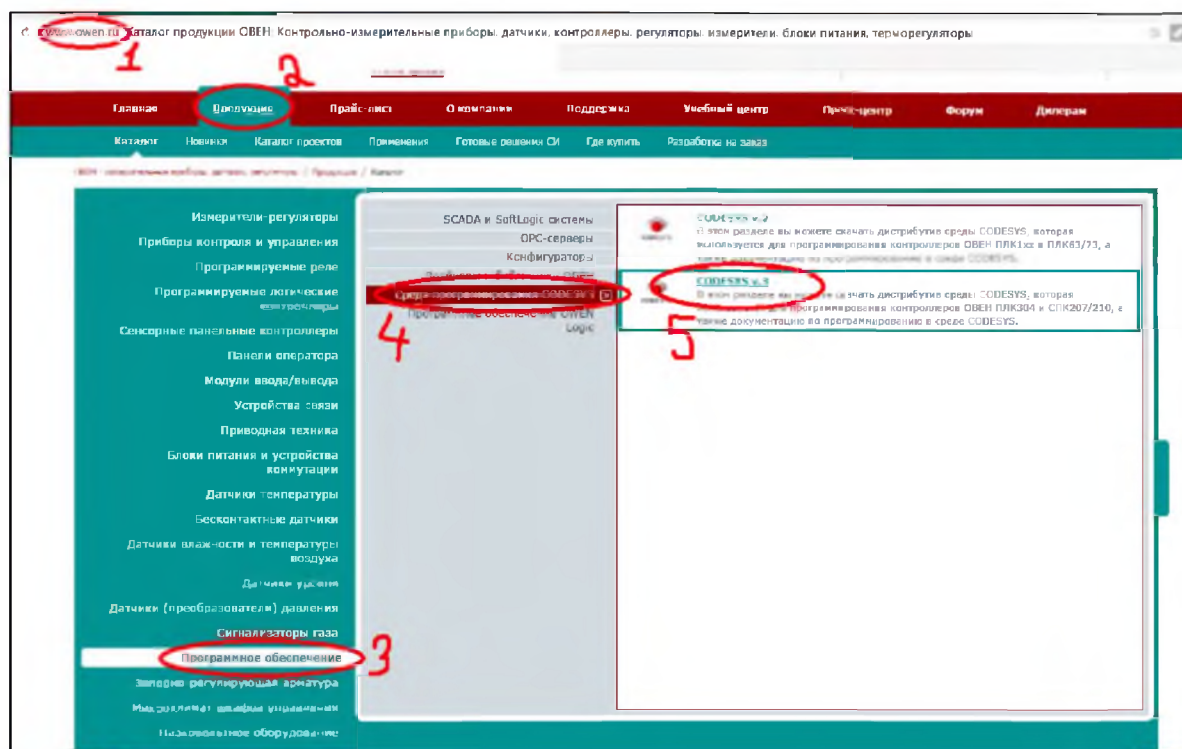


Рисунок 4 – Шаги для скачивания программного обеспечения

5. Начальные настройки программы:

Запускаем программу, переходим по вкладки «*Файл – Создать*». В открывшемся окне выбираем тип ПЛК, на который будем ставить программу. Мы выбрали PLC63-M:

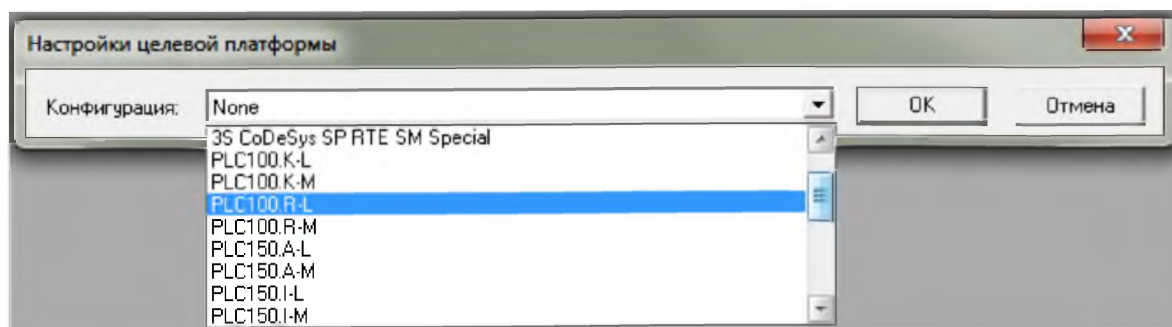


Рисунок 5 – Окно выбора целевой платформы

Выбрав, нажимаем на кнопку «OK». Далее, в появившемся окне выбираем язык программирования. Мы будем использовать язык CFC. После выбора нажимаем «OK» (см. рис. 5).

6. Создание программы:

После правильного выполнения предыдущих пунктов, перед нами окажется рабочая область программирования, показанная на рисунке 6. На панели управления находятся значки, позволяющие редактировать программу, а также блоки, для создания самой программы, использующиеся в языке CFC. Каждая кнопка описывается программой при наведении на нее курсора.

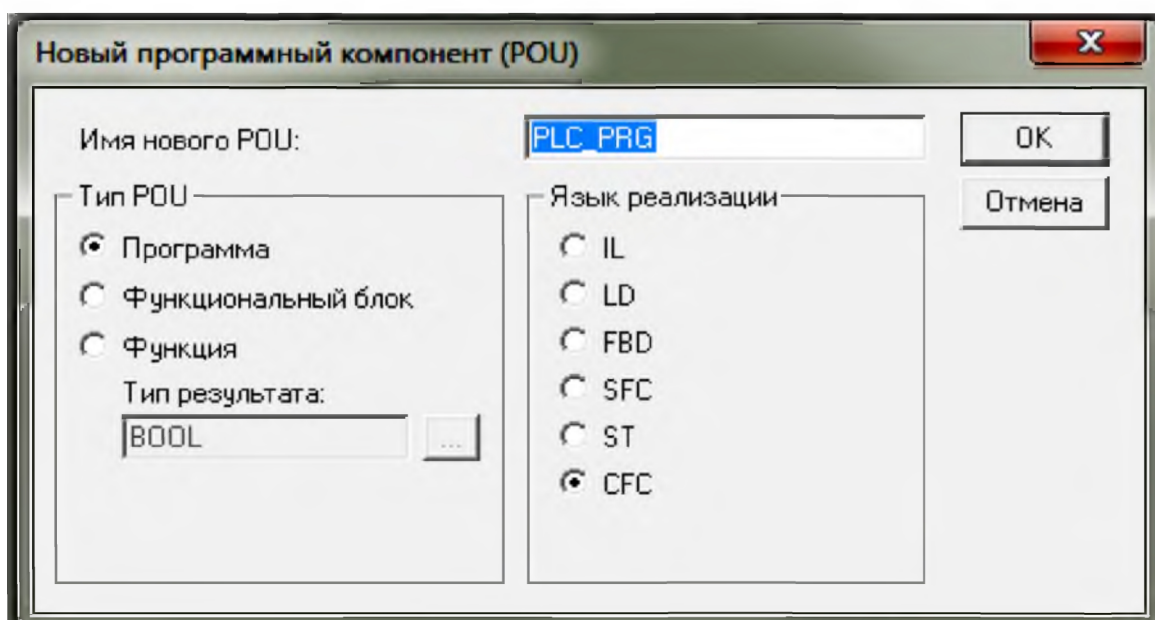


Рисунок 6 – Окно выбора языка программирования

Для примера мы создадим схему реверсивного пускателя (см. рис. 7). Она будет содержать две кнопки пуска, которые будут включать двигатель, и для защиты от одновременного его включения в обе стороны будет предусмотрена защита.

роль двух катушек пускателя. После выполненных настроек переходим обратно на вкладку «POU» и выбираем из списка нашу программу. Далее с помощью блоков создаем схему:

- 1) Блоки «Вход» – задает входную величину;
- 2) Блоки «Выход» - задаёт выходную величину;
- 3) Блок «XOR» – выдает на выходе 1, если на вход поступает только один сигнал 1.
- 4) Блок «AND» – выдаёт на выходе 1, если все значения на входе равны 1.
- 5) Блок «BOOL_TO_REAL» – преобразует тип входного сигнала из логического в число с плавающей запятой.

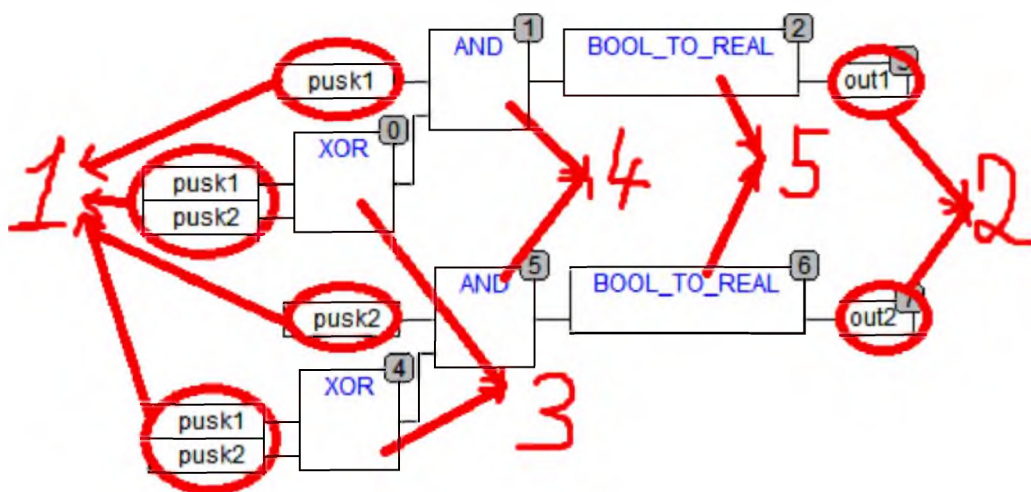


Рисунок 9 – Схема программы

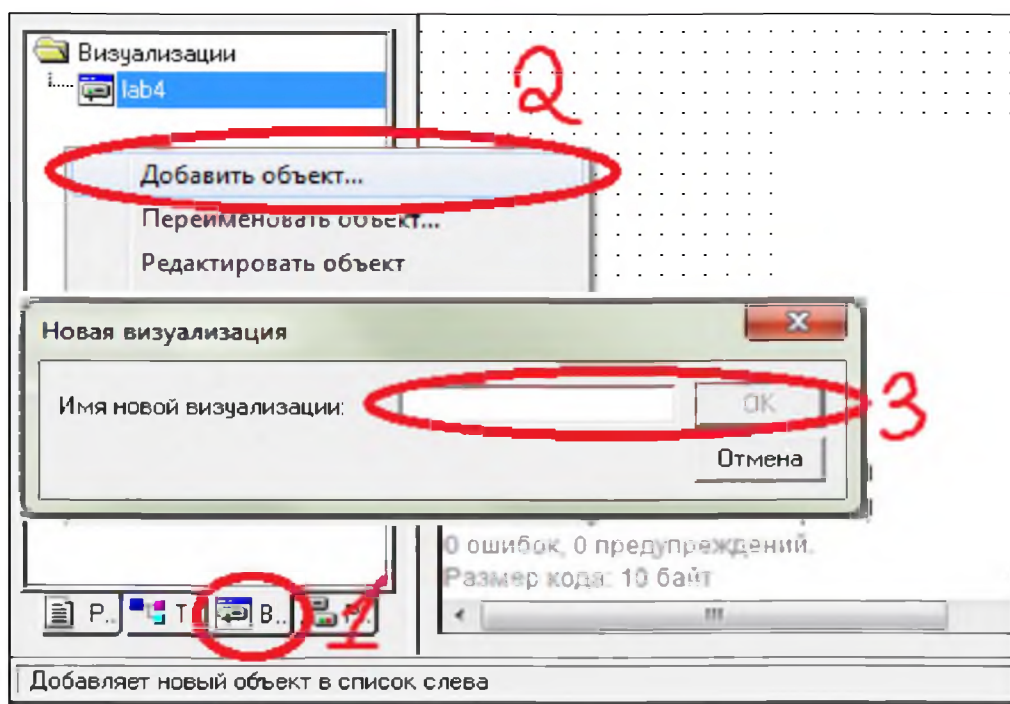


Рисунок 10 – Окно создания файла визуализации

После создания программы, создадим её визуализацию, для этого перейдем во вкладку «Визуализация», как показано на рисунке 10, щелкаем правой кнопкой мыши и выбираем пункт «Добавить объект». После даем название в появившемся окне и нажимаем «ОК».

В появившемся окне рисуем два прямоугольника (две кнопки) и две окружности (показывающие направления двигателя) с помощью блоков, расположенных вверху на панели управления. Для того чтобы задать прямоугольнику функцию кнопки, необходимо щелкнуть по нему левой кнопкой мыши два раза, в открывшемся окне перейти во вкладку «Ввод» и поставить галочку напротив «Пер-я переключения», в окне, расположенном левее, задать имя входа.

Для этого нажимаем клавишу «F2» и выбираем нужный вход, после подтверждаем. Те же действия проделываем для окружности: кликаем левой кнопки мыши два раза, переходим во вкладку «Переменные» и в окне изменения цвета выбираем необходимый выход, также через клавишу «F2» на клавиатуре.

В конфигурациях элемента можно задать цвет фигуры, набрать текст внутри неё и воспользоваться многими другими настройками.

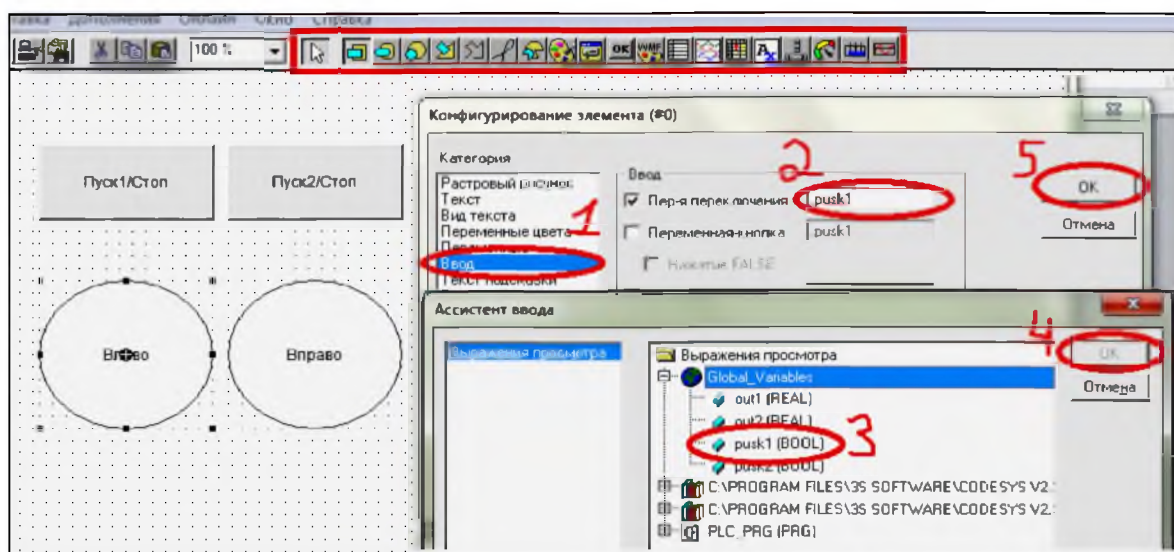


Рисунок 11 – Создание визуализации и ее настройка

Для запуска программы необходимо одновременно нажать клавиши Alt+F8 и затем F5.

Были рассмотрены только некоторые функции данного программного комплекса, при создании других проектов, возможно, могут понадобиться другие элементы программы, описание их работы можно найти в справке, но принцип их использования остаётся такой же.

2.2 Основные функциональные блоки в CODESYS

На большинстве производств есть десятки или даже сотни машин от очень простых до сложных. Некоторыми по-прежнему управляют только квалифицированные рабочие, однако, большинство сложных машин полагается на контроллеры с программированной логикой, сокращенно ПЛК. Это промышленный компьютер, который используется для управления машинами и процессами. Они отличаются от офисных компьютеров по типам задач,

которые они выполняют, а также аппаратным и программным обеспечением, которое они требуют. Несмотря на то, что определённые приложения значительно различаются, все ПЛК принимают входные сигналы от переключателей и датчиков, принимают решение на основе сохранённой программы и управляют выходными для автоматизации машин и процессов. Для программирования данных контроллеров используют различное программное обеспечение, одним из которых является CODESYS. Пользователи имеют возможность на базе этой программы использовать несколько языков программирования:

- IL
- LD
- FBD
- SFC
- ST
- CFC

Далее мы рассмотрим язык CFC. Он используется для программирования систем управления непрерывными технологическими процессами. CFC представляет собой библиотеку функциональных блоков, имеющих свои входы и выходы и возможность настройки индивидуальных параметров. Ранее было показано поэтапная установка программного обеспечения CODESYS. Сейчас мы рассмотрим основные блоки и их функции.

Первые блоки, которые мы рассмотрим – это “вход” и “выход”. Они нужны для того, чтобы задавать входную и выходную переменные соответственно.



Рисунок 12 – Блоки входа и выхода

Для того чтобы задать в вашем алгоритме ту или иную логическую операцию, необходимо воспользоваться блоком “Элемент”.

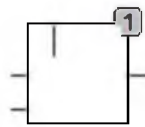


Рисунок 13 – Блок “элемент”

По умолчанию данный блок будет выполнять логическую операцию И (AND). Для того чтобы получить другой элемент необходимо поменять названия блока.

Так, например, написав слова “OR” мы получим блок, выполняющий логическую операцию ИЛИ. По такому же принципу можно получить блок НЕ, написав команду “NOT” и исключающее или – команда “XOR”.

Все рассмотренные функциональные блоки принимают и передают переменную “BOOL”, то есть имеющие три состояния “true”, “false”, “null”.

Для примера рассмотрим рисунок 14, на котором показаны четыре логические операции “AND”, “OR”, “XOR”, “NOT”. Создадим переменные “a=1” и “b=0”. На рисунке показано, что при входе 1 и 0 на выходе у блока “AND” – 0, у “OR” – 1 и у “XOR” – 1, блок “NOT” выдает противоположное входной переменной значение.

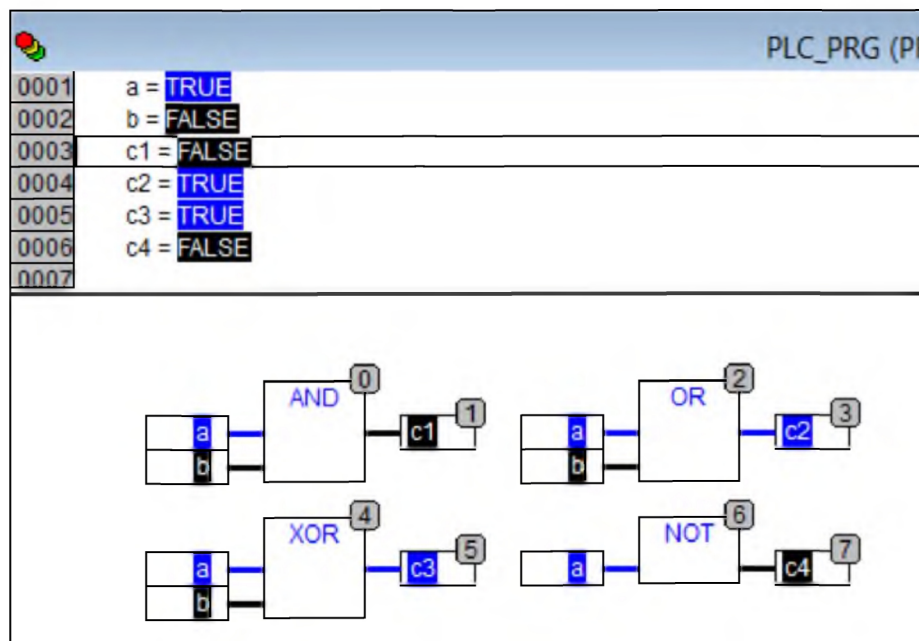


Рисунок 14 – Основные логические операции в CODESYS

Также хотелось бы рассмотреть еще два функциональных блока – это “RS” и “SR” триггеры.

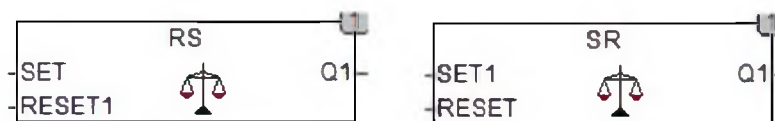


Рисунок 15 – “RS” и “SR” триггеры

Для того чтобы добавить один из триггеров на рабочую область необходимо выбрать блок “элемент”, как это делалось ранее, и прописать в нем команду “RS” или “SR”.

Принцип их работы достаточно прост на вход “SET” поступает команда установки, а на вход “RESET” команда сброса. Различия данных триггеров в том, что в первом приоритет идет на вход сброса, а на втором на вход установки. То есть, если на два входа поступает одновременно сигнал “true”, то на выходе в “RS” триггере будет “false”, а в “SR” триггере “true”, как показано на рисунке 16.

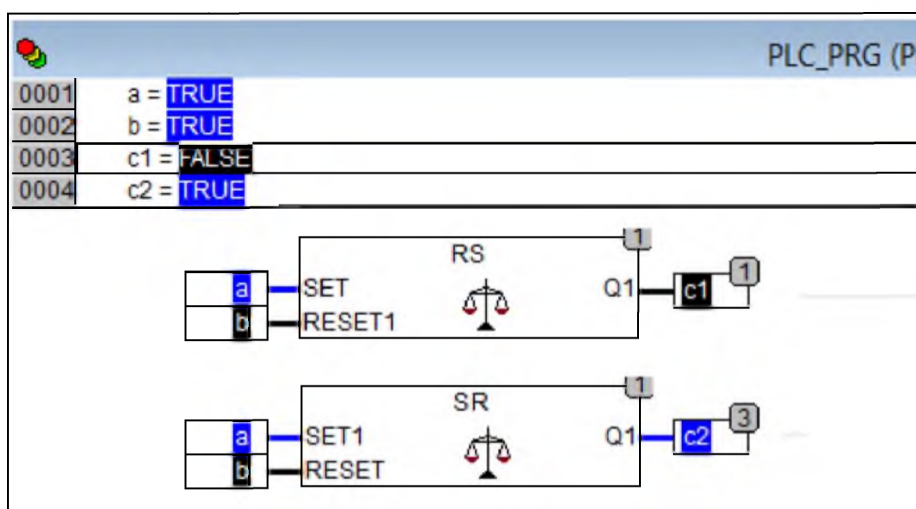


Рисунок 16 – Работа “RS” и “SR” триггеров

Триггеры также как и блоки логических операций принимают на вход и передают переменные “BOOL”. Для того чтобы перевести переменную “BOOL” в “REAL”, это может потребоваться при работе с некоторыми блоками, или наоборот, можно воспользоваться блоком “элемент” и прописать в

нем команду “BOOL_TO_REAL” или “REAL_TO_BOOL”, приведенные на рисунке 17.

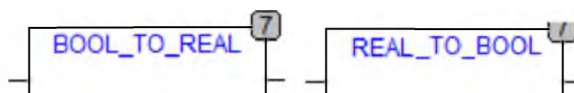


Рисунок 17 – Блоки для изменения типа данных

Используя рассмотренные ранее функциональные блоки в CODESYS, возможно создать различные программы автоматизации технологических процессов.

2.3 Создание программы реверсивного пуска двигателя в CODESYS

Сейчас предстоит создать программу реверсивного пуска двигателя, а также покажем привязку ее к ПЛК.

Для начала запустим программное обеспечение CODESYS, затем выберем в верхнем левом углу вкладку «Файл – Создать». В открывшемся окне выбираем тип ПЛК, на который будем ставить программу. Мы выбрали PLC63-M. Выбрав, нажимаем на кнопку «OK». Далее, в появившемся окне выбираем язык программирования. Будем использовать язык SFC. После выбора нажимаем «OK».

Перед нами появилась рабочая область для создания программы. Перед тем, как начать программирование нам необходимо разобраться в схеме реверсивного пуска двигателя (рис. 18). А, В, С – три фазы. QF – автоматический выключатель. КМ1 и КМ2 – катушки пускателя на 380 В и их контакты. Р – тепловое реле.

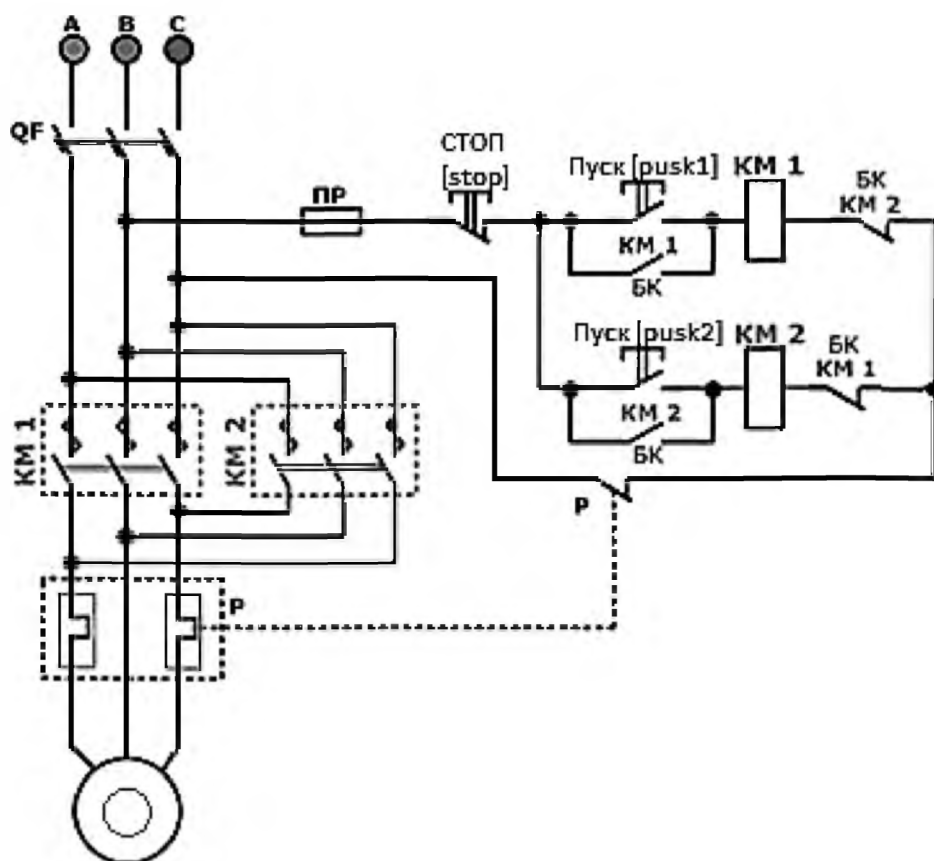


Рисунок 18 – Схема реверсивного пуска двигателя

Нам понадобится три входа, которые будут эмитировать «СТОП», «ПУСК1» и «ПУСК2», и два выхода, эмитирующие вращения двигателя в одну и в другую стороны. Для их создания перейдем в пункт «Ресурсы – Конфигурация ПЛК»

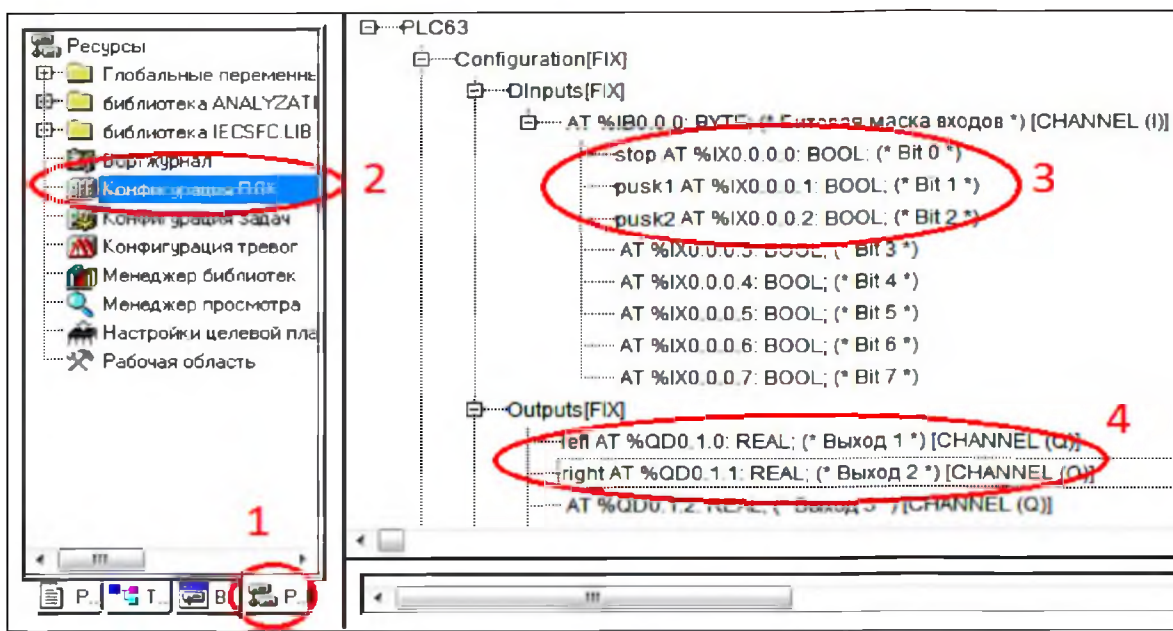


Рисунок 19 – Окно настройки конфигураций ПЛК

После выполненных настроек переходим обратно на вкладку «*POU*» и выбираем из списка нашу программу. Далее с помощью блоков создаем схему (рис. 20).

Одним из основных элементов данной программы является блок – RS триггер. Принцип его работы заключается в том, что, если положительный сигнал подаётся на вход SET, то на выходе мы получаем «1», если на RESET, то на выходе «0». Если на оба входа, то приоритет идет на вход RESET и на выходе мы также получаем «0» (в SR триггере приоритет отдаётся входу SET).

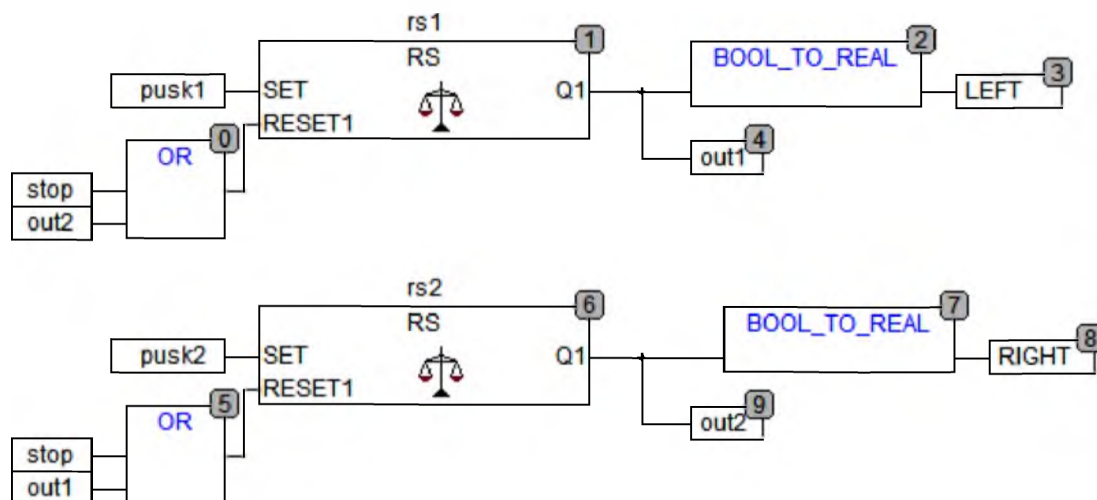


Рисунок 20 – Программа реверсивного пуска двигателя

Также для защиты от двойного включения мы ввели две переменные «out1» и «out2». Принцип ее работы прост: если двигатель запущен влево, то переменная «out1» равна «1» и на вход «RESET» триггера «rs2» подаётся единица, которая препятствует включению двигателя в обратную сторону. Тот же принцип защиты при включении двигателя для вращения вправо.

На вход «SET» триггеров мы подключили кнопки пуска двигателя, на входы «RESET» – кнопку «СТОП» и введенные переменные «out1», «out2». Для этого нам понадобится блок «OR», который выдает единицу при положительном сигнале хотя бы одного из входов.

Перед выходами «LEFT» и «RIGHT» необходимо поставить блок «BOOL_TO_REAL» для преобразования булевого сигнала в реальный (также существует обратный блок «REAL_TO_BOOL»)

После создания программы необходимо сделать ее визуализацию, для этого переходим в соответствующую вкладку (рис. 21).

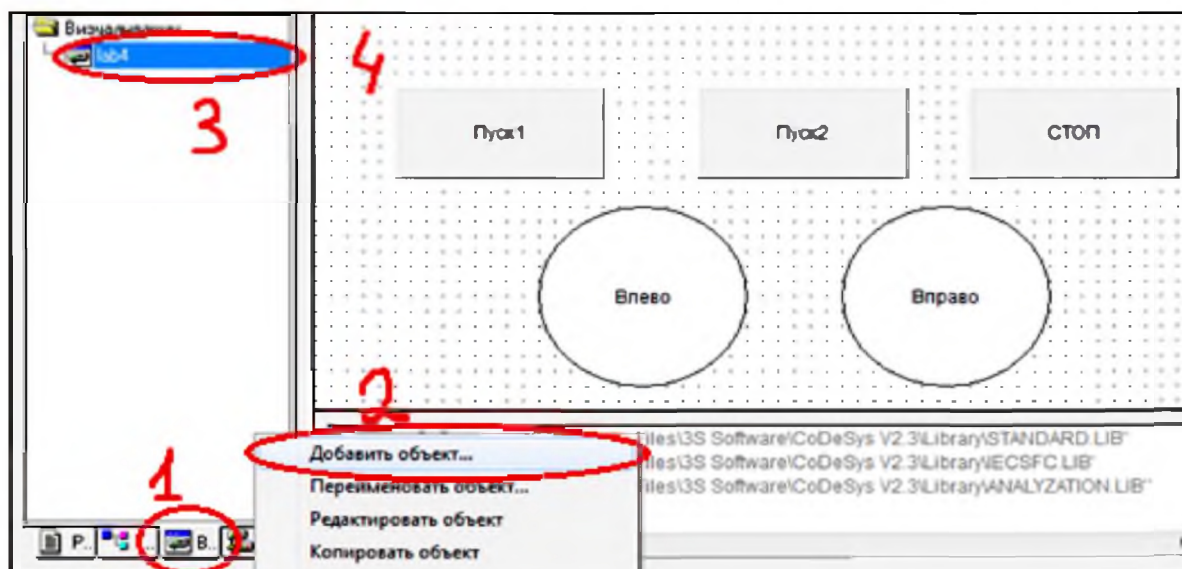


Рисунок 21 – Окно создания визуализации

В результате работы была создана программа реверсивного пуска двигателя с защитой от повторного запуска двигателя в другую сторону в программной среде CODESYS на языке CFC.

2.4 Модернизация токарно-винторезного станка

Модернизация – это обновление объекта, улучшение его качеств в соответствии с новыми требованиями и стандартами. Прогресс не стоит на месте и с каждым годом появляются все новые и новые открытия, все глубже изучаются свойства материалов. Для решения новых задач требуется новое оборудование или улучшенное, способное достичь поставленного результата. Для этой задачи и существует понятие модернизация. Если говорить о промышленности, под модернизацию попадают все виды машин и аппаратов, а также технологические процессы. Конечно же, основным видом усовершенствования производства является ее автоматизация, что приведет к экономии повышенной надежности, качеству и в целом развития предприятия.

Рассмотрим пример модернизации на токарно-винторезной модели станка. Для этого мы используем программное обеспечение CODESYS и программированный логический контроллер (ПЛК).

Для начала рассмотрим схему станка, его назначение и принцип работы. Электрооборудование станка служит для подключения силовых агрегатов, осветительных и сигнальных устройств к трёхфазной сети переменного тока и обеспечения их защиты от других вредных факторов. На рисунке 22 показана принципиальная электрическая схема станка.

Силовая цепь состоит из трех асинхронных трехфазных электродвигателей, предохранительных устройств и выключателей. В цепи управления находятся релейно-контакторные и другие аппараты и кнопки пуска, остановки и управления электродвигателем ускоренных перемещений.

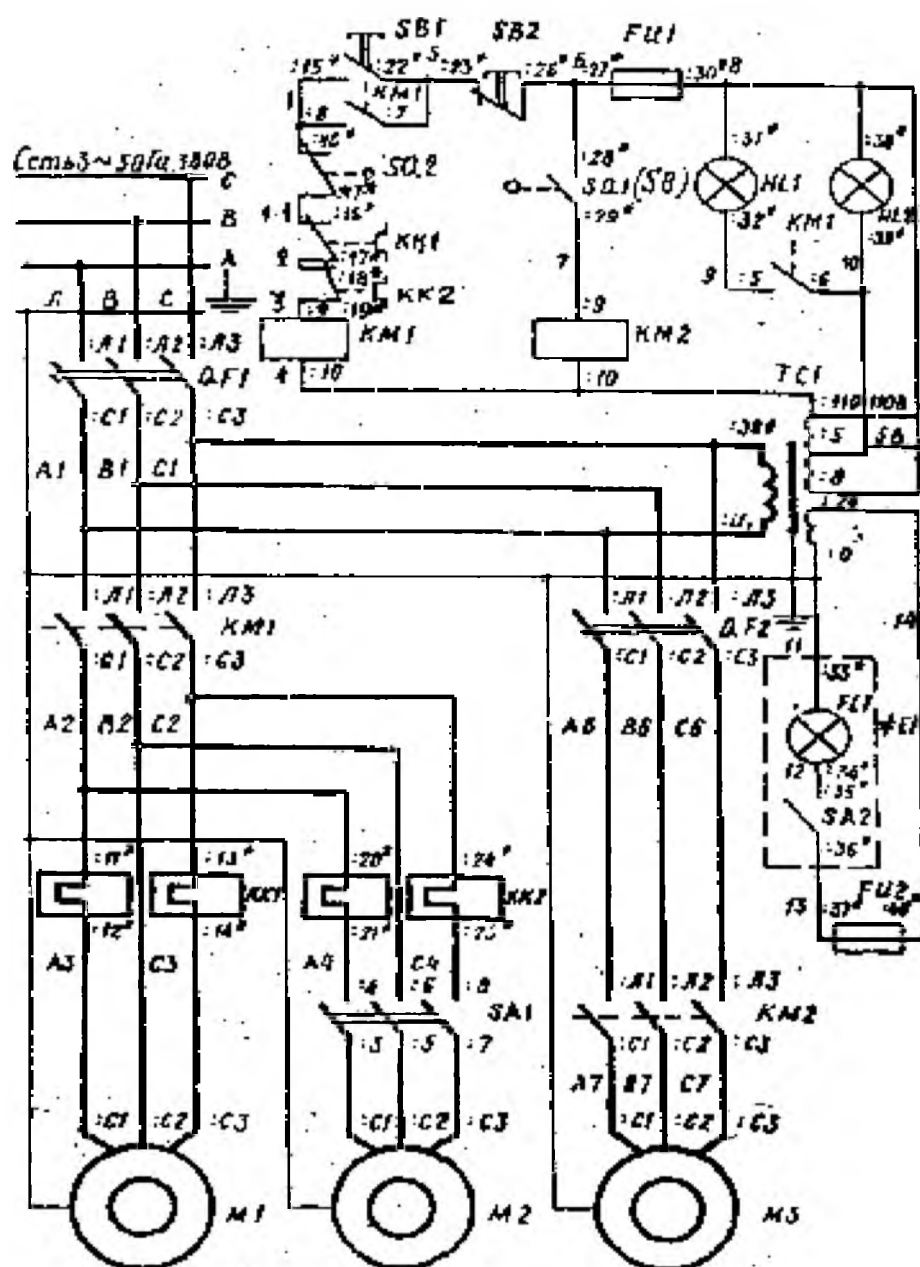
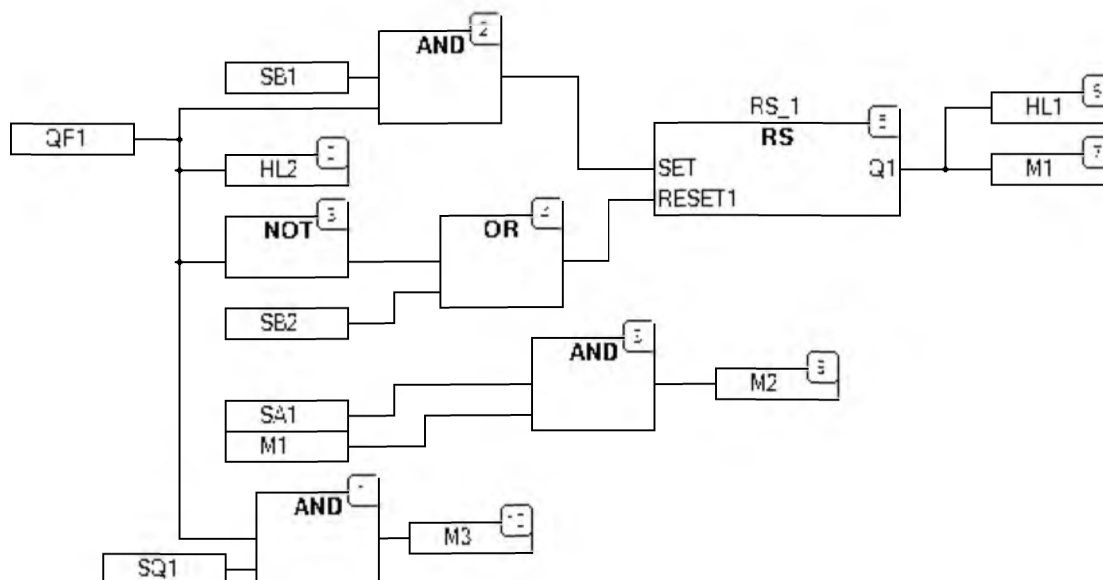


Рисунок 22 – Схема электрическая принципиальная



**Рисунок 23 – Реализация схемы управления
токарно-винторезного станка в CODESYS**

Для реализации программы в CODESYS необходимо понять принцип работы схемы. Подача питания на станок происходит с помощью вводного выключателя QF1. При наличии напряжения загорается лампа HL2. Пуск электродвигателя главного привода M1 происходит с помощью кнопки с SB1, которая замыкает цепь катушки магнитного пускателя. При отпускании кнопки, благодаря вспомогательным контактам катушки цепь не разомкнется, то есть при включении происходит самоподхват. При включении электродвигателя M1 происходит загорание лампы HL1. Остановка главного электропривода осуществляется посредством нажатия на кнопку SB2. Пуск электродвигателя M3 происходит нажатием кнопки, воздействующей на конечный выключатель SQ1. Нормально разомкнутый контакт замыкает цепь питания катушки электромагнита пускателя, который замыкает контакты силовой цепи электродвигателя быстрых перемещений. При отпускании кнопки SQ1 цепь разомкнется, катушка пускателя обесточится и электродвигатель M3 выключится. Пуск и остановка электронасоса M2 осуществляется с помощью переключателя SA1. Электронасос M2 включается только в том случае, если включен электродвигатель главного привода M1.

В написанной программе в CODESYS на языке CFC были использованы основные логические блоки и RS триггер. Для создания программы нам потребуются блоки ввода и вывода, основные функциональные логические блоки «AND», «OR» и «NOT», а также «RS» триггер. Для вводного выключателя используем блок ввода и назовём его «QF1» как показано на рисунке 23. Лампу, показывающей наличие сети, обозначим с помощью блока вывода и назовем его «HL2» и подключим напрямую к блоку «QF1». По условию двигатель M1 включается и отключается при помощи кнопочной станции. Для этого создадим еще два блока ввода и назовем их «SB1» и «SB2», которые будут включать, и отключать двигатель соответственно. Затем установив RS-триггер, подключим к его выходу два блока вывода: «M1» – двигатель и «HL1» – лампа, загорающая при работающем двигателе M1. Ко входу «SET» RS-триггера подводим два блока ввода: «QF1» и «SB1», через логический блок «AND». То есть двигатель M1 включится при включенном вводном выключателе и нажатой кнопкой SB1. На вход «RESET» подключаем также блок «QF1», но через блоки «NOT» и «OR», последовательно, для того, чтобы при отключенном вводном выключателе на вход «RESET» триггера поступала единица, и двигатель отключался, и блок «SB1», также через блок «OR». То есть при отключении вводного выключателя или при нажатии на кнопку SB2 двигатель M1 отключиться, а с ним и лампа HL1.

Двигатель M2 обозначим через блок вывода, для его включения и отключения установим выключатель «SA1» через блок ввода, также по условию работа двигателя M2 возможна только при работе двигателя M1, поэтому создадим еще один блок ввода «M1», который принимает те же значения, что и блок вывода «M1» и подключим его с блоком «SA1» через логический блок «AND». С этим условием двигатель M2 будет отключаться если работа двигателя M1 прекратиться, даже если выключатель SA1 будет включен.

Двигатель M3 работает при нажатии на толчковую кнопку SQ1, для нее создадим блок ввода «SQ1», и подключим его с вводным выключателем че-

рез блок «AND». Таким образом, двигатель МЗ будет работать при нажатии на толчковую кнопку и включенном выключателе QF1.

Работа программы аналогична условиям работы схемы станка, а также для удобного восприятия все переменные названы так же, как и на схеме. Таким образом, мы реализовали схему управления токарно-винторезного станка в CODESYS.

2.5 Вариации решений задач по автоматизации производственных процессов в программном комплексе CODESYS

Ранее мы рассмотрели такое программное обеспечение как CODESYS, познакомились с его интерфейсом, изучили основные элементы и создали простую программу для практического закрепления навыков. Для программирования в CODESYS, возможно использование нескольких языков, но ранее мы рассматривали только лишь язык функциональных блоков – “CFC”. Одно из преимуществ данного языка является то, что логика работы программы легко понимаема, и весь цикл работы виден перед глазами. Имея необходимые функциональные блоки, достаточно правильно соединить их входы и выходы для того, чтобы получить необходимую программу управления тем или иным технологическим процессом. Но одну и ту же программу возможно написать на разных языках, или используя несколько языков, это может помочь при решении сложных задач. Поэтому необходимо знать, что есть не один путь для их решения, все будет зависеть от поставленных требований. Сейчас мы рассмотрим некоторые вопросы по автоматизации процессов и их различные способы реализации на языке “CFC” и “ST”.

Первой задачей рассмотрим программу управления реверсивным пуском двигателя. Один из вариантов ее исполнения (рис. 24) был представлен и пояснен ранее.

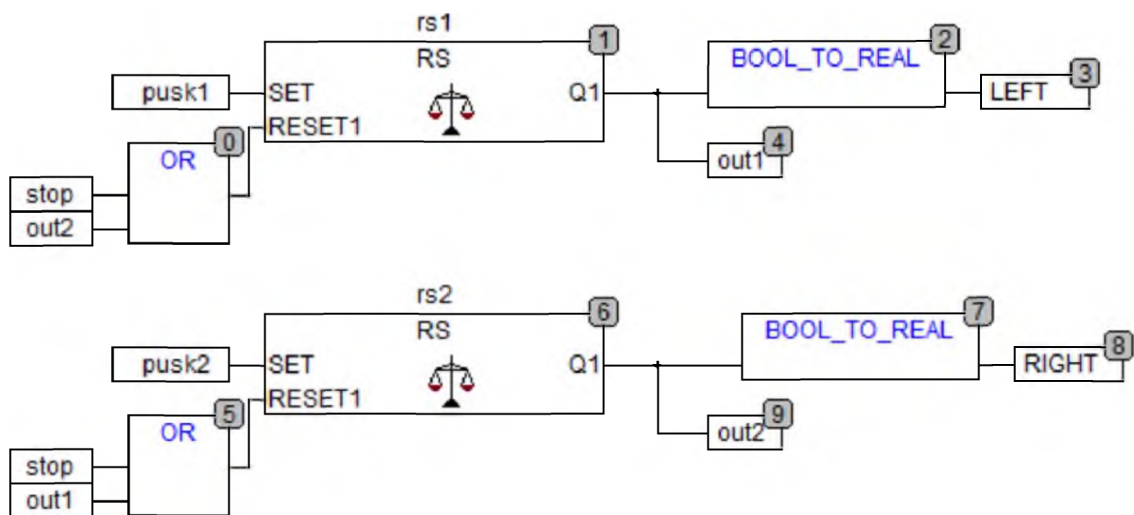


Рисунок 24 - Программа управлений реверсивным пуском двигателя на триггерах

В другом варианте реализации данной программы (рис. 25) отсутствуют триггеры, но вместо них мы использовали инверсию, которая отвечает за логическую операцию “НЕ”. Здесь также присутствуют три переменные входа, отвечающие за пуск в одну и другую стороны двигателя, а также за его остановку, и две переменные выхода, обозначающие вращение двигателя влево и вправо. Принцип работы программы заключается в следующем: на вход блока “AND” подаем три команды, первая – отвечает за пуск двигателя в том или ином направлении, вторая – команда остановки двигателя и третья – переменная, означающая пуск двигателя в противоположном направлении. Важно заметить, что последние две команды подаются через инверсию, то есть если она передает логическую единицу, то на вход будет поступать логический ноль. Таким образом, при присвоении только лишь первой команде логической единицы на вход блока “AND” поступает все три, и двигатель начинает вращаться. Третья команда предназначена для защиты от двойного включения. То есть, если двигатель уже вращается в одном направлении, то на противоположный блок “AND” подастся ноль и будет невозможным запустить двигатель в другом направлении.

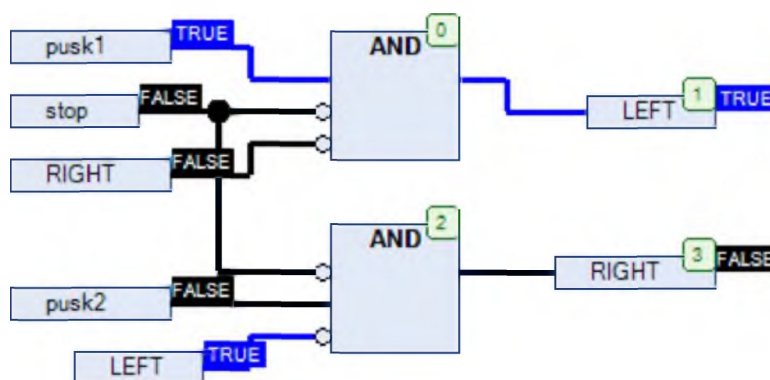


Рисунок 25 - Программа управления реверсивным пуском двигателя на инверсии

Таким образом, мы создали одну программу, но разными способами, что показывает гибкость языка “CFC” и различные возможные его инструменты.

Теперь рассмотрим решение одной задачи сначала на языке “CFC”, а затем на языке “ST”. На рисунке 26 показана схема управления двигателями многосекционного конвейера, перемещающего насыпные грузы из правого бункера в левый. Электропривод конвейеров осуществляется от асинхронных короткозамкнутых двигателей М1, М2, М3. Во избежание завалов грузов пуск конвейеров начинается с включения двигателя М1 нажатием кнопки КнП1. Контакт К1, срабатывая, своими главными контактами подает питание на двигатель М1. Затем последовательно, нажатием кнопок КнП1 и КнП3, включающих контакторы К2 и К3, пускаются двигатели М2 и М3. Пуск двигателя М2 раньше пуска М1 невозможен, так как в цепи катушки контактора К2 находится замыкающий блок-контакт контактора К1. Аналогично заблокированы двигатели М2 и М3. Общая остановка всех конвейеров производится нажатием кнопки КнС1.

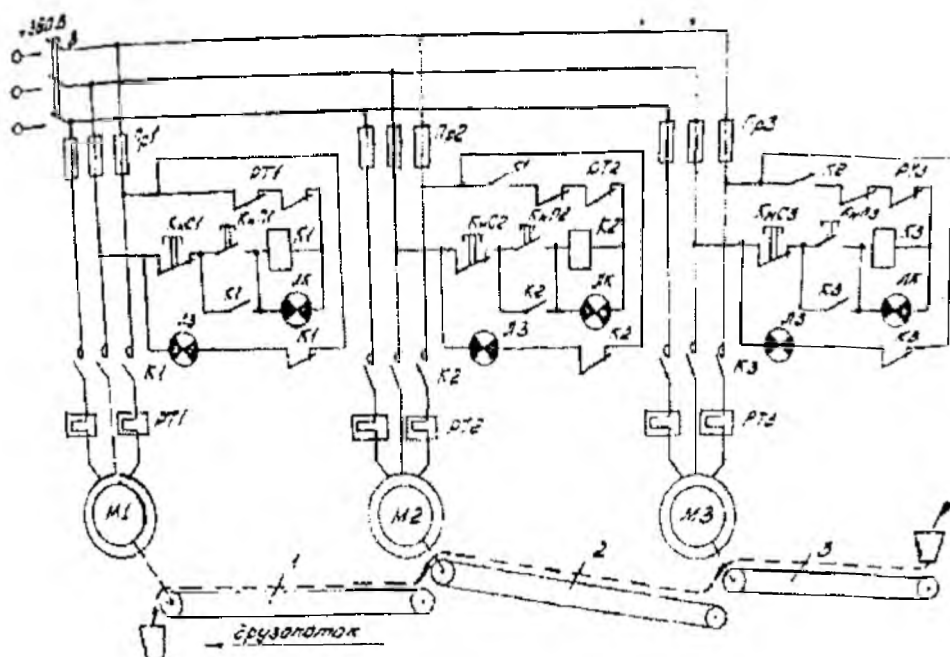


Рисунок 26 – Схема управления многосекционным конвейером

На основе данной схемы сформулируем задачу. Нам необходимо реализовать программу управления пуском и остановкой ленточного конвейера, состоящего из трех ступеней, в двух режимах: автоматическом и ручном. В первом необходимом условии является то, что три ленты конвейера должны включаться друг за другом с небольшой задержкой. В ручном режиме включение ступеней конвейера не зависят друг от друга и могут управляться отдельно. Также одним из условий работы является то, что при включенном ручном режиме невозможно запустить автоматический, но не наоборот.

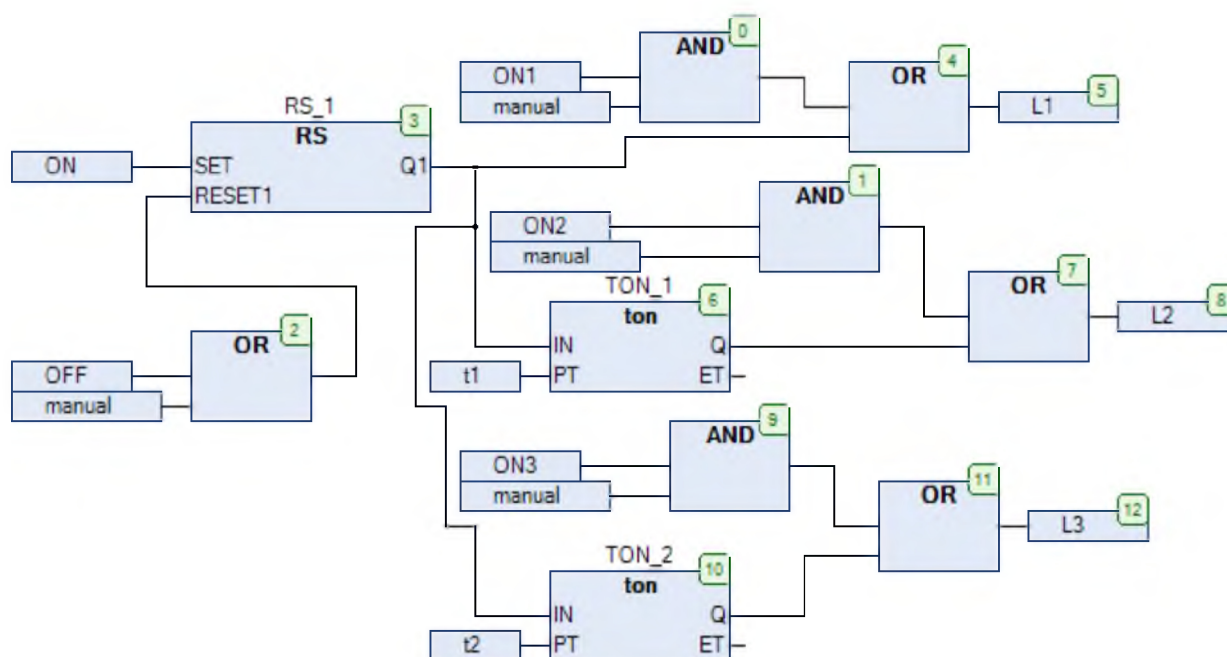


Рисунок 27 – Программа управление ленточным конвейером на языке “CFC”

В данной программе нам необходимо реализовать два способа управления конвейером – это ручной и автоматический. В автоматическом режиме три линии конвейера включаются друг за другом, начиная с нижней, с небольшой задержкой. В ручном режиме каждую линию конвейера можно включить независимо от другой, например для каких-либо проверок. Три выходных блока “L1”, “L2” и “L3”, отображают работу линий конвейера. Для запуска автоматического режима работы служит входной блок “ON”. После его включения на вход “SET” RS-триггера поступает единица, посредством чего включается первая линия “L1”. Затем, после определенной задержки, установленной с помощью таймера “TON”, включаются остальные линии. Для отключения работы конвейера служит входной блок “OFF”. Он передает логическую единицу на вход “RESET” RS-триггера, и работа прекращается. Входной блок “manual” служит для включения ручного режима управления. Запуск и остановка линий конвейера производиться посредством входных блоков “ON1”, “ON2” и “ON3”. Важно заметить, что при включении ручного режима будет невозможно запустить автоматический, так как на вход “RESET” RS-триггера будет подана логическая единица.

Теперь рассмотрим данную программу, только реализованную на языке “ST” (Structured Text). Это язык высокого уровня, для сравнения похож на Паскаль. Для работы в нем необходимо знать, как записываются основные команды: AND, OR, NOT, IF, THEN, ELSE, WHILE и т. д. Рассмотрим код программы представленный на рисунке 28.

0001	PROGRAM PLC_PRG	0001	(*Ручной режим*)
0002	VAR	0002	IF manual=TRUE AND ON1=TRUE THEN
0003	L1: BOOL;	0003	L1:=TRUE; ELSE L1:=FALSE;
0004	L2: BOOL;	0004	END_IF
0005	L3: BOOL;	0005	IF manual=TRUE AND ON2=TRUE THEN
0006	ON: BOOL;	0006	L2:=TRUE; ELSE L2:=FALSE;
0007	OFF: BOOL;	0007	END_IF
0008	manual: BOOL;	0008	IF manual=TRUE AND ON3=TRUE THEN
0009	ON1: BOOL;	0009	L3:=TRUE; ELSE L3:=FALSE;
0010	ON2: BOOL;	0010	END_IF
0011	ON3: BOOL;	0011	(*Автоматический режим*)
0012	OFF1: BOOL;	0012	IF manual=TRUE THEN
0013	IN: BOOL;	0013	ON:=FALSE;
0014	PT: TIME;	0014	END_IF
0015	TONInst2: TON;	0015	IF ON=TRUE AND manual=FALSE THEN
0016	TONInst1: TON;	0016	L1:=TRUE;
0017	END_VAR	0017	END_IF
0018		0018	IF manual=FALSE THEN
0019		0019	TONInst1(IN := L1, PT:= T#3s);
0020		0020	L2:=TONInst1.Q;
0021		0021	TONInst2(IN := L2, PT:= T#3s);
0022		0022	L3:=TONInst2.Q;
0023		0023	END_IF

а

б

Рисунок 28 - Программа управление ленточным конвейером на языке “ST”.

а) объявление переменных б) листинг программы

В начале прописан код управления в ручном режиме. Рассмотрим значения команд:

- “IF” – если;
- “THEN” – тогда;
- “ELSE” – иначе;
- “END_IF” – конец команды если;
- “manual” – ручной режим;
- “ON” – автоматический режим;
- “TRUE” - истина (логическая единица);

“FALSE” – ложь (логический ноль);

“AND” - и;

“ON1”, “ON2”, “ON3” - кнопка включения первой, второй и третьей линии конвейера соответственно;

“L1”, “L2”, “L3” – три линии конвейера.

Во второй, пятой и восьмой строках записаны условия включения каждой линии конвейера с первой по третью соответственно. В двенадцатой строке мы указываем условие отключения автоматического режима при включении ручного. В пятнадцатой строке записано условие включения автоматического режима, то есть первой линии конвейера. Для включения последующих линий с некоторой задержкой нам необходимо прописать команду, включающая таймер “TON”, она показана в девятнадцатой и двадцать первой строке. Команда “TONInst” – это название таймера, в скобках переменной “IN” присваиваем переменную после включения, которой у нас начнётся отсчет времени, в нашем случае это “L1”. Переменной “PT” присваиваем время задержки, как показано на рисунке. Затем, присваиваем той переменной, которой хотим дать значение “TRUE” после того, как пройдет заданное время, команду: L2:=TONInst1.Q.

Таким образом, мы создали программу управления включением ленточного конвейера на двух языках “CFC” и “ST”. В каждом есть свои плюсы и минусы, и каким путем пойти решать вам.

Мы рассмотрели решения нескольких задач разными способами, это говорит о том, что программный комплекс CODESYS очень гибок и имеет множество возможностей для автоматизации различных технологических процессов.

2.6 Автоматизация автономной системы подачи воды на животноводческий комплекс

А теперь разберем задачу без использования ПЛК, применим измеритель – регулятор ОВЕН ТРМ202. Сейчас активно развиваются и строятся животноводческие фермы и комплексы, которые находятся на отдаленном расстоянии от крупных населенных пунктов, что приводит к затратному подключению к системе водоснабжения. Для решения данной проблемы можно использовать грунтовые, межпластовые или артезианские воды, что намного эффективнее, дешевле в подключении и имеет самокупаемость.

Грунтовая вода располагается на первом от поверхности земли водонепроницаемом слое (рисунок 29), получая постоянную атмосферную подпитку в виде дождя или снега, имеет небольшой запас, и обычно не подходит для питья. Для добычи грунтовой воды обычно бурят скважины. Из-за особенности гидравлики для скважины глубиной 9 метров используют поверхностные насосы, а свыше 9 метров поверхностные насосы с внешним эжектором.

Артезианские воды чище и пригодны для питья, располагаются на втором водонепроницаемом слое. Артезианский водоносный горизонт (рисунок 39) вообще не подвержен прямому влиянию климата и времени года, поэтому вода в скважине и зимой и летом имеет одинаковый уровень. Глубина скважины от 20 – 100 метров и выше, требует установки более мощных скважных насосов.

Для бесперебойного водоснабжения необходимо автоматизировать процесс добычи воды. Для этой задачи используем датчик давления (преобразователь давления), измеритель регулятор, скважной насос, и емкость для хранения воды. Согласно нормативному документу СП 31.13330.2012 актуализированная редакция СНиП 2.04.02-84 /3/, нормативное давление для потребителей являются: от 0,3 до 6 атмосфер для холодной воды, от 0,3 до 4,5 атмосферы для горячей воды.

Для создания нужного давления, регулирования напора и расхода воды, создания ее запаса и выравнивания графика работы насоса, необходимо построить водонапорную башню. Для определения необходимой высоты водонапорной башни используем формулу гидростатического давления:

$$p = \rho \cdot g \cdot h, \quad (1)$$

где $\rho = 1000 \text{ кг/м}^3$ - плотность воды.

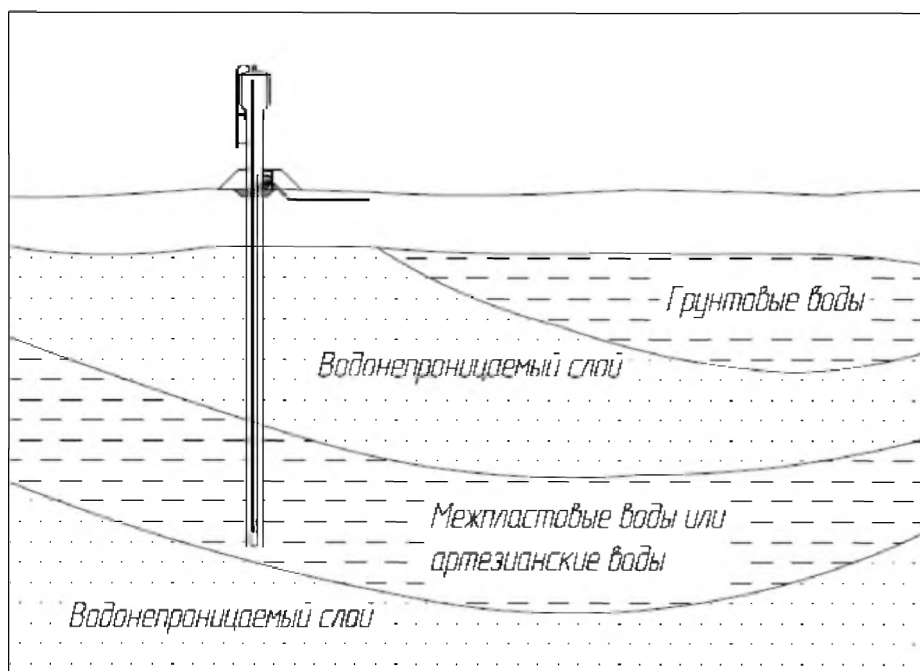


Рисунок 29 - Уровень грунтовых вод

Необходимое давление холодной воды для животноводческой фермы 0,3 -4 атм. Для расчета примем 1,2 атм = 121590 Па:

$$h = \frac{p}{\rho \cdot g} = \frac{121590 \text{ Па}}{1000 \text{ кг/м}^3 \cdot 9,8 \text{ м/с}^2} = 12,4 \text{ м}$$

Для определения объема водонапорного бака, нужно рассчитать максимальный часовой расход воды и обеспечить 2-3 часа бесперебойного водоснабжения в случае отключения электричества.

Предположим, что автоматизация проходит на животноводческой ферме крупного рогатого скота. На ферме 200 коров, 8 доярок, 4 скотника и 2 электрик, тогда среднесуточный расход воды:

$$Q_{cc} = \sum_{i=1}^n q_i \cdot m_i \quad (2)$$

где q_i - среднесуточный расход воды одним потребителем m_i

Норма потребления воды одной коровой 80л/сут, одного работника 25л/сут /4/:

$$Q_{cc} = 200 \cdot 80 \text{ л} + (8 + 4 + 2) \cdot 25 = 16350 \text{ л/сут}$$

Максимальный часовой расход воды:

$$Q_{чр} = \frac{Q_{cc} \cdot K_c \cdot K_{ч}}{24} \quad (3)$$

где: $K_c = 1,3$ коэффициент неравномерности среднесуточного потребления воды.

$K_{ч} = 2,3$ коэффициент часовой неравномерности расхода воды;

$$Q_{чр} = \frac{16350 \cdot 1,3 \cdot 2,3}{24} = 2037 \text{ л}$$

Итоговый объем водонапорного бака:

$$V = V_{п} + 2 \cdot Q_{чр} + 0,4 \cdot (V_{п} + 2 \cdot Q_{чр}) \quad (4)$$

где: $V_{п} = 21600 \text{ л}$ – противопожарный запас, из расчета тушения в 2 разных местах в течении 10 минут с общим расходом воды 10л/с

$2 \cdot Q_{чр}$ – объем воды из расчета 2-х часового бесперебойного водоснабжения в случае аварийного отключения электроэнергии

$0,4 \cdot (V_{п} + 2 \cdot Q_{чр})$ – регулируемый объем бака водонапорной башни.

$$V = 21600 + 2 \cdot 2037 + 0,4 \cdot (21600 + 2 \cdot 2037) = 36700 \text{ л}$$

Исходя из этого, выбираем водонапорную башню (Рожновского) БР-50У-15, объемом 50 м^3 , высотой 15 метров.

Насос выбирается в зависимости от глубины залегания артезианского водоносного горизонта с учетом максимального часового расхода воды $2 \text{ м}^3/\text{ч}$, например, если глубина скважины 100 м, необходимое давление напора на выходе трубы 2 атм, расстояние от насоса до дна скважины 5 м, высота башни 15 м, то напор насоса должен быть:

$$100 + 20 - 5 + 15 + (100 + 15) \cdot 0,1 = 141,5 \text{ м}$$

В таком случае выбираем насос UNIPUMP ECO 8, напор 150 м, производительность 3 м³/ч, мощность 2200 Вт.

После выбора водонапорной башни и насоса, необходимо, разработать схему автоматизированного управления насосом, для бесперебойного обеспечения водой животноводческой фермы. Главным элементом для поддержания постоянного уровня воды в водонапорной башне служит преобразователь давления Cerabar M PMC 45 с керамическим сенсором и гигиеническим фланцем для подключения к системе водоснабжения, рассчитанный на давление в 4 бар и измеритель-регулятор ОВЕН ТРМ202 для управление насосом (рисунок 30).

Основным элементом преобразователя давления служит керамический сенсор – сухой чувствительный элемент с прямым воздействием давления процесса на керамическую диафрагму, деформируемую максимально на 0,025 мм. Изменение электрической емкости, измеряемой между электродами на диафрагме и керамической основе, пропорционально изменению давления.

Преобразователь Cerabar M PMC45 (на схеме элемент AA2) подключается к источнику постоянного напряжения 11,5 – 45В (на схеме элемент G1) и при изменении давления на керамический сенсор выдает сигнал в виде постоянного тока в диапазоне 4 – 20мА на измеритель-регулятор ТРМ 202 (на схеме элемент AA1), который в свою очередь управляет пускатель КМ1. Пускатель КМ1 рассчитан на питание от того же источника питания G1 и управляет контактной группой КМ1.1, включая/отключая насос М1.

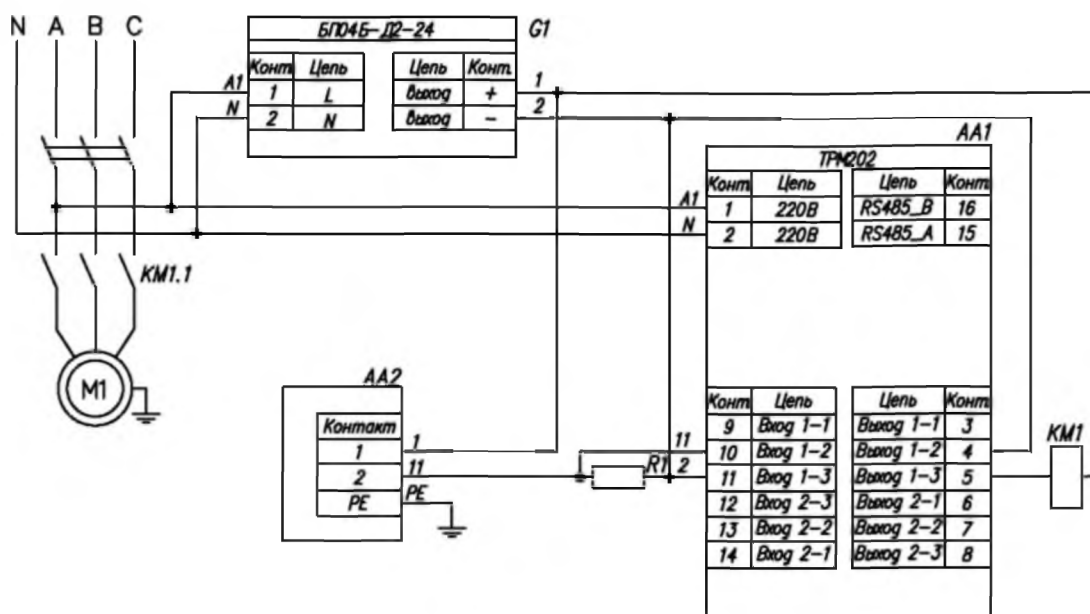


Рисунок 30 Схема подключения комплектующих к измерителю-регулятору

Для правильной работы преобразователя давления его необходимо откалибровать, используя калибраторы давления, к примеру, портативный калибратор давления Метран-517 или эталонный модуль давления Метран-518, после калибровки, нужно настроить работу измерителя – регулятора, пример настройки представлен в таблице 3.

Таблица 3. Настройка измерителя – регулятора ОВЕН ТРМ202.

Настройка подключенного преобразователя давления. Группа LuLn .			
Обозначение	Название	Значение	Комментарии
Ln.t1 / (in.t1)	Тип входного датчика	4.20	Сигнал тока от 4 до 20 мА
dPt1 / (dPt1)	Точность вывода	0	Задаёт число знаков после запятой при отображении на индикаторе
dP1 / (dP1)	Положение десятичной точки	0	Задаёт число знаков после запятой при отображении измеряемой величины
Ln.L1 / (in.L1)	Нижняя граница диапазона измерения	0	Нижняя граница Serabar 0 бар, т.е. 0 кПа

<u>SH1</u> (In.H1)	Верхняя граница диапазона измерения	400	Верхняя граница Cerabar 4 бар, т.е. 400 кПа
<u>SH1</u> (SH1)	Сдвиг характеристики датчика	!	Настраивается при подключенном калибраторе давления. При 0 бар на датчике, на индикаторе ТРМ202 должно быть 0кПа.
Настройка регулирования и регистрации. Группа <u>LVAL</u>			
<u>SL.L1</u> (SL.L1)	Нижняя граница задания уставки	0	Ограничена максимальной высотой столба воды башни:
<u>SL.H1</u> (SL.H1)	Верхняя граница задания уставки	180	h1=23,57м (полная высота башни). h2=4,5м (глубина заложения основания башни) Тогда по формуле (1): $p = \rho \cdot g \cdot (23,57 - 4,5) = 186,9 \text{ кПа}$
Параметры ключевого выхода			
<u>COMP1</u> (CmP1)	Тип логики работы компаратора	1	Обратное управление («нагреватель»), поддержания давления в заданном значении
<u>HYS1</u> (HYS1)	Значение гистерезиса для компаратора	60	Максимальное и минимальное разрешенное отклонение
Параметры индикации. Группа <u>RDu</u>			
<u>disP</u> (diSP)	Режим индикации текущих измерений	<u>stat</u>	На индикаторе постоянно отображается входная величина ЛУ1
<u>rEST</u> (rEST)	Время выхода из программирования	5-10	Время, по истечении которого прибор возвращается к индикации

Принцип работы основывается на типе логики работы компаратора обратное управление («нагреватель»), допустим, задано давление в 1,2 бар/120кПа, гистерезис по умолчанию 60 кПа (можно изменять в соответствующем пункте настроек, см. Таблица 1). При пустой емкости водонапорной башни, измеритель-регулятор АА1 запустит насос М1 через пускатель

КМ1. Как только давление в емкости станет 180 кПа ($120+60$), насос отключится (рис.31).

При потреблении воды давление в емкости будет падать, и когда упадет до 60кПа ($120-60$), насос повторно включится. Учитывая объем емкости 50м^3 и среднесуточный расход воды в 17м^3 при нормальной работе фермы, насос будет запускаться раз в 2ое суток.

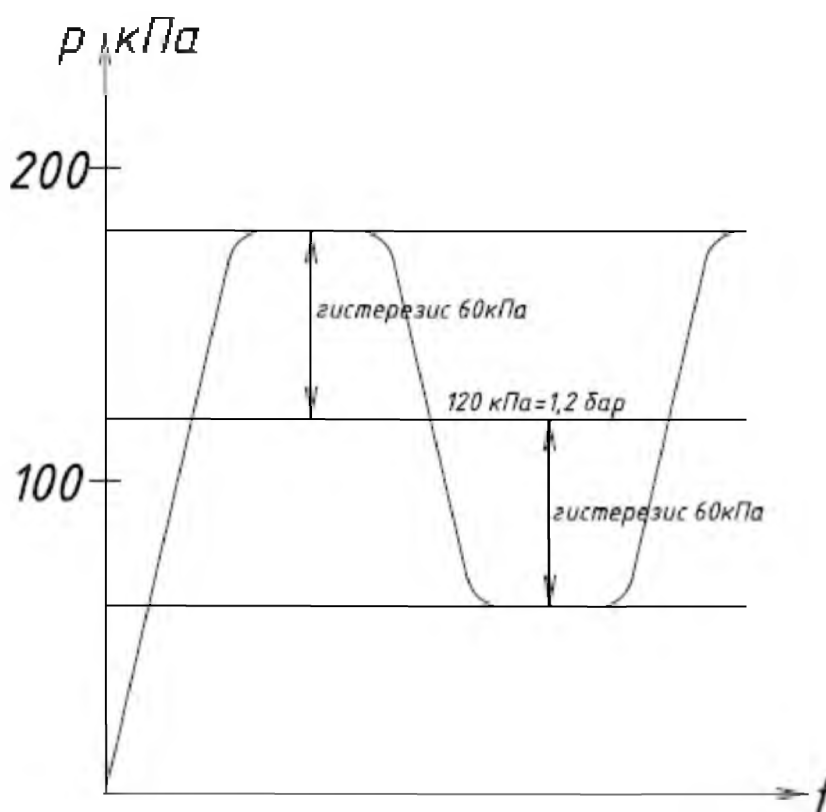


Рисунок 31. Примерный график работы автоматизированного водоснабжения

Как видим, задача была решена. Данная автоматизация водоснабжения животноводческой фермы и комплекса, способна не только бесперебойно обеспечивать водой, но и значительно увеличить срок работы насоса.

2.7 Автономная система подачи воды с применением пид регулирования, реализованная на плк154

Нами был рассмотрен способ автоматизации водоснабжения животно-водческого комплекса от одного автономного источника – межпластовые или артезианские воды. Теперь реализуем его немного по другому. Водоносный слой не постоянен, в засушливый сезон или зимой, объема воды в слое может не хватить для водоснабжения животноводческого комплекса. Необходимо подключение к централизованному водоснабжению, для такого подключения необходимо согласовывать работу насосов и запорных клапанов.

Для согласования используем программируемый логический контроллер, настроенный на ПИД регулирование. ПИД – регулятором называется устройство, применяемое в системе управления, для формирования сигнала управления в автоматических системах, где необходимо достичь высоких требований качеству и точности переходных процессов.

ПИД – регулятор оснащен звеном обратной связи, который в совокупности с заданным значением регулятора образует сигнал рассогласования. Управляющий сигнал ПИД – регулятора получается в результате сложения трех составляющих: первая пропорциональна величине сигнала рассогласования, вторая – интегралу сигнала рассогласования, третья – его производной (рисунок 32).

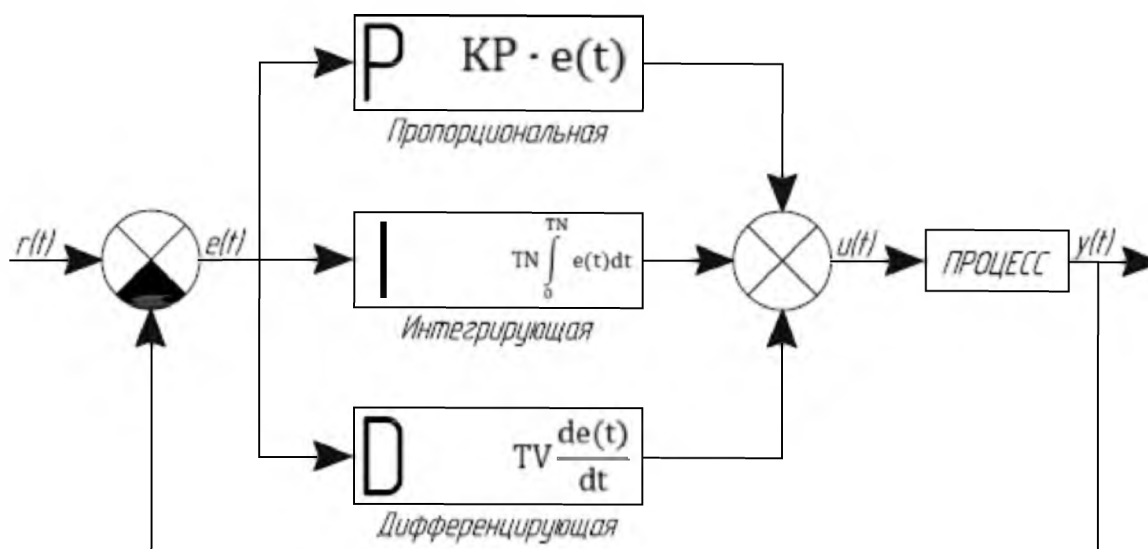


Рисунок 32 - Схема ПИД-регулятора

После выбора системы регулирования необходимо, разработать схему автоматизированного управления насосами и запирающими клапанами для бесперебойного обеспечения водой животноводческого комплекса (рисунок 33).

Для звена обратной связи используем преобразователь давления Cerabar M PMC 45 (на схеме элемент AA3) с керамическим сенсором и гигиеническим фланцем для подключения к системе водоснабжения, рассчитанный на давление в 4 бар. Для измерения давления центрального водоснабжения и давление насосной станции используем датчик давления EWPA007 4..20мА (на схеме AA1 и AA2, соответственно), рассчитанный на давление до 8 бар. Для питания используем источник постоянного напряжения 11,5 – 45В (на схеме элемент G1).

Все датчики подключены к ПЛК154 (на схеме AA4), который в свою очередь управляет пускателями KM1 – KM4, рассчитанные на напряжение 220В. Пускатель KM1 включает насос закачки воды из водоносного слоя (на схеме элемент M1), KM2 запускает насос подкачки из центрального водоснабжения. KM3 и KM4 включает запорные клапаны. Выбираем запорные чугунные соленоидный клапаны SMART HF6752, диаметром 80мм, управляющее напряжение AC 220В.

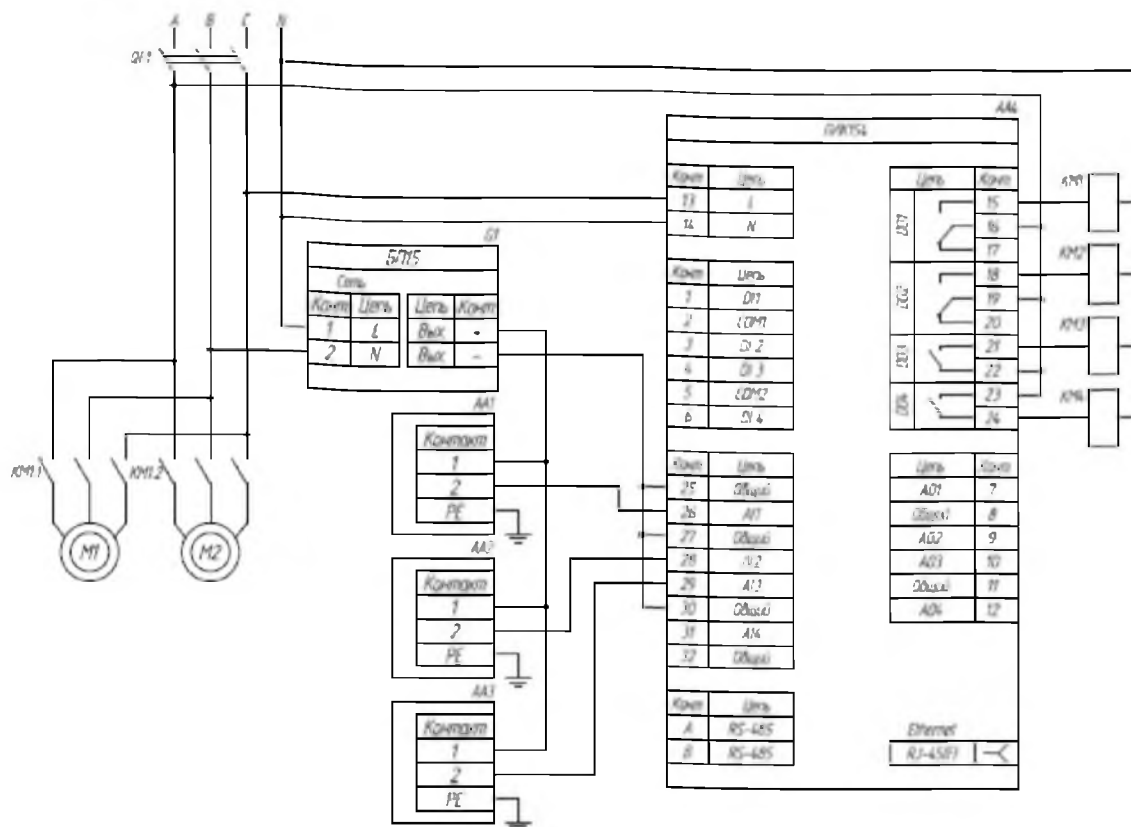


Рисунок 33 - Схема подключения комплектующих к ПЛК154 Овен

Программирование ПЛК осуществляется с помощью программы CODESYS на языке визуального программирования CFC. Для управления запорными клапанами используем блоки сравнения, которые по выполнению условия выдают сигнал TRUE – логическая 1, в противном случае сигнал FALSE – логический 0. Схема представлена на рисунке 34.

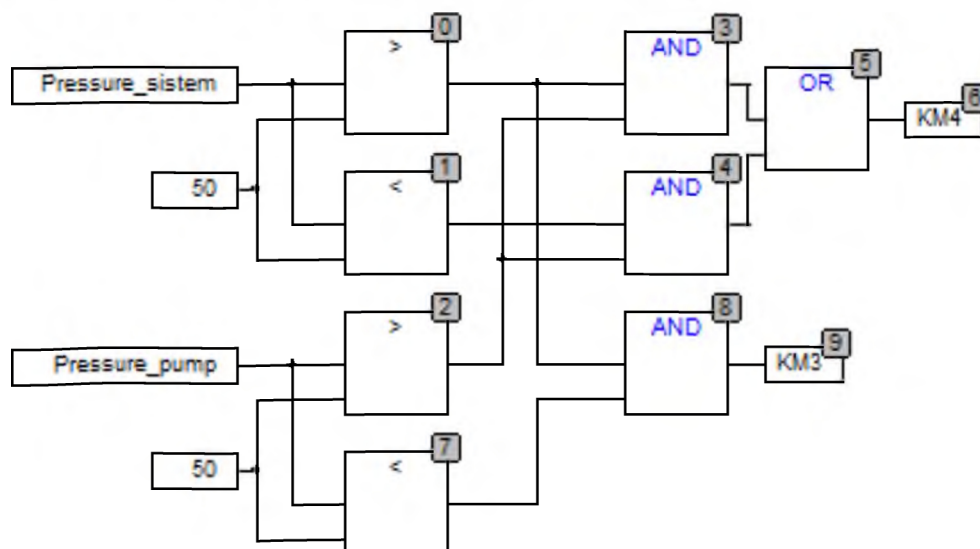


Рисунок 34 - Схема управления запорными клапанами

Вводные аналоговые сигналы Pressure_sistem и Pressure_pump от датчиков давления AA1 и AA2. Для преобразования сигнала 4..20мА в давление 0...120 кПа в ПЛК встроен аналого-цифровой преобразователь – устройство, преобразующее входной аналоговый сигнал в дискретный код (цифровой сигнал).

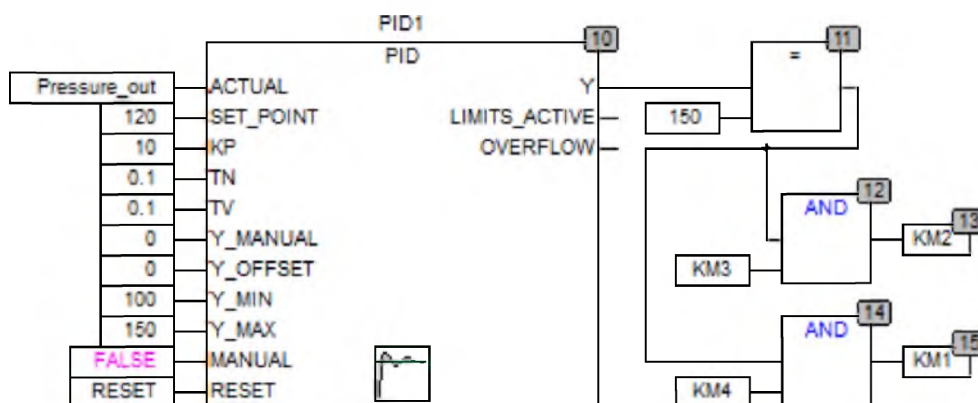


Рисунок 35 - ПИД-регулятор в программе CODESYS

Параметры ПИД-регулятора CODESYS следующие:

- ACTUAL – это текущее значение измеряемого параметра, сигнал обратной связи ПИД – регулятора.
- SET_POINT – задание регулятору (установленное давление).
- KP –коэффициент усиления пропорциональной части регулятора.
- TN –время постоянного интегрирования (в секундах).
- TV – время дифференцирования (в секундах).
- Y – выход регулятора.
- Y_MIN, Y_MAX – минимальное и максимальное значение выхода регулятора.
- RESET – сбрасывает выход ПИД-регулятора.

Функциональный блок реализует ПИД закон регулирования:

$$Y = Y_OFFSET + KP \left(e(t) + \frac{1}{TN} \int_0^{TN} e(t) + TV \frac{de(t)}{dt} \right)$$

Для подбора коэффициентов ПИД регулирования используем метод Зиглера – Никольса, суть метода состоит в следующем:

1. Выставляем все коэффициенты (K_P , T_N , T_V) в 0.
2. Начинаем постепенно увеличивать значение K_P и следим за реакцией системы. Нам нужно добиться, чтобы в системе начались устойчивые колебания (вызванные перерегулированием). Увеличиваем K_P , пока колебания системы не стабилизируются (перестанут затухать).
3. Остальные коэффициенты подбираются аналогично.

Принципиальная схема автономной системы подачи воды представлена на рисунке 36. ПЛК154 постоянно контролирует значение давления центрального водоснабжения и водоносного слоя с помощью датчиков давления АА1 и АА2.

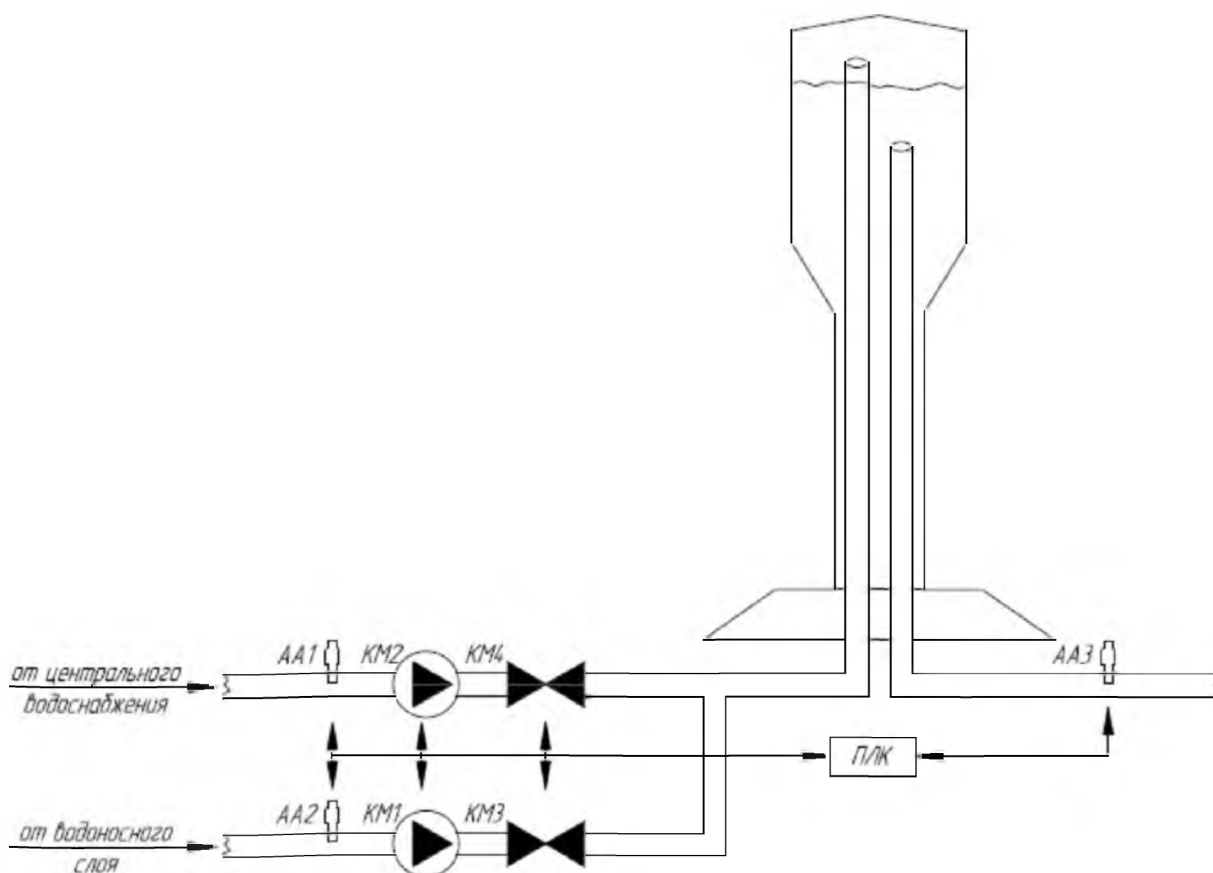


Рисунок 36 - Принципиальная схема автономной подачи воды на животноводческий комплекс

Если давление водоносного слоя меньше 0,5 бар, а центрального водоснабжения выше чем 0,5 бар, то предпочтение отдается центральному водоснабжению, закрывается запорный клапан КМ3 и по команде ПЛК включается насос М2 (с помощью контактной группы КМ2.1). В случае, когда давление водоносного слоя выше 0,5 бар, закрывается запорный клапан КМ4 и разрешается запуск М1 (с помощью контактной группы КМ1.1).

Таблица 4. Значение ПИД – регулятора при различных значениях давления.

Текущее значение	Задание регулятору	Выход регулятора	Состояние М1 или М2
0 – 100	120	150	Запуск
110	120	150	Запуск
120	120	150	Запуск
125	120	100	Остановка
130	120	100	Остановка
140	120	100	Остановка
150	120	100	Остановка

Таблица 4 показывает реакцию ПИД – регулятора при различных значениях преобразователя давления (текущее значение) и состояние насоса М1 или М2.

Данная автоматизация с применением ПИД – регулирования позволяет достаточно точно поддерживать давление воды на заданном уровне.

Вопросы для самопроверки:

1. Какие языки программирования используются в CODESYS?
2. В чем отличие RS и SR триггеров?
3. Перечислите основные функциональные блоки в CODESYS.
4. Объясните назначение ПИД-регулятора.
5. Перечислите основные логические операции в CODESYS.
6. Перечислите параметры ПИД-регулятора в CODESYS.

РАЗДЕЛ 3. ИЗУЧЕНИЕ ARDUINO

3.1 Установка и работа с программным обеспечением

Arduino IDE основана на языке Processing, являющимся подязыком программирования C. Выбрать и скачать версию программы, соответствующую вашей операционной системе, можно на странице загрузки официального сайта проекта Arduino: <https://www.arduino.cc>.

После распаковки и установки (желательно не на диск C, чтобы при переустановке операционной системы не пострадали файлы с программой и проектами) требуется сделать еще пару вещей. Выбрать тип платы, с которой будем работать. В нашем случае это Arduino Uno.

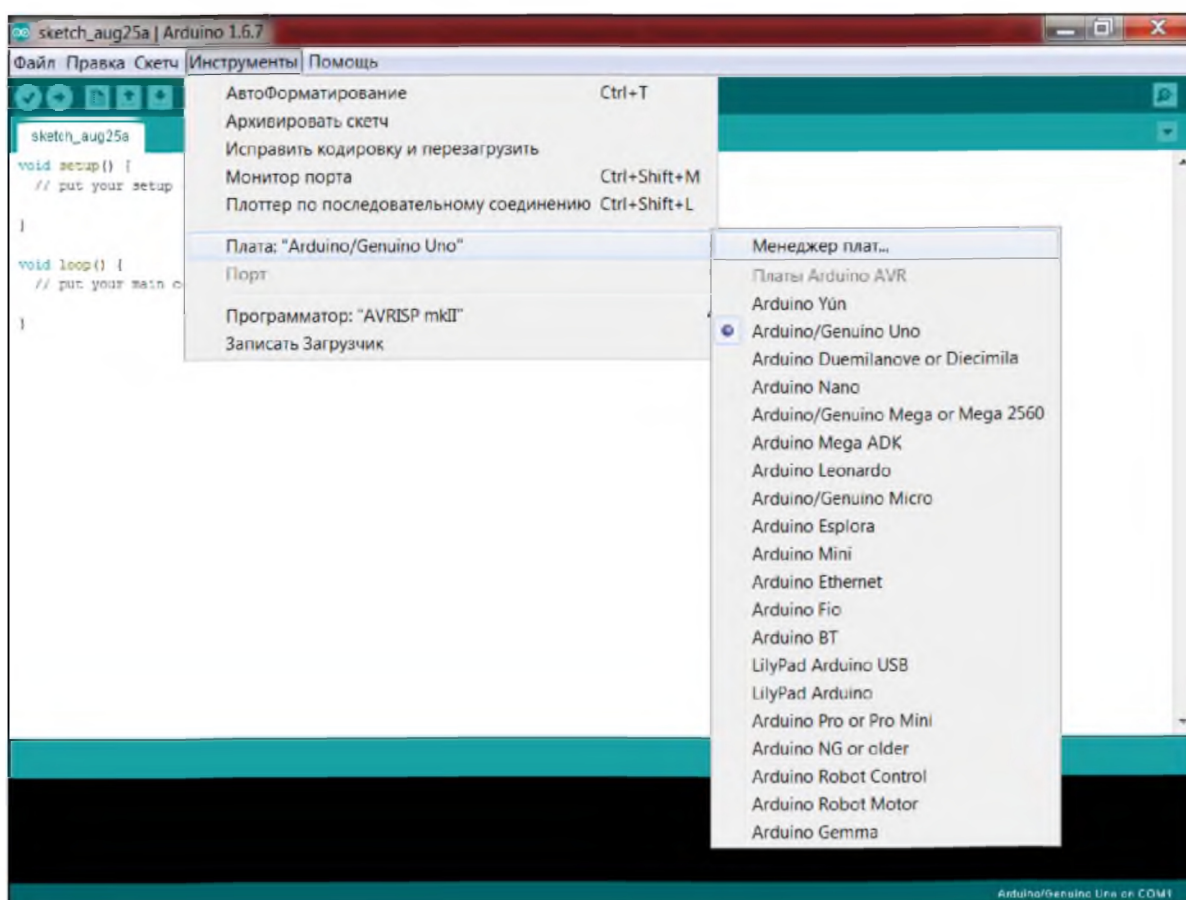


Рисунок 37 - Выбор платы

И требуется, уточнив предварительно в диспетчере устройств, указать в **Инструменты – Порт** используемый порт.

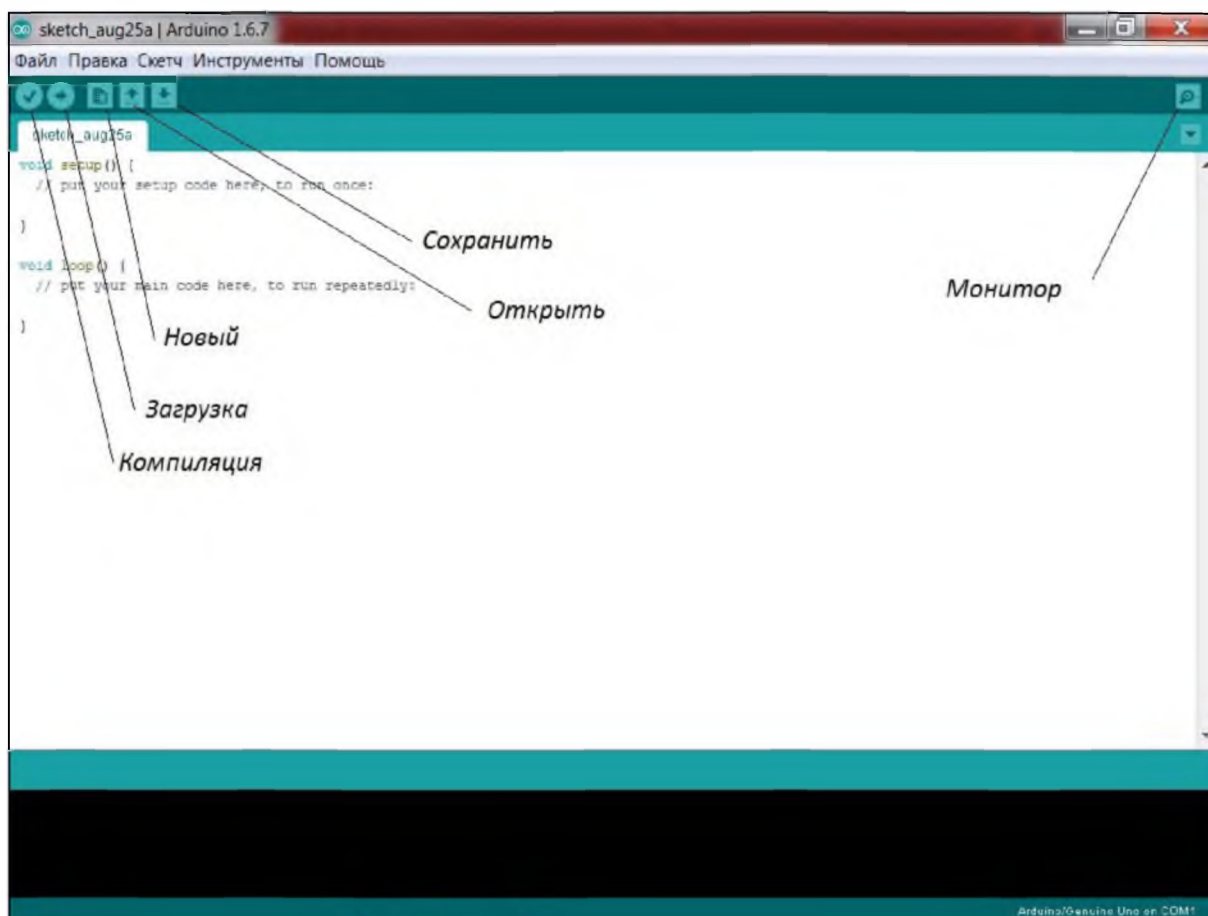


Рисунок 38 - Рабочее окно

Под главным меню находится панель инструментов:

- **Компиляция** – создание файла для передачи микроконтроллеру платы;
- **Загрузка** – передача нашей программы микроконтроллеру;
- **Новый** – создание нового файла;
- **Открыть** – открытие программы;
- **Сохранить** – сохранение программы;
- **Монитор** – открытие встроенного ASCII-терминала.

Теперь следует, нажав значок **Монитор**, задать скорость 9600 бод.



Рисунок 39 - Плата Arduino Uno

Применяемый микроконтроллер ATmega328p.



Рисунок 40 - Кабель для подключения платы Arduino к компьютеру

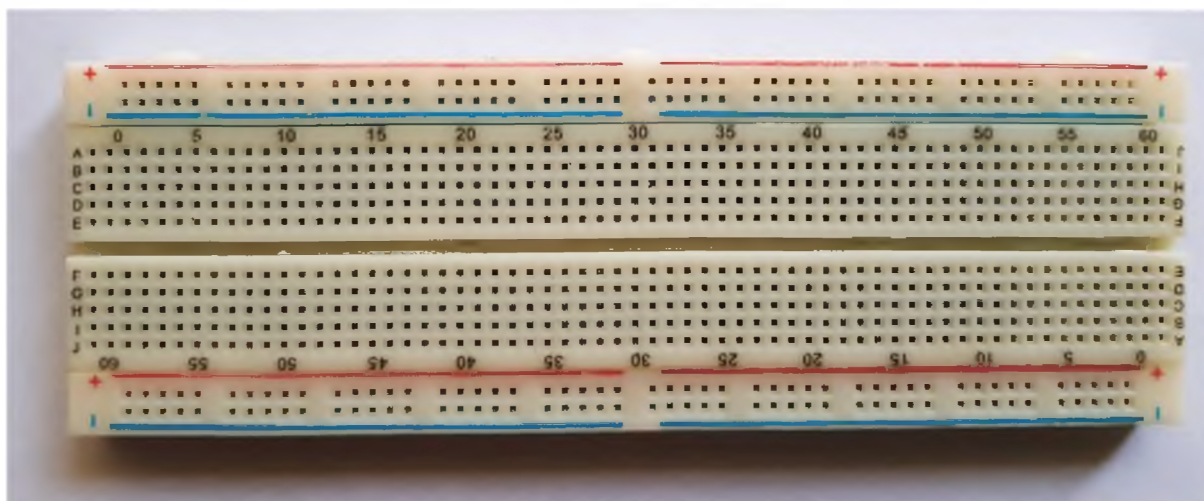


Рисунок 41 - Плата для наладки Breadboard

Следует учесть, что на плате разъемы уже соединены между собой.

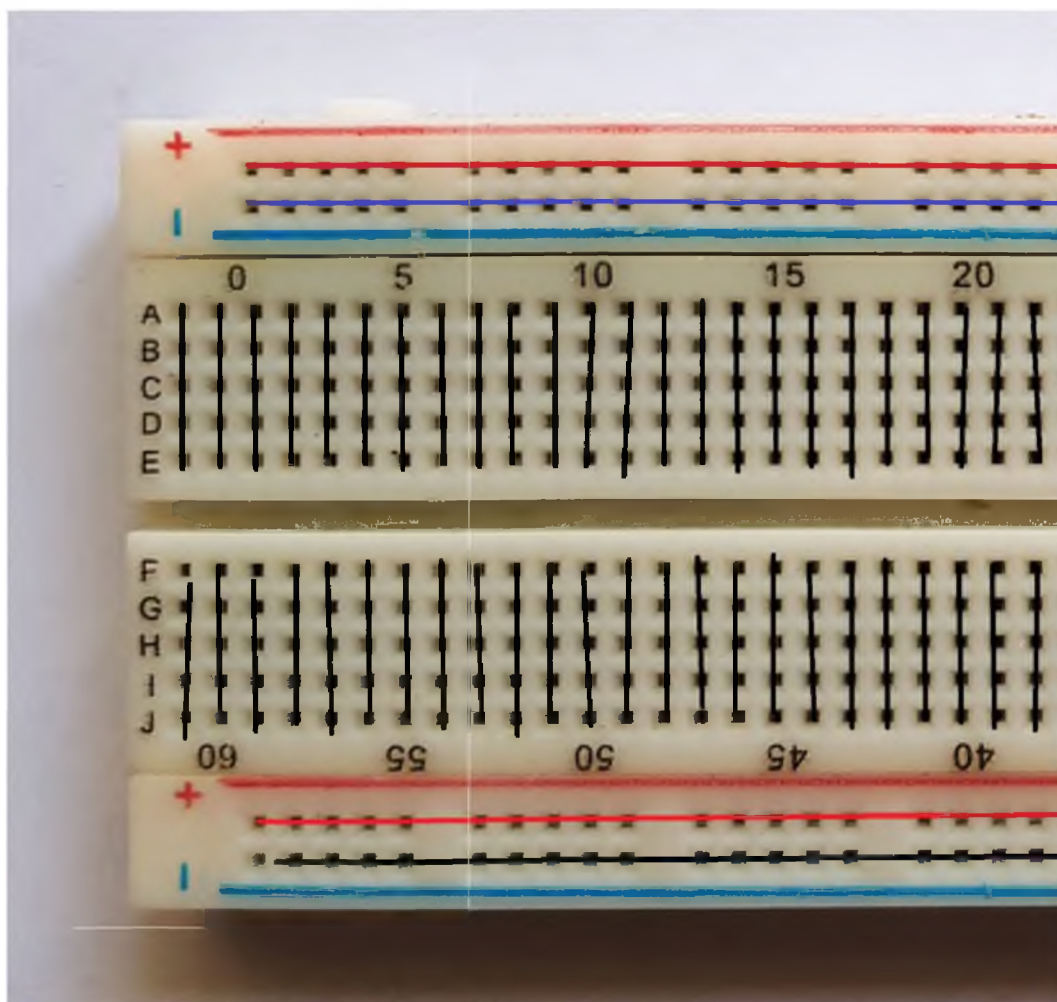


Рисунок 42 - Фрагмент платы с показанными внутренними соединениями

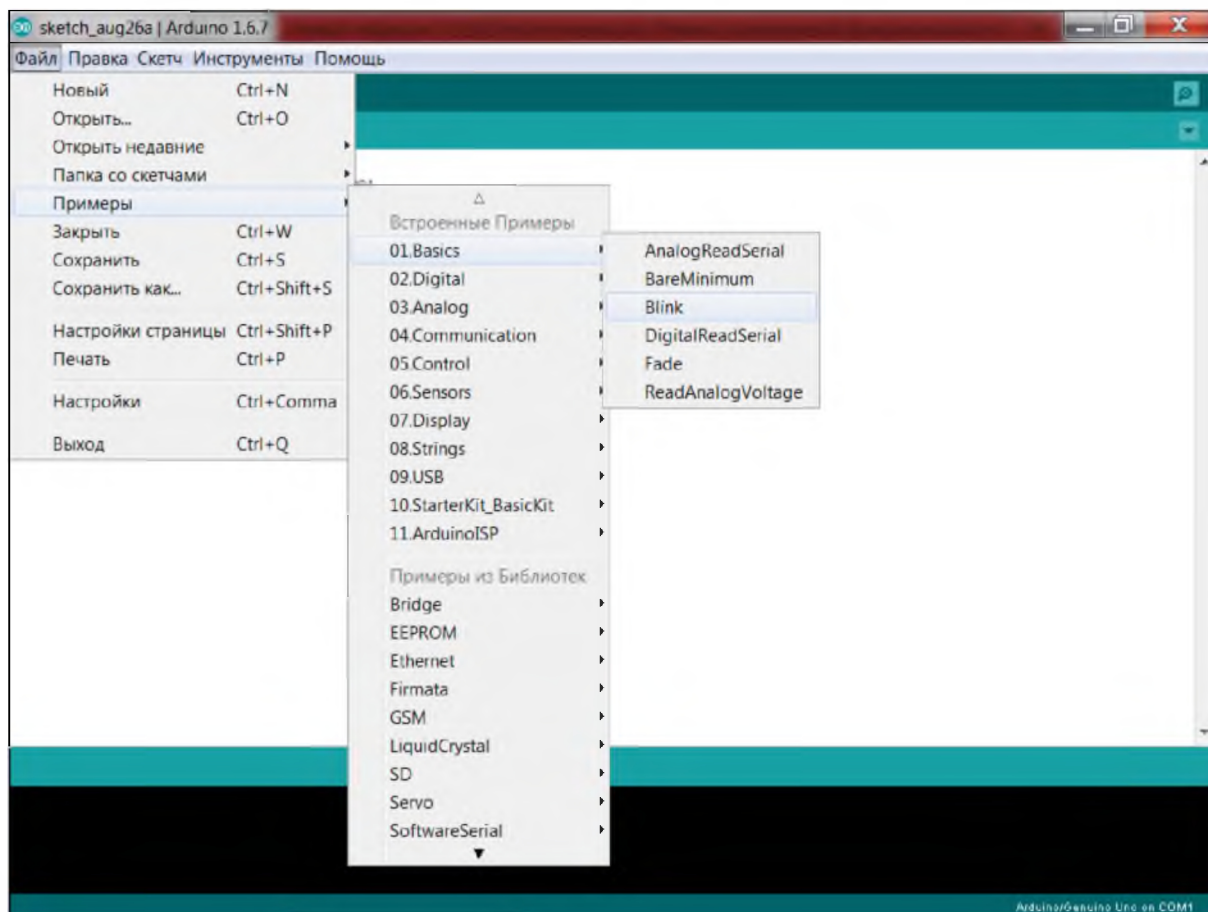


Рисунок 43 - Расположение программы Blink

Теперь пройдем по пути, указанному на рисунке 43 и откроем программу Blink. Программа позволяет мигать светодиодом с периодичностью в 1 секунду.

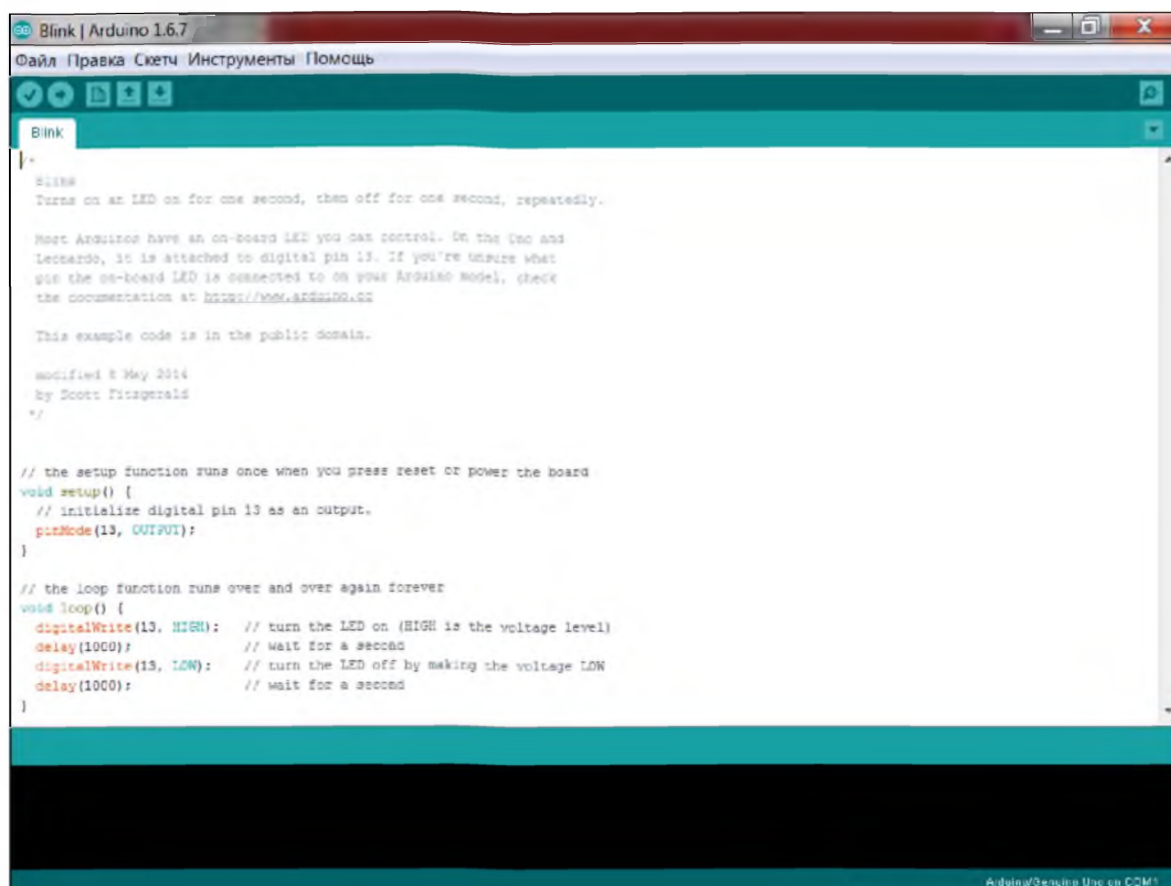


Рисунок 44 - Окно с программой Blink

Рассмотрим ее. Вначале следуют комментарии. Весь текст внутри крайних символов `/*` и `*/` не обрабатывается программой. Следует оставлять такие комментарии для заметок по работе программы, так как с течением времени что-нибудь может подзабыться.

```
/*
```

```
Blink
```

```
Turns on an LED on for one second, then off for one second, repeatedly.
```

```
Most Arduinos have an on-board LED you can control. On the Uno and  
Leonardo, it is attached to digital pin 13. If you're unsure what  
pin the on-board LED is connected to on your Arduino model, check  
the documentation at http://www.arduino.cc
```

```
This example code is in the public domain.
```

modified 8 May 2014

by Scott Fitzgerald

*/

Далее после символов // следует однострочный комментарий. Он также не обрабатывается программой и нужен для пояснения действий в строке кода.

```
// the setup function runs once when you press reset or power the board
```

Далее следует конструкция:

```
void setup() {
```

```
}
```

Стандартная программа void setup() вызывается в начале программы один раз и инициализирует (объявляет в программе) переменные, порты и т.д. при запуске программы. Обратите внимание, что скобки стоят с двух сторон и если выделить одну из них, то высветится и вторая, что бывает удобно при проверке большого кода. При недостатке скобок программа «ругается» при компиляции.

Далее с помощью команды pinMode мы настраиваем цифровой порт, а именно пин (вывод) 13 устанавливаем как выход OUTPUT

```
pinMode(13, OUTPUT);
```

Затем сразу после void setup() идет основная программа void loop():

```
void loop() {
```

```
  digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
```

```
  delay(1000);           // wait for a second
```

```
  digitalWrite(13, LOW); // turn the LED off by making the voltage LOW
```

```
  delay(1000);           // wait for a second
```

```
}
```

В ней с помощью команды digitalWrite включается светодиод на 13 выводе (переводится в состояние HIGH).

```
  digitalWrite(13, HIGH);
```

Затем с помощью команды delay дается пауза в исполнении новых строк кода, новых команд.

```
delay(1000);
```

Учтите, что время указано в миллисекундах. Затем мы отключаем (переводим в состояние LOW светодиод строчкой:

```
digitalWrite(13, LOW);
```

И далее мы опять делаем паузу в 1000 миллисекунд.

Следует скомпилировать готовую программу, загрузить скетч в подключенную плату Arduino и посмотреть результат.

При сборке схемы мигающего светодиода Blink следует подсоединить последовательно со светодиодом сопротивление номиналом 220 Ом, чтобы ограничить ток в цепи. При подсоединении светодиода следует учесть полярность, так как ток при работе протекает от анода к катоду. Вывод катода светодиода короче, чем у анода. Катод подключить к выводу GND (земля) платы.

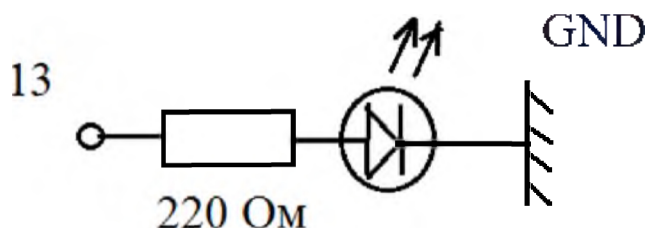


Рисунок 45 - Схема подключения

3.2 Применение команд Arduino

Тип данных:

Переменная типа **Boolean** может принимать два состояния: true (истина) или false (ложь). True соответствует логической единице. False - состояние, отличное от логической единицы. Boolean занимает 1 байт.

Переменная типа **Byte** может принимать значения от 0 до 255, так как 1 байт равен 8 битам.

Переменная типа **Char** символ в одиночных кавычках, занимает 1 байт. Символьные переменные могут иметь значения от -127 до +127.

Переменная типа **Unsigned Char** имеют только положительные значения от 0 до 255.

Переменная типа **Int** (Integer) может принимать значения в диапазоне от -32768 до +32768.

Переменная типа **Long** находится в диапазоне от -2147483648 до +2147483647.

Переменная типа **Unsigned Long** может принимать только положительные значения в диапазоне от 0 до 4294967295.

Переменная типа **Float** может принимать значения от -3.4028235E+38 до +3.4028235E+38.

Переменная типа **String** обозначает массив переменных типа Char и нулевого символа.

Таблица 5. Команды Arduino

Команда	Применение
Функции	<code>void setup() {}</code> <code>void loop() {}</code>
Точка с запятой ;	Завершает команду. При отсутствии компилятор выдаст команду об ошибке
Объявление переменных (тип Integer, значение = 4)	<code>int in1 = 4;</code>
Устанавливает режим работы заданного входа/выхода (pin)	<code>pinMode(led, OUTPUT);</code> OUTPUT — выход INPUT — вход
Комментарии, одна строка	//
Комментарии, несколько строк	/*Комментарий*/
Арифметический оператор синус	<code>sin ()</code> в радианах

нус	
Арифметический оператор косинус	$\cos()$ в радианах
Арифметический оператор тангенс	$\tan()$ в радианах
Арифметический оператор присваивание	=
Арифметический оператор сложения	+
Арифметический оператор вычитание	-
Арифметический оператор деление	/
Арифметический оператор умножение	*
Функция $\text{sq}(x)$	Вычисляет квадрат аргумента
Функция $\text{sqrt}(x)$	Вычисляет квадратный корень числа
Оператор сравнения равно	==
Оператор сравнения не равно	!=
Оператор сравнения больше,	>
Оператор сравнения меньше	<
Оператор сравнения меньше или равно	<=
Оператор сравнения больше или равно	>=
Битовое И	&
Битовое Или	

Битовое Не	~
Логическое И	&&
Логическое ИЛИ	I I
Логическое отрицание	!
Инкремент (увеличение значения переменной на единицу)	++
Декремент (уменьшение значения переменной на единицу)	--
Функция, рассчитывающая наименьшее значение из двух (x или y)	min(x, y)
Функция, рассчитывающая наибольшее значение из двух (x или y)	max(x, y)
Функция, ограничивающая число x в определенном диапазоне	constrain (x,a,b)
Функция, рассчитывающая абсолютную величину (значение x)	abs (x)
Команда, задающая паузу	delay (1000); пауза 1 секунду
Генерация псевдослучайных чисел. Указывается диапазон	random(100,150)
Оператор if	if(условие) {}
Оператор (конструкция) if...else	if (условие){ } else{ }
Конструкция	switch (x) {case 1: case 2: default:}
Выход из цикла	break;

Цикл for	for (x = 1; x < 10; x++) { }
Цикл while, Цикл do while	while (variable < 22) { } или: do { } while (variable < 22);
Инициализация последовательного интерфейса (задает скорость)	Serial.begin(9600);
Завершение работы с последовательным интерфейсом	Serial.end();
Считывание одного байта из последовательного интерфейса	Serial.read();
Выводит данные из последовательного интерфейса	Serial.print();
Выводит данные из последовательного интерфейса и переходит сразу на новую строку	Serial.println();
Выводит данные по байтам из последовательного интерфейса	Serial.write();
Подключение библиотеки	#include <название библиотеки>
Функция, устанавливающая значение HIGH (или LOW) на заданный цифровой выход	digitalWrite(led,HIGH);
Функция, считывающая значение с заданного цифрового входа: HIGH или LOW	digitalRead(led)
Выдает аналоговую величину за счет ШИМ (широтно-импульсной модуляции) на выходе. Задается или в виде константы, или в виде переменной от 0 до 255	analogWrite(led, 255);

Функция, считывающая значение с аналогового входа	<code>analogRead(pin)</code>
Генерирует сигнал (pin, частота x, длительность y)	<code>tone(pin, x, y)</code>
Останавливает сигнал на выходе (pin)	<code>noTone(pin)</code>
Функция, задающая диапазон преобразований. Удобно для работы с большими диапазонами.	<code>map(значения датчика, 0, 1024, 0, 255);</code>

3.3 Частичное замещение элементов схемы реверсивного пускателя на платформе Arduino

Проблема автоматизации промышленных предприятий в настоящее время становится все более явной. В прошлом её решали с применением механических устройств. В энергоснабжении ярким примером является применение реверсивного пускателя для контроля направления вращения электродвигателей.

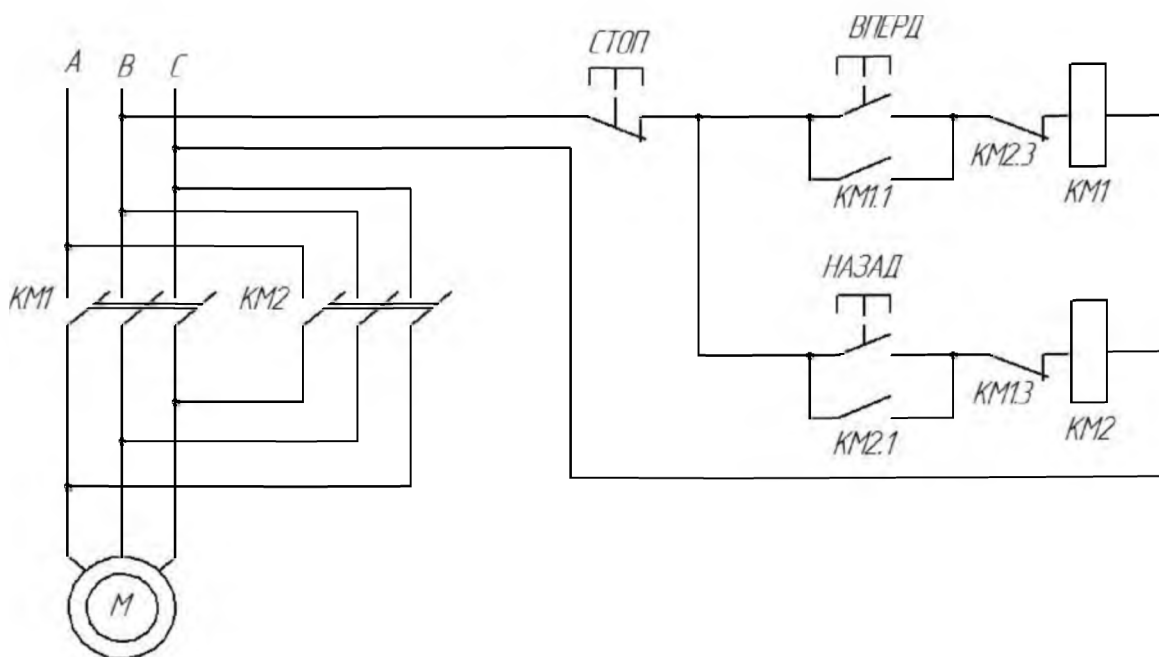


Рисунок 46 - Схема реверсивного пускателя

Данная схема включает в себя две катушки пускателя и кнопочный пост, состоящий из трех кнопок. Суть ее работы сводится к тому, что при подключении катушки К1, нажатием кнопки «Вперед», замыкаются нормально разомкнутые контакты и двигатель начинает вращаться в определенную сторону. Для изменения направления вращения необходимо нажать кнопку «Стоп», а затем подать питание на катушку К2. Во избежание одновременного срабатывания двух катушек применяется своеобразный «предохранитель» (на профессиональном сленге называемый «защита от дурака») в виде нормально замкнутых контактов: КМ1.3 и КМ2.3, которые подают питание на соседние катушки. То есть при срабатывании катушки 2 нормально замкнутый контакт КМ2.3 размыкается и не пропускает через себя ток, делая невозможным подключение катушки 1, пока работает катушка 2. В схематичном изображении данная «конструкция» кажется, довольно простой. Но на практике представляет из себя клубок проводов, в которых человеку, не имеющему профильного образования довольно сложно разобраться. К тому же она требует определенного пространства для размещения.

Выходом из данного положения является применение схем автоматизации, использующих разнообразные аналоги микроконтроллеров. В нашем случае используется микроконтроллер на базе Arduino Uno.

Arduino – это платформа для программирования с открытым кодом на основании встроенного микроконтроллера и среды разработки с программным интерфейсом API. Мозг данной платформы - микроконтроллер Atmega, программируемый на языке “С”, который многим людям всерьез не занимающимся программированием может показаться крайне сложным и запутанным. Однако среда программирования Arduino, основанная на программе Processing, являвшейся подязыком “С”, упрощает эту задачу. Благодаря этому, человек, который раньше даже не имел представления о программировании, через несколько часов свободно сможет создавать собственные программы на Arduino.

Плюсом данной платформы является наличие разнообразных модулей, в том числе реле, которые могут коммутировать соединения напряжением до 220 В и силой тока до 10 А. Чем мы и воспользовались при создании нашей схемы.

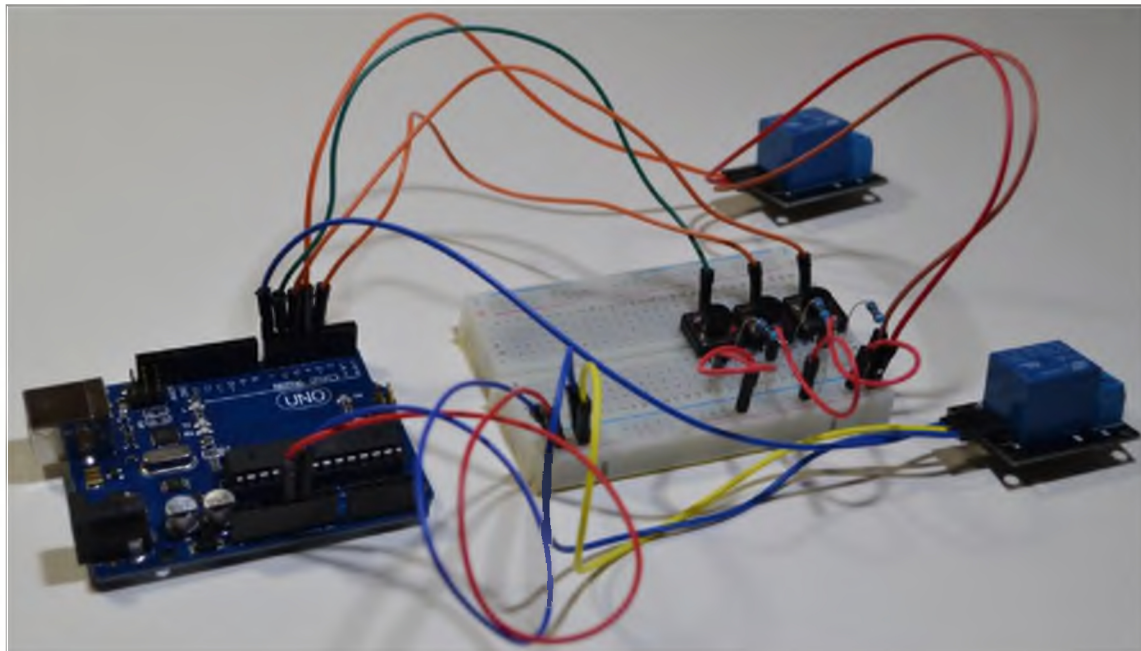


Рисунок 47 - Сборочная схема реле на Arduino вид сбоку

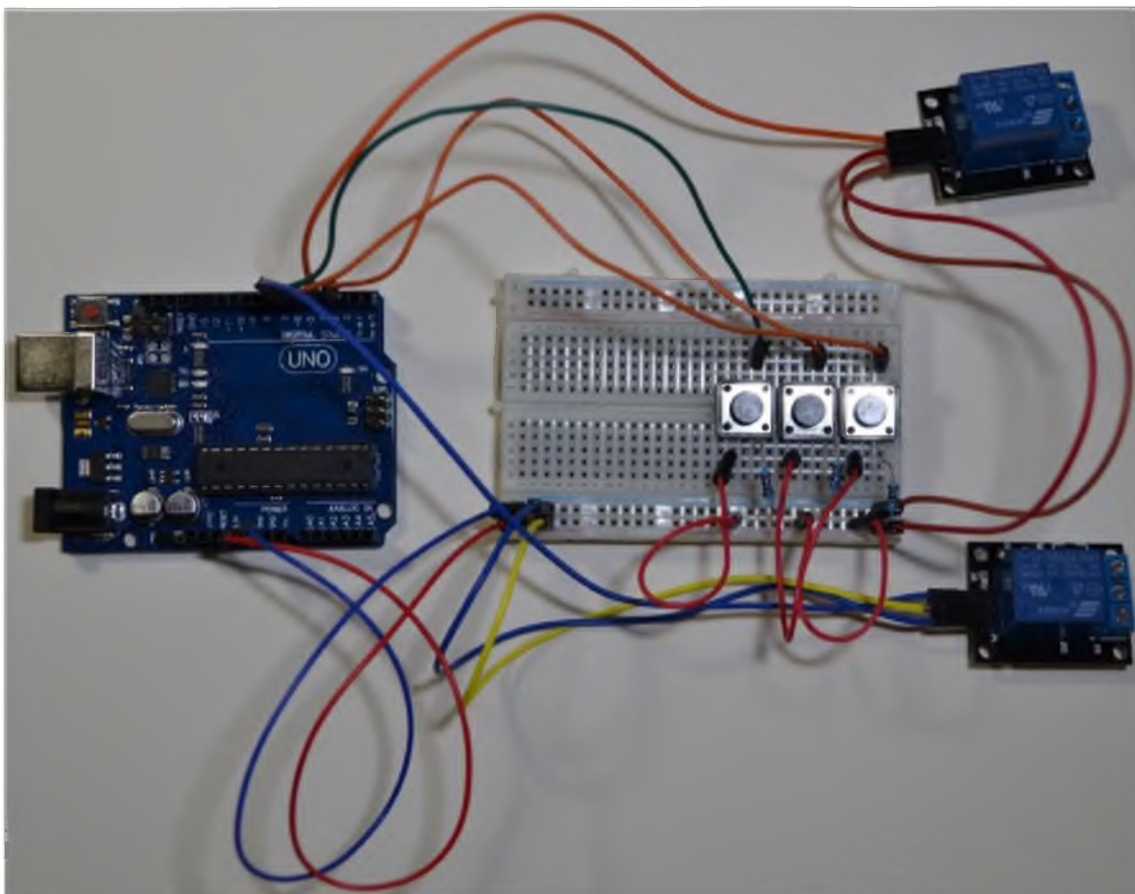


Рисунок 48 - Сборочная схема реле на Arduino вид сверху

Подобная схема занимает минимальное пространство и позволяет реализовать принцип защиты, от одновременного включения катушек используя логику программы.

```
int in1 = 4; “управляющий вывод реле1 на вывод 4”
int in2 = 7; “управляющий вывод реле2 на вывод 7”
int but1 = 5; “кнопка “Вперед” на вывод 5”
int but2 = 6; “кнопка “Назад” на вывод 6”
int but3 = 3; “кнопка “Стоп” на вывод 3”
int flag = 0; “дополнительная переменная состояния”
void setup() { “команда инициализирующая порты, переменные и т.д.”
    pinMode(in1, OUTPUT); “при OUTPUT, контакт устанавливается как выход
”
    pinMode(in2, OUTPUT);
```

```

pinMode(but1, INPUT); “при INPUT, контакт устанавливается как вход ”
pinMode(but2, INPUT);
pinMode(but3, INPUT);
digitalWrite(in1, LOW); “при LOW, на контактах напряжение отсутствует”
digitalWrite(in2, LOW);
}
void loop() “бесконечный цикл”
{
  if (digitalRead(but1) == HIGH && flag == 0) “если нажата кнопка “Вперед” и
  дополнительная переменная равна нулю”
  {
    digitalWrite(in1, HIGH); “включается реле1”
    flag=1; “дополнительная переменная приравняется к единице”
  }
  if (digitalRead(but3) == HIGH) “если нажата кнопка “Стоп”
  {
    digitalWrite(in1, LOW); “выключается реле1”
    digitalWrite(in2, LOW); “выключается реле2”
    flag=0; “дополнительная переменная приравняется к нулю”
  }
  if (digitalRead(but2) == HIGH && flag == 0) “если нажата кнопка “Назад” и
  дополнительная переменная равна нулю”
  {
    digitalWrite(in2, HIGH); “включается реле2”
    flag=2; “дополнительная переменная приравняется к двойке”
  }
}
}

```

В данном коде для реализации защиты от одновременного включения двух пускателей нами была использована дополнительная переменная “flag”, которая исполняет роль своеобразного нормально замкнутого контакта. Из-

начально переменная “flag” приравнена к нулю, но при включении одного из реле она принимает значение отличное от нуля, что не позволяет подать питание на другое реле. Значение переменной “flag” обнуляется при нажатии кнопки “Стоп”.

В схеме были использованы следующие компоненты:

- тактовая кнопка 3 шт.
- резистор (10кОм) 3шт.
- плата для сборки Breadboard
- провода (типа “папа-папа” и “папа-мама”)
- реле низкого уровня 2 шт.

Данная схема является примером применения программирования для замены механических частей и соединений в схеме автоматизации и существенном сокращении размеров аппарата управления.

3.4 Реверс электродвигателя на платформе Arduino

В настоящее время на промышленных предприятиях активно применяются разнообразные схемы автоматизации производства, включающие в себя различные микроконтроллеры и релейные схемы. Использование подобных систем значительно сокращает размеры управляющих устройств, однако зачастую они довольно сложны для обслуживания и требуют значительных денежных затрат.

Достаточно сложные схемы автоматизации можно реализовать, не используя дорогостоящие промышленные контроллеры. В нашем случае мы использовали платформу Arduino и блок реле для реализации схемы реверсивного пускателя.

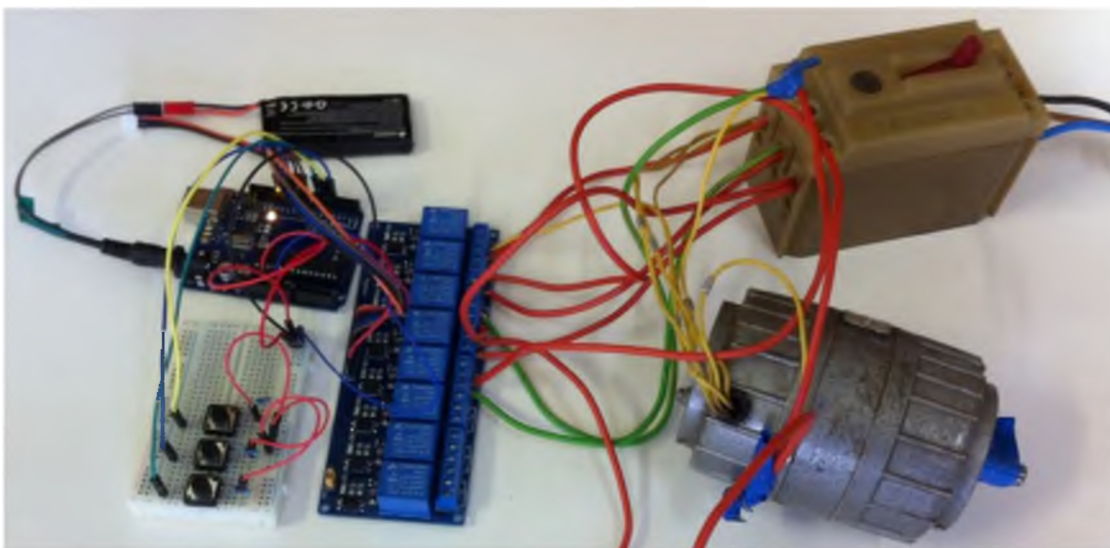


Рисунок 49 - Схема реверсивного пускателя, реализованная на блоке релейных модулей

В настоящей работе нами используется блок восьмиканального реле, представляющий плату на которой собраны релейные модули, имеющие общий канал питания и ноль. Отдельно выведены контакты управления реле, которые подключаются к плате Arduino. В схеме используются пять реле для изменения направления вращения двигателя. Первое реле используется для коммутации только одной фазы, четыре других используются попарно для переключения двух фаз между обмотками двигателя, что позволяет реверсировать вращение. Так как обмотки двигателя соединены по схеме "звезда", то первую обмотку мы подключаем в реле 1, а силовой вывод реле в фазе А. Вторую обмотку подключаем к реле 2 и 5, а их силовые выводы к фазе В. Третью обмотку подключаем к реле 3 и 4, и их силовые выводы к фазе С. Подобная схема позволяет менять местами две фазы что и делает возможным изменять направление вращения двигателя. Также в программном коде предусмотрена система самоподхвата (исключающая необходимость удерживать кнопки постоянно нажатыми) и защита от одновременного нажатия двух кнопок запуска ("защита от дурака").

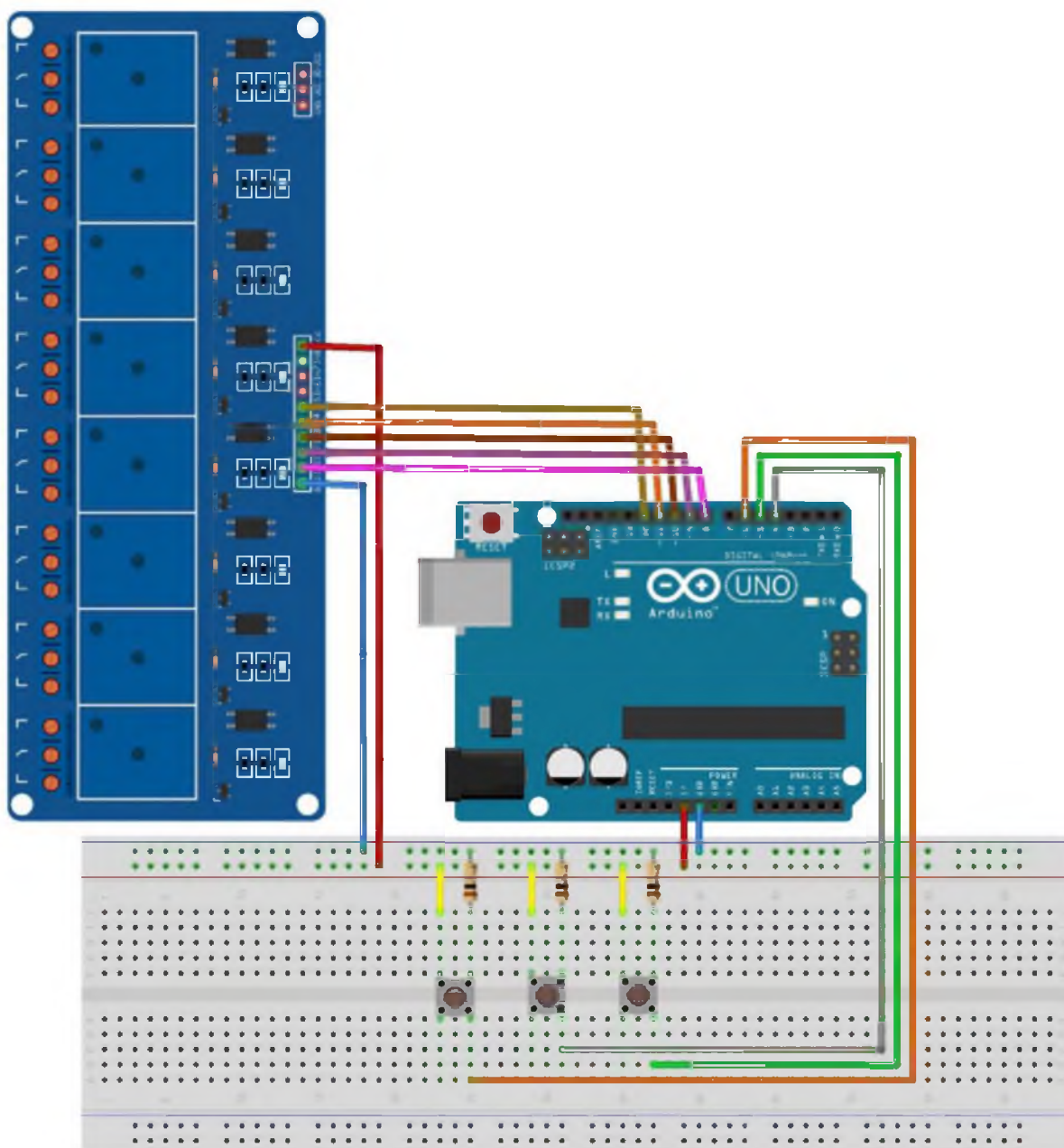


Рисунок 50 - Монтажная схема установки

Код программы управления:

```
int in1 = 8; «управляющий вывод реле 1 на вывод 8»
int in2 = 9; « управляющий вывод реле 2 на вывод 9»
int in3 = 10; «управляющий вывод реле 3 на вывод 10»
int in4 = 11; « управляющий вывод реле 4 на вывод 11»
int in5 = 12; «управляющий вывод реле 5 на вывод 12»
int but1 = 5; «кнопка “НАЗАД” на вывод 5»
int but2 = 6; «кнопка “ВПЕРЕД” на вывод 6»
```

```

int but3 = 4; «кнопка “СТОП” на вывод 4»
int flag = 0; «дополнительная переменная состояния»
void setup() «команда инициализирующая порты, переменные и т.д.»
{
    pinMode(in1, OUTPUT); «при OUTPUT, контакт устанавливается как
ВЫХОД»
    pinMode(in2, OUTPUT);
    pinMode(in3, OUTPUT);
    pinMode(in4, OUTPUT);
    pinMode(in5, OUTPUT);
    pinMode(but1, INPUT); «при INPUT, контакт устанавливается как
ВХОД»
    pinMode(but2, INPUT);
    pinMode(but3, INPUT);
    digitalWrite(in1, HIGH); «при HIGH, на контакты податься напряже-
ние»
    digitalWrite(in2, HIGH);
    digitalWrite(in3, HIGH);
    digitalWrite(in4, HIGH);
    digitalWrite(in5, HIGH);
    digitalWrite(but1, LOW); «при LOW, на контактах напряжение отсут-
ствует»
    digitalWrite(but2, LOW);
    digitalWrite(but3, LOW);
}
void loop() «бесконечный цикл»
{
    if (digitalRead(but1) == HIGH && flag == 0) «если нажата кнопка
“НАЗАД” и дополнительная переменная равна нулю»
    {

```

```

digitalWrite(in1, LOW); «включается реле1»
digitalWrite(in2, LOW); «включается реле2»
digitalWrite(in3, LOW); «включается реле3»
flag=1; «дополнительная переменная приравняется к единице»
}
if (digitalRead(but3) == HIGH) «если нажата кнопка «Стоп»
{
digitalWrite(in1, HIGH); «выключается реле 1»
digitalWrite(in2, HIGH); «выключается реле 2»
digitalWrite(in3, HIGH); «выключается реле 3»
digitalWrite(in4, HIGH); «выключается реле 4»
digitalWrite(in5, HIGH); «выключается реле 5»
flag=0; «дополнительная переменная приравниваться к нулю»
}
if (digitalRead(but2) == HIGH && flag == 0 «если нажата кнопка «ВПЕ-
РЕД» и дополнительная переменная равна нулю»
{
digitalWrite(in1, LOW); «включается реле 1»
digitalWrite(in4, LOW); «включается реле 4»
digitalWrite(in5, LOW); «включается реле 5»
flag=2; «дополнительная переменная приравниваться к двойке»
}
}

```

В схеме были использованы следующие компоненты:

- тактовая кнопка 3 шт.
- резистор (10кОм) 3шт.
- плата для сборки Breadboard
- провода (типа “папа-папа” и “папа-мама”)
- сборка реле низкого уровня (восьмиканальное реле).

Применяемые нами реле позволяют коммутировать напряжения до 250 В и токи до 10 А в сети переменного тока, что позволяет применять их для управления нагрузкой до 2 кВт. Преимуществом данных реле является низкое управляющее напряжение 5 В.

3.5 Схема парктроника, реализованная на Arduino

Владельцы отечественных авто, особенно неопытные водители, сталкиваются с проблемой парковки своего автомобиля. Далеко не все могут сразу освоить данный элемент вождения и на это потребуется определённый промежуток времени. Многие люди поначалу плохо ориентируются по зеркалам, во время парковки своего авто и прибегают к помощи прохожих. Но случается так, что вокруг может никого не оказаться и "чайнику" приходится выпутываться из ситуации самостоятельно, что может привести к неутешительным последствиям, а именно, повреждение своего авто вследствие неаккуратного выполнения манёвров. На помощь таким автолюбителям приходит парктроник. Это устройство замеряет расстояние между авто и окружающими его предметами и передаёт эту информацию водителю, который опираясь на помощь данной электрической системы, осуществляет безопасную парковку своего автомобиля.

На сегодняшний день, цены на такой девайс хорошей сборки могут ударить по карману автовладельца, а дешёвые сборки не всегда работают как положено. Далее будет предложено создать недорогой вариант парктроника.

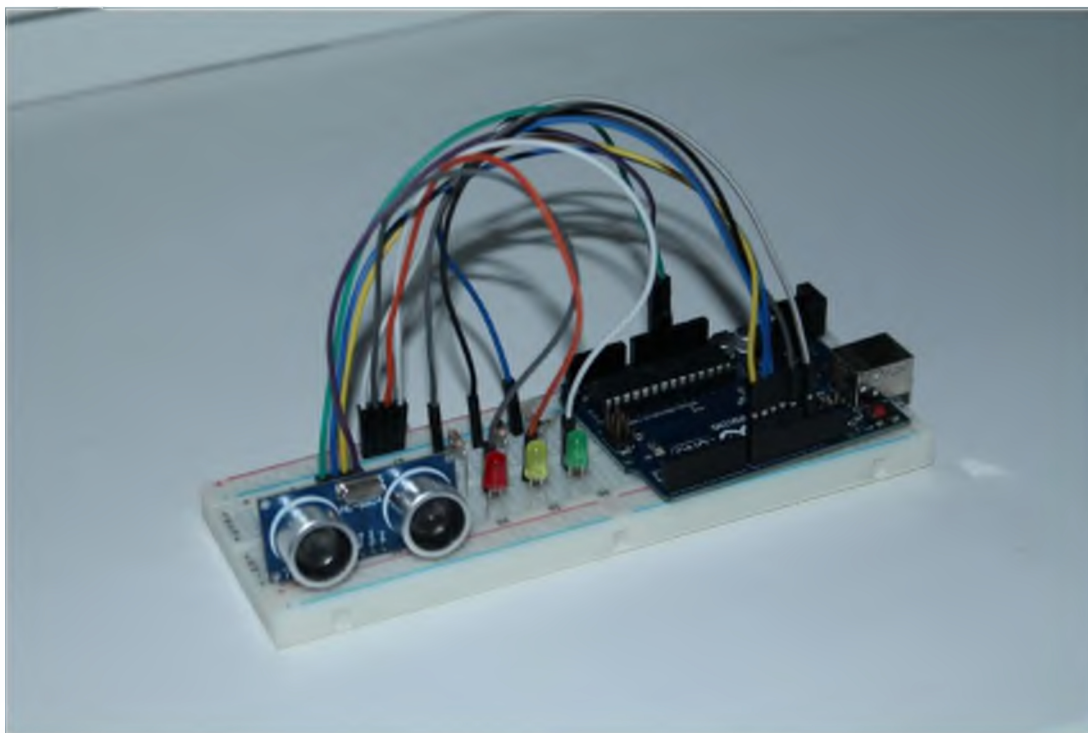


Рисунок 51 - Схема парктроники, собранная на Arduino, вид сбоку

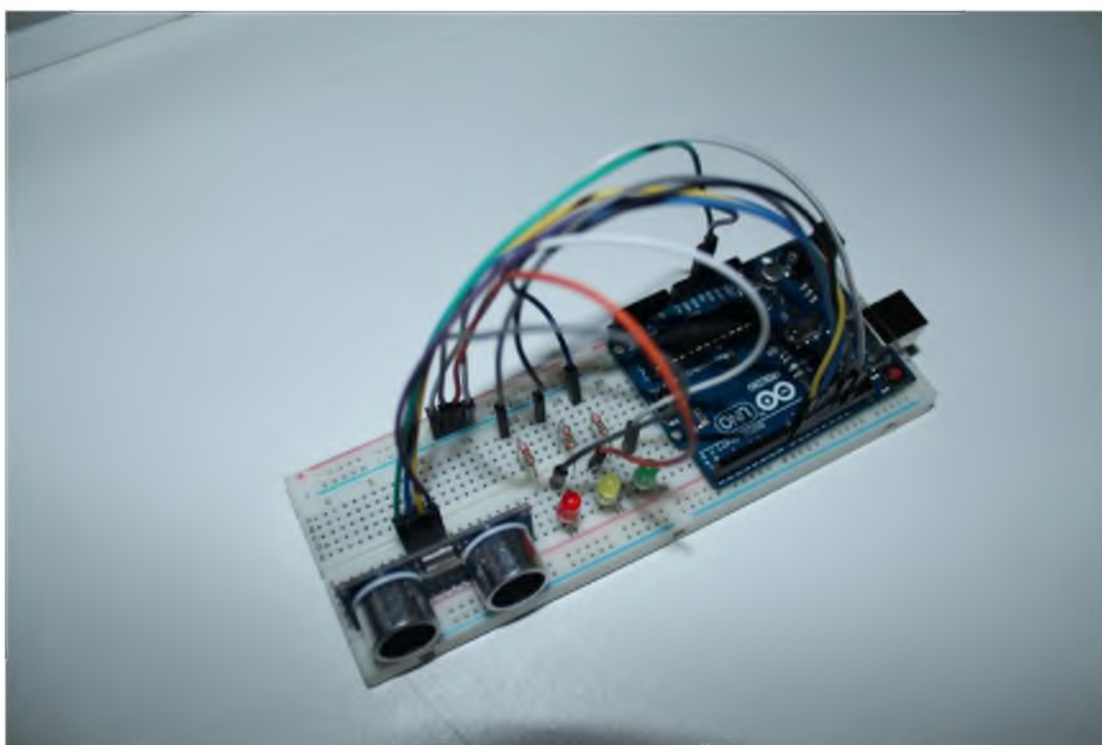


Рисунок 52 - Схема парктроники, собранная на Arduino, вид сверху

Рассмотрим программный код, загруженный на плату:

```
#define echoPin 8 "echo контакт датчика на 8 вывод"
```



```

#define trigPin 9    "trig контакт датчика на 9 вывод"
int LEDred=12;      "красный светодиод на 12 вывод"
int LEDyellow=11;   "жёлтый светодиод на 11 вывод"
int LEDgreen=10;    "зелёный светодиод на 10 вывод"
int Speaker=6;      "пищалка на 6 вывод"
void setup() {      "эта команда инициализирует порты, переменные и
т.д"

    // put your setup code here, to run once:
    Serial.begin (9600);    "скорость обмена данными между пк и платой
Arduino"

    pinMode(Speaker,OUTPUT); "при OUTPUT, контакт устанавливается
как выход"

    pinMode(LEDred,OUTPUT);
    pinMode(LEDyellow,OUTPUT);
    pinMode(LEDgreen,OUTPUT);
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT); "при INPUT, контакт устанавливается как
вход"

    digitalWrite(LEDgreen,LOW); "при LOW на контактах напряжение от-
сутствует"

    digitalWrite(LEDyellow,LOW);
    digitalWrite(LEDred,LOW);
    digitalWrite(Speaker,LOW);
}

void loop() {        "бесконечный цикл"

    // put your main code here, to run repeatedly:
    int duration, cm;    "объявление переменной duration типа int"
    digitalWrite(trigPin, LOW);    "на данном этапе мы прописываем код,
который

    delayMicroseconds(2);    будет активировать датчик, для считывания

```



```

digitalWrite(trigPin, HIGH);    расстояния до окружающих предметов"
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
cm = duration / 58;
Serial.print(cm);
Serial.println(" cm");
delay(200);
if(cm>30)                        "если расстояние больше 30см"
{
    digitalWrite(LEDgreen,HIGH);  "то начинает светиться зелёный свето-
диод"
    digitalWrite(LEDyellow,LOW);
    digitalWrite(LEDred,LOW);
}
if(cm<30)                        "если расстояние меньше 30см"
{
    digitalWrite(LEDgreen,LOW);
    digitalWrite(LEDyellow,HIGH); "то начинает светиться жёлтый свето-
диод"
    digitalWrite(LEDred,LOW);
}
if(cm<20)                        "если расстояние меньше 20 см"
{
    digitalWrite(LEDyellow,LOW);
    digitalWrite(LEDred,HIGH);    "то начинает светиться красный све-
тодиод"
    tone(Speaker,550,450);
}
}

```

Данный код работает как при приближении к препятствию, так и при отдалении от него. При расстоянии более 30см светится зелёный светодиод, от 30см до 20см светится жёлтый светодиод, меньше 20см светится красный светодиод и пищалка начинает издавать звук.

В схеме были использованы следующие компоненты:

- светодиоды (красный, жёлтый, зелёный) 3шт
- пьезопреобразователь
- резисторы (1.5кОм) 3шт
- плата для сборки Breadboard
- провода (типа "папа-папа")
- ультразвуковой датчик измерения расстояния HC-SR04



Рисунок 53 - Ультразвуковой датчик измерения расстояния HC-SR04

Принцип данного датчика прост. Посылается звуковой сигнал, при подаче импульса на вывод trig длиной 10мкс, вслед за этим будут выпущены 8 импульсов, на echo выводе будет высокий уровень. Активируется режим ожидания отражённого сигнала. После его приёма, уровень на выводе echo станет низким.

Рассмотрим схему подключения: к выводу 9 на плате подключается trig контакт датчика, к 8 подключается echo контакт, масса датчика gnd соединяется с соответствующим выводом на плате, vcc подсоединяется к выводу 5v, катод красного светодиода к выводу 12 и через резистор сопротивлением 1.5кОм(для остальных светодиодов аналогично), жёлтый к 11, зелёный к 10, их аноды соединяются параллельно на общую минусовую шину, которая в свою очередь идёт к выводу gnd на плате, плюс пищалки соединен с 6 выводом, а минус с gnd.

Схема является бюджетной и очень простой. Этот парктроник станет незаменимым помощником для начинающего водителя.

3.6 Выводы по платформе Arduino

Приведенные выше примеры дают базу для обучения программированию и не ограничиваются какой то определенной областью применения. На рисунке 54 показана гусеничная платформа с манипулятором, с управлением на расстоянии. При перепрограммировании самостоятельно работает от датчиков по заданному алгоритму.



Рисунок 54 - Гусеничная платформа с манипулятором

Как видите, дальнейшее применение зависит только от Вашего желания и фантазии. То есть платформа Arduino позволяет с наименьшим уровнем сложности начать заниматься программированием микропроцессоров и сборкой роботов. Arduino популярно во всем мире и является открытой платформой (открытая техдокументация по плате, коду и т.д.). И что немаловажно, наименее затратной в финансовом плане.

Вопросы для самопроверки:

1. Какова последовательность установки Arduino IDE?
2. Перечислите основные элементы платы Arduino Uno.
3. Что выполняет программа Blink?
4. Объясните строчки кода программы Blink.
5. Для чего нужна компиляция?

ЗАКЛЮЧЕНИЕ

В данном пособии было предложено изучение автоматизации производственных процессов и основ программирования на конкретных примерах. Начальные примеры были подобраны из релейно-контакторной логики, первым, с чего начинается изучение электротехники, то есть даны на базе уже освоенного, что облегчает задачу обучения. Следующие примеры даны с усложнением. Сталкиваясь с каждой производственной задачей, приходится проходить через несколько этапов:

- постановка задачи;
- изучение технологического процесса;
- выбор конкретного оборудования, исходя из параметров производственного процесса;
- написания программы для ПЛК или микропроцессора, отладка программы;
- и, наконец, монтаж оборудования и наладка.

При выборе ПЛК рекомендуется остановить свой выбор на ПЛК Овен с CODESYS. Приведенные уроки следует изучить. Они созданы с позиции обучения. Овен приводит на своем официальном сайте документацию, имеется форум и техподдержка.

В пособии использованы работы /6-45/.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Электрические и электронные аппараты. Под ред. Розанова Ю.К. - М.: Энергоатомиздат, 2001. - 730 с.
2. Счётчик ватт-часов активной энергии переменного тока статический «Меркурий 200». Руководство по эксплуатации. 2007
3. Министерство регионального развития Российской Федерации. Свод правил СП31.13330.2012 Водоснабжение. Наружные сети и сооружения. Актуализированная редакция СНиП 2.04.02-84*, издание официальное, Москва, 2012
4. Поение коров - нормы потребления воды и организация водопоя. Ссылка <https://farmers.ru/zhivotnovodstvo/krs/poenie> (дата обращения 01.12.17)
5. Системы автоматизации, автоматизация технологических процессов, АСУ ТП [Электронный ресурс] URL: http://smi.su/product/sistemy_avtomatizatsii/ (дата обращения: 20.09.2016)
6. Татинцян А.А., Моногаров С.И. «Программа управления системой охлаждения в CODESYS» В книге: Наука и технологии в нефтегазовом деле Тезисы докладов Международной научно-практической конференции, посвященная 100-летию Кубанского государственного технологического университета и 25-летию кафедры машин и оборудования нефтяных и газовых промыслов Армавирского механико-технологического института. 2018. С. 306-308.
7. Татинцян А.А., Моногаров С.И. «Программа управления насосной пожаротушения в CODESYS» В книге: Наука и технологии в нефтегазовом деле Тезисы докладов Международной научно-практической конференции, посвященная 100-летию Кубанского государственного технологического университета и 25-летию кафедры машин и оборудования нефтяных и газо-

вых промыслов Армавирского механико-технологического института. 2018. С. 304-306.

8. Сморкачев А.А., Моногаров С.И. «Обзор и анализ готовых платформ для создания роботов» Международный студенческий научный вестник. 2018. № 3-2. С. 226-229.

9. Татинцян А.А., Моногаров С.И. «Основные функциональные блоки в CODESYS» В сборнике: Сборник докладов победителей и лауреатов XXIII студенческой научной конференции АМТИ 2017. С. 95-98.

10. Моногаров С.И. «Автоматизация производственных процессов» Учебное пособие по изучению дисциплины для студентов всех форм обучения направления 13.03.02 - Электроэнергетика и электротехника / Армавир, 2017.

11. Пожидаев Н.К., Моногаров С.И. «Автоматизация процесса сбора отходов» В сборнике: Развитие природоохранной системы и экологии города. Материалы региональной научно-практической молодежной интернет-конференции. 2017. С. 213-217.

12. Моногаров С.И. «Электрические и электронные аппараты» Учебное пособие / Армавир, 2017.

13. Койдо В.В., Моногаров С.И. «Современные устройства автоматизации систем управления» В сборнике: Сборник докладов победителей и лауреатов XXII студенческой научной конференции АМТИ. 2016. С. 122-124.

14. Моногаров С.И. «Системы цифровой обработки сигналов» Методические указания по самостоятельной работе студентов всех форм обучения направления 09.03.01 Информатика и вычислительная техника / Армавир, 2015.

15. Моногаров С.И. «Системы цифровой обработки сигналов» Методические указания по лабораторным работам для студентов всех форм обучения направления 09.03.01 Информатика и вычислительная техника / Армавир, 2015.

16. Моногаров С.И. «Микропроцессорные устройства контроля и управления» Методические указания по самостоятельной работе студентов для всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

17. Моногаров С.И. «Электрические и электрические аппараты» Методические указания по выполнению курсового проекта для студентов всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

18. Моногаров С.И. «Основы микропроцессорной техники» Методические указания по выполнению лабораторных работ для студентов всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

19. Моногаров С.И. «Основы микропроцессорной техники» Методические указания по изучению дисциплины и выполнению контрольных работ для студентов всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и предприятий») / Армавир, 2015.

20. Моногаров С.И. «Основы микропроцессорной техники» Методические указания по самостоятельной работе студентов для всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и предприятий») / Армавир, 2015.

21. Моногаров С.И. «Микропроцессорные устройства контроля и управления» Методические указания по выполнению лабораторных работ для студентов всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

22. Моногаров С.И. «Микропроцессорные устройства контроля и управления» Методические указания по изучению дисциплины и выполнению контрольных работ для студентов заочной формы обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

23. Моногаров С.И. «Системы цифровой обработки сигналов» Методические указания по изучению дисциплины и выполнению контрольных работ для студентов заочной формы обучения направления 09.03.01 Информатика и вычислительная техника / Армавир, 2015.

24. Моногаров С.И. «Электрические и электронные аппараты» Методические указания по выполнению лабораторных работ для студентов всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

25. Моногаров С.И. «Электрические аппараты» Методические указания по изучению дисциплины и выполнению контрольных работ для студентов заочной формы обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

26. Моногаров С.И. «Электротехника, электроника и схемотехника» Методические указания по самостоятельной работе студентов всех форм обучения направления 09.03.01 Информатика и вычислительная техника / Армавир, 2015. Том Часть 1

27. Моногаров С.И. «Электротехника, электроника и схемотехника» Методические указания по изучению дисциплины и выполнению контрольных работ для студентов заочной формы обучения направления 09.03.01 Информатика и вычислительная техника / Армавир, 2015. Том Часть 1

28. Моногаров С.И. «Электрические аппараты» Методические указания по выполнению лабораторных работ для студентов всех форм обучения

направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

29. Моногаров С.И. «Электротехника, электроника и схемотехника» Методические указания по самостоятельной работе студентов всех форм обучения направления 09.03.01 Информатика и вычислительная техника / Армавир, 2015. Том Часть 2

30. Моногаров С.И. «Электротехника, электроника и схемотехника» Методические указания по выполнению лабораторных работ для студентов всех форм обучения направления 09.03.01 Информатика и вычислительная техника / Армавир, 2015. Том Часть 2

31. Моногаров С.И. «Электрические и электронные аппараты» Методические указания по самостоятельной работе студентов для всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

32. Моногаров С.И. «Электрические и электронные аппараты» Методические указания к практическим (семинарским) занятиям для студентов всех форм обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

33. Моногаров С.И. «Электрические и электронные аппараты» Методические указания по изучению дисциплины и выполнению контрольных работ для студентов заочной формы обучения направления 13.03.02 Электроэнергетика и электротехника (профиль «Электрооборудование и электрохозяйство предприятий, организаций и учреждений») / Армавир, 2015.

34. Моногаров С.И. «Электротехника, электроника и схемотехника» методические указания по выполнению лабораторных работ для студентов всех форм обучения направления 109.03.01 Информатика и вычислительная техника / Армавир, 2015. Том ЧАСТЬ 1

35. Моногаров С.И., Коняев И.С. «Исследование микроконтроллера at tiny 2313» В сборнике: Сборник докладов по материалам юбилейной XX студенческой научной конференции АМТИ Кубанский государственный технологический университет, Армавирский механико-технологический институт; Москвитин А.А. ответственный редактор, Шарнов А.И. зам. ответственного редактора. 2014. С. 94-97.

36. Моногаров С.И., Рахманин Е.Г. «Перспективы применения цифровых счетчиков электроэнергии» В сборнике: Сборник докладов по материалам юбилейной XX студенческой научной конференции АМТИ Кубанский государственный технологический университет, Армавирский механико-технологический институт; Москвитин А.А. ответственный редактор, Шарнов А.И. зам. ответственного редактора. 2014. С. 80-82.

37. Моногаров С.И. «Электрические аппараты» Методические указания по выполнению лабораторных работ для студентов всех форм обучения и МИППС направления 140400.62 Электроэнергетика и электротехника / Краснодар, 2014. Том Часть 1

38. Моногаров С.И. «Опыт практико-ориентированного обучения студентов технических специальностей» В сборнике: Теоретические и методологические проблемы современного образования Материалы IX международной научно-практической конференции . 2012. С. 60-61.

39. Моногаров С.И., Гридин С.М. «Автоматизированное управление экструзионной установкой» В сборнике: Сборник лучших докладов XIX Студенческой научной конференции АМТИ сборник лучших докладов XIX Студенческой научной конференции Армавирского механико-технологического института (АМТИ), посвященной 95-летию КубГТУ. под общей редакцией А.И. Шарнова. 2013. С. 93-98.

40. Моногаров С.И., Тимченко Ю.Н. «Возможное использование eeprom памяти» В сборнике: Сборник лучших докладов XIX Студенческой научной конференции АМТИ сборник лучших докладов XIX Студенческой научной конференции Армавирского механико-технологического института

(АМТИ), посвященной 95-летию КубГТУ. под общей редакцией А.И. Шарнова. 2013. С. 144-151.

41. Моногаров С.И., Гридин С.М. «Автоматизированное управление нанесением клея на деталь» В сборнике: Научный потенциал вуза - производству и образованию материалы Региональной научно-практической конференции, посвященной 75-летию Краснодарского края и 95-летию КубГТУ. Редакторы: А.А. Москвитин, О.А. Лучков. 2013. С. 180-183.

42. Моногаров С.И., Коняев И.С. «Принципы построения больших интегральных схем» Научно-методический электронный журнал Концепт. 2014. № S20. С. 3126-3130.

43. Рахманин Е.Г., Моногаров С.И. «Принципы организации и функционирования автоматизированных систем коммерческого учёта электроэнергии» Научно-методический электронный журнал Концепт. 2014. № S20. С. 1671-1675.

44. Коняев И.С., Моногаров С.И. «Анализ работы универсального программатора для микроконтроллеров семейства AVR» Международный научно-исследовательский журнал. 2014. № 6-1 (25). С. 52-54.

45. Мищенко М.Е., Моногаров С.И. «Применение компьютерных технологий в практическом образовании инженера электрика» Международный научно-исследовательский журнал. 2012. № 3 (3). С. 13-15.

46. Николайчук Н. О. Современная автоматизация. М.: СОЛОН-Пресс, 2003. 256 с.

47. Arduino. Базовый набор. Руководство пользователя. – СПб.: БХВ-Петербург, 2015.

ПРИЛОЖЕНИЯ

Лабораторная работа № 1

Исследование магнитного пускателя

Магнитный пускатель – это электрический аппарат, предназначенный для пуска, остановки, реверсирования и защиты электродвигателей.

Его практически единственное отличие от контактора - наличие устройства защиты (обычно – тепловое реле) от токовых перегрузок. Бесперебойная работа асинхронных двигателей в значительной степени зависит от надежности пускателей. Поэтому к ним предъявляются высокие требования в отношении четкости срабатывания, надежности защиты двигателя от перегрузок, минимального потребления мощности.

Особенности условий работы пускателя состоят в следующем. При включении асинхронного двигателя пусковой ток достигает 6-7-кратного значения номинального тока. Даже незначительная вибрация контактов при таком токе быстро выводит пускатель из строя. Это накладывает высокие требования в отношении вибрации контактов и их износа.

При отключении пускатель работает в облегченных условиях, так как напряжение на контактах равно разности напряжения сети и ЭДС двигателя, составляющей всего 15-20% от номинального.

В технических данных пускателя указывается не только его номинальный ток, но и мощность двигателя, с которым пускатель может работать при различных напряжениях.

При необходимости повысить срок службы пускателя целесообразно выбрать его с запасом по мощности. При уменьшении мощности двигателя возрастает допустимое число включений в час. Дело в том, что двигатель меньшей мощности быстрее достигает номинальной скорости вращения. Поэтому при отключении пускатель разрывает установившийся номинальный ток двигателя, что облегчает работу пускателя.

С учетом исключительно широкого распространения пускателей большое значение имеет снижение потребляемой ими мощности. В пускателе мощность расходуется в электромагните и тепловом реле.

Для защиты двигателя от перегрузки в двух фазах устанавливаются тепловые реле.

Схема включения нереверсивного пускателя показана на рисунке 1. Главные контакты К включаются в разрыв проводов силовой цепи, питающей двигатель М. В проводах двух фаз включаются также нагревательные элементы тепловых реле F2. Катушка электромагнита контактора К подключается к сети через кнопки управления S1, S2 и размыкающий контакт теплового реле F2. При нажатии кнопки S2 напряжение на катушку подается через замкнутые контакты кнопки S1 и замкнутые контакты тепловых реле F2. После притяжения якоря электромагнита замыкается блок-контакт К, шунтирующий контакты кнопки S2. Это дает возможность отпустить кнопку «Пуск». Для отключения пускателя нажимается кнопка S1 («Стоп»). При перегрузке двигателя срабатывает тепловое реле F2, которое разрывает цепь катушки К, якорь электромагнита отходит и разрывает контакты К, отключая двигатель от сети.

Пускатель обеспечивает также защиту двигателя от понижения напряжения питания. Электромагнит отпускает якорь при напряжении 60-70% от номинального. Если напряжение сети возрастает до своего номинального значения, самопроизвольного включения пускателя не происходит, так как блок-контакт К разомкнется при отпуске якоря.

Схема включения реверсивного пускателя приведена на рисунке 3. Кнопка управления «Вперед» имеет замыкающие контакты 1-2 и размыкающие контакты 4-6. Аналогичные контакты имеет кнопка пуска двигателя в обратном направлении «Назад». При пуске «Вперед» замыкаются контакты 1-2 этой кнопки и процесс протекает так же, как и у нереверсивного пускателя, с той лишь разницей, что цепь катушки КВ замыкается через размыкающие контакты 1-6 кнопки «Назад». Одновременно разрывается цепь катушки

КН. При нажатии кнопки «Назад» в начале размыкаются контакты 1-6, обесточивается катушка КВ и отключается пускатель «Вперед». Затем контактами 4-3 запускается электромагнит пускателя «Назад». При одновременном нажатии кнопок «Вперед» и «Назад» ни один из пускателей не будет включен.

Схема включения нереверсивного пускателя может быть реализована на программируемом реле. На рисунке 2 показано подключение ко входам программируемого реле кнопок П и С для подачи команды на пуск и остановку двигателя. К выходам реле подключена катушка контактора КМ1 и сигнальная лампа НЛ1.

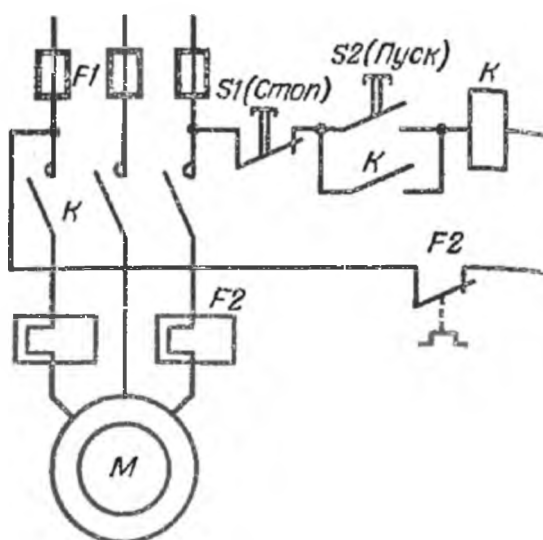


Рисунок 1 Схема включения нереверсивного пускателя

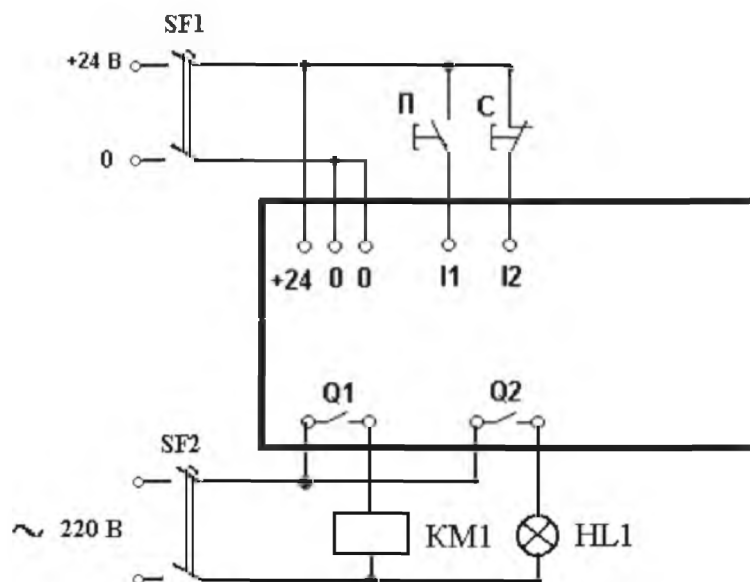


Рисунок 2 Цепи управления программируемого реле

Цель и программа работы

Целью работы является изучение реверсивного пускателя в схеме управления и защиты асинхронного короткозамкнутого электродвигателя.

Программа работы включает:

1. Ознакомление с конструкцией магнитных пускателей.
2. Изучение схемы включения нереверсивного и реверсивного пускателей.
3. Определение напряжения на катушке электромагнита, при котором происходит замыкание и размыкание контактов пускателя.

Описание установки

Лабораторная работа выполняется на установке, схема которой показана на рисунке 3. К питающей сети напряжением 380 В установка подключается с помощью автомата В. С помощью автотрансформатора Тр можно плавно изменять напряжение на катушках КВ и КН магнитных пускателей. Величина напряжения контролируется по показаниям вольтметра. Переключатель В1 служит для шунтирования контактов 1-2 или 3-4 кнопок управления «Вперед» и «Назад» соответственно.

При этом обеспечивается подача напряжения на катушки КВ или КН магнитных пускателей при отпущенных кнопках «Вперед» или «Назад». Используется при определении напряжения срабатывания и напряжения отпущения электромагнита пускателя.

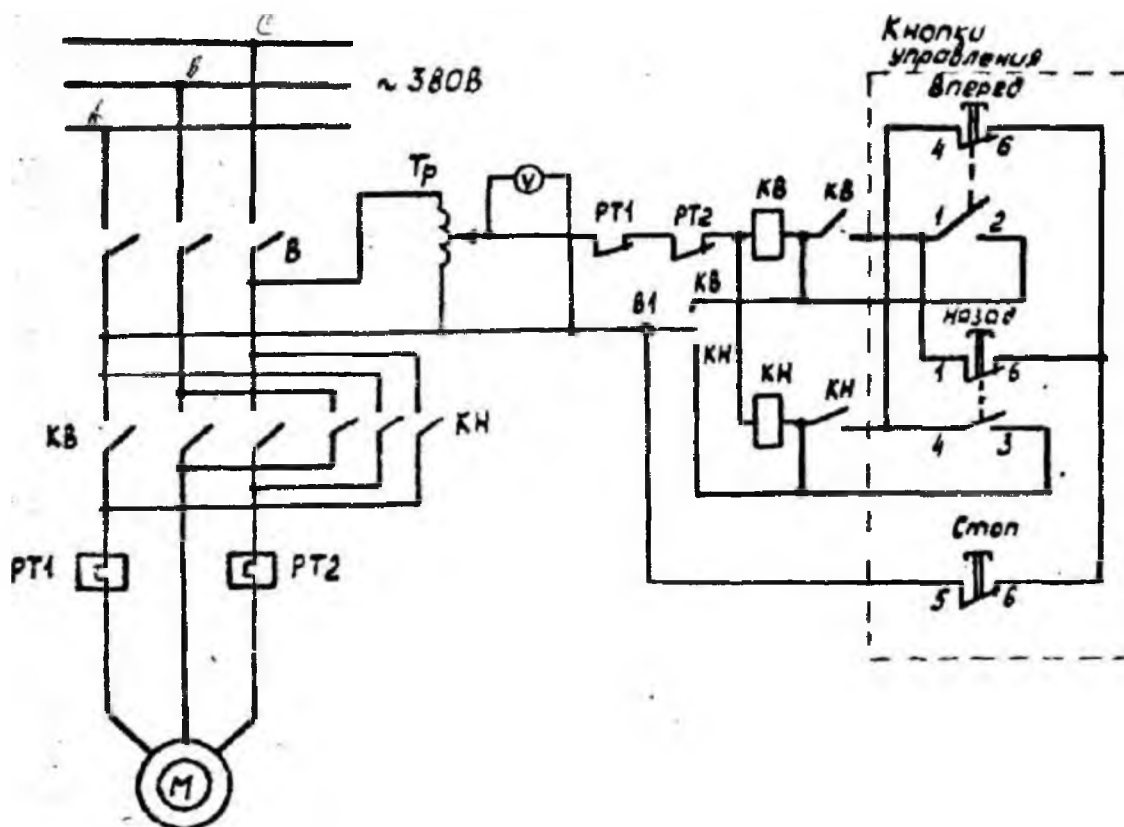


Рисунок 3 Схема лабораторной установки

Методические указания

Работу следует выполнять в следующей последовательности:

1. Изучить конструкцию магнитных пускателей.
2. Изучить схемы включения нереверсивного и реверсивного пускателей (рисунки 1, 2 и 3).
3. Ознакомиться с лабораторной установкой, назначением и работой всех ее элементов.
4. Произвести определение напряжения срабатывания и напряжения отпускания электромагнита пускателя. Для этого:
 - установить ЛАТРоМ минимальное напряжение в цепи управления;
 - переключатель В1 установить в положение «КВ» и, плавно повышая

напряжение до срабатывания контактора, заметить показание вольтметра и записать его;

- плавным уменьшением напряжения определить по вольтметру напряжение отпускания электромагнита контактора;

- переключатель В1 установить в положение «КН» и в такой же последовательности определить напряжение срабатывания и отпускания контактора КН. Полученные данные записать в таблицу;

- установить переключатель в положение «О».

Таблица результатов измерений и вычислений

Исследуемый контактор	Номера опытов	Напряжение включения, $U_{вк}, В$	Напряжение отпускания, $U_{от}, В$	Коэффициент возврата, $K = U_{от}/U_{вк}$
КВ	1			
	2			
	3			
КН	1			
	2			
	3			

5 Произвести реверсивное включение асинхронного электродвигателя кнопками «Вперед» и «Назад», останавливая его кнопкой «Стоп» перед каждым пуском.

6 После окончания работ выключить установку от сети автоматическим выключателем В.

Содержание отчета

В отчете необходимо:

- указать цель работы;

- привести схему лабораторной установки;
- записать измеренные значения напряжений срабатывания и отпускания каждого магнитного пускателя;
- записать назначение каждого из аппаратов в схеме лабораторной установки;
- устно ответить на все контрольные вопросы.

Контрольные вопросы

1. Рассказать назначение и устройство магнитного пускателя, показав его условное обозначение на электрической схеме
2. Рассказать назначение и устройство теплового реле. Как оно включается в электрических схемах?
3. Рассказать об особенностях условий работы магнитных пускателей.
4. Какими техническими данными характеризуется каждый пускатель?
5. Назовите наиболее распространенные серии пускателей, их отличие по условиям применения.
6. Рассказать, пользуясь схемой лабораторной установки, как осуществляется реверсирование электродвигателя и какие аппараты участвуют при этом?
7. Объяснить, пользуясь схемой, возможен ли пуск электродвигателя при одновременном нажатии кнопок «Вперед» и «Назад»?
8. Показать цепи, по которым протекают токи при пуске электродвигателя «Вперед» и «Назад».
9. Какие виды защиты применены в схеме установки и какими аппаратами они обеспечиваются?
10. Чем объяснить, что напряжение срабатывания контактора магнитного пускателя больше напряжения отпускания?
11. Как вычислить коэффициент возврата контактора?
12. Какие неисправности возможны в схеме установки, вызванные

работой магнитных пускателей?

Указания по технике безопасности

1. Изучение лабораторной установки производить только при обесточенной схеме.
2. Перед включением установки закрыть защитным кожухом магнитные пускатели.
3. При пуске двигателя и его работе не касаться вращающихся частей.
4. Не открывать кожух, закрывающий вал двигателя.
5. После окончания работы обесточить схему автоматическим выключателем.

Лабораторная работа № 2

Исследование теплового реле

Техника безопасности

1. Перед началом проведения работы необходимо ознакомиться с лабораторной установкой, обратив особое внимание на ее токоведущие элементы и наличие заземления корпуса установки.
2. Включение установки, перевод ее с одного режима на другой разрешается только после изучения методики проведения опыта в присутствии преподавателя.
3. В процессе выполнения опыта следует быть внимательным и осторожным, не касаться токоведущих элементов установки.
4. После окончания работы отключить установку от сети автоматическим выключателем.
5. Студент в лаборатории должен быть предельно дисциплинирован, бережно относиться к оборудованию лаборатории, не включать другие лабораторные установки, не относящиеся к данной работе.

Цель и программа работы

Целью настоящей работы является углубление знаний о физических процессах в тепловом реле и его применении для защиты электроустановок. В работе необходимо:

1. Изучить конструкцию и принцип действия теплового реле серии ТРТ и ТРН.
2. Изучить принцип работы «прыгающего» контакта, обеспечивающего быстроту размыкания цепи.
3. На примерах конкретных схем рассмотреть тепловых реле для защиты электроустановок.
4. Снять опытным путем и построить время-токовые характеристики реле.
5. Сравнить экспериментальные время-токовые характеристики с теоретическими и сделать выводы.

Описание экспериментальной установки

В состав для исследования теплового реле, схема которой изображена на рисунке 1, входят:

1. Исследуемое тепловое реле РТ типа ТРП.
2. Автотрансформатор Т1 для регулирования тока нагрузки
3. Контактёр включения нагрузки в сеть К.
4. Автоматический выключатель сети Р.
5. Выключатель цепи управления установки S1.
6. Кнопочная станция управления включением и отключением нагрузки в сеть S2, S3.
7. Электроизмерительные приборы для измерения напряжения сети VI; напряжения V2, снимаемого с автотрансформатора Т2; тока нагрузки А. В качестве нагрузки используется трансформатор Т2, вторичная обмотка которого замкнута накоротко через чувствительный элемент исследуемого теплового реле РТ. Размыкающий контакт теплового реле включен в цепь пита-

ния обмотки контактора К. При срабатывании реле контактор отключает нагрузку от питающей сети.

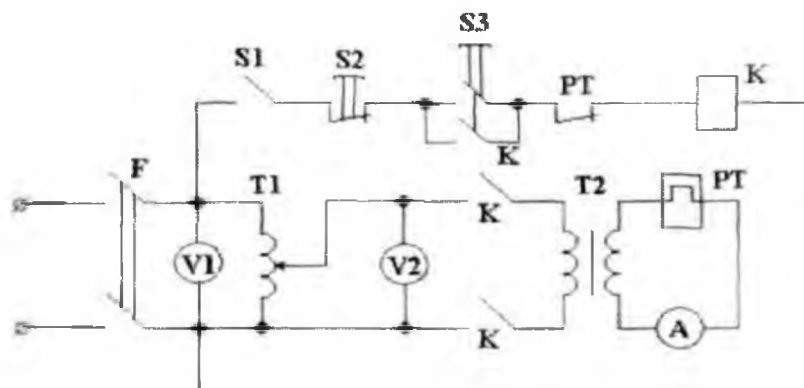


Рисунок 1 Принципиальная схема экспериментальной установки

Краткие теоретические сведения

Надежная и длительная работа электроустановок во многом зависит от перегрузки по току. Протекание тока, большего, чем номинальный, приводит к дополнительному повышению температуры и ускоренному старению изоляции.

Для защиты электродвигателей и различных электроустановок от токовых перегрузок наибольшее распространение получили тепловые реле с биметаллической пластиной, являющейся чувствительным элементом реле.

Биметаллическая пластина состоит из двух пластин, имеющих различные температурные коэффициенты расширения α_1 и α_2 . В качестве материалов пластин широко применяются инвар (малое значение α) и хромоникелевая сталь (большое значение α). Пластины плотно прилегают друг к другу и жестко скреплены. Если один конец такой пластины закрепить неподвижно и нагреть, то произойдет изгиб пластины в сторону материала с меньшим α (рисунок 2). Свободный конец пластины при этом воздействует на контактную систему с силой F , определяемой по формуле (1):

$$F = 3/16 * (\alpha_1 - \alpha_2) * E * \tau * (b * \delta^2) / l \quad (1)$$

где α_1 и α_2 - температурные коэффициенты расширения пластин биметаллического элемента;

δ - суммарная толщина пластины;

b и l - ширина и длина пластины;

E - средний модуль упругости пластины; $E = (E_1 + E_2)/2$;

τ - превышение температуры, равное разности температур тела Θ_2 и окружающей среды Θ_1 ; $\tau = \Theta_2 - \Theta_1$.

Максимальный прогиб пластины $x_{\max}$ равен

$$x_{\max} = 3/4 * (\alpha_1 - \alpha_2) * l^2 * \tau / \delta \quad (2)$$

Из уравнений (1) и (2) видно, что сила и прогиб пластины тем больше, чем больше $(\alpha_1 - \alpha_2)$.

Нагрев биметаллического элемента производится от специального нагревателя, по которому протекает ток нагрузки. Лучшие характеристики получаются при комбинированном нагреве, когда ток нагрузки протекает как непосредственно через биметалл, так и через специальный нагреватель (реле типа ТРП).

Прогиб биметаллической пластины происходит медленно. Если она непосредственно связана с размыкающим контактом, то образуется дуга. Для гашения дуги пластина действует на контакт через ускоряющее устройство, называемое «прыгающим» контактом (рисунок 3).

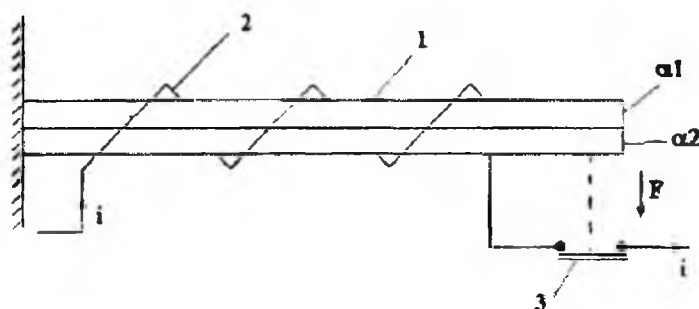


Рисунок 2 Принцип действия теплового реле:

1 - биметаллическая пластина; 2 - нагреватель; 3 - размыкающий контакт.

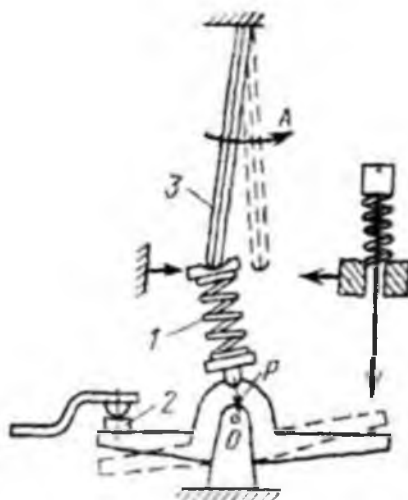


Рисунок 3 Устройство «прыгающего контакта»:

1 - пружина; 2 ось вращения «прыгающего» контакта; 3 - контакт; 4 - биметаллическая пластина; 5 - кнопка возврата.

В обесточенном состоянии пружина 1 создает момент относительно оси вращения 2 «прыгающего» контакта и замыкает контакт 3. Биметаллическая пластина 4 при нагреве изгибается вправо, положение пружины изменяется. Теперь она создает момент в противоположном направлении, обеспечивая мгновенное замыкание контакта 3 и гашение дуги.

Основной характеристикой теплового реле при коротком замыкании является зависимость времени срабатывания от тока нагрузки, называемая время-токовой характеристикой (рисунок 4).

Время срабатывания тепловых реле при коротком замыкании больше, чем время термической стойкости аппарата и объектов защиты. Поэтому тепловые реле не могут быть использованы для защиты от коротких замыканий. Цепь и само реле нужно защищать от коротких замыканий предохранителями.

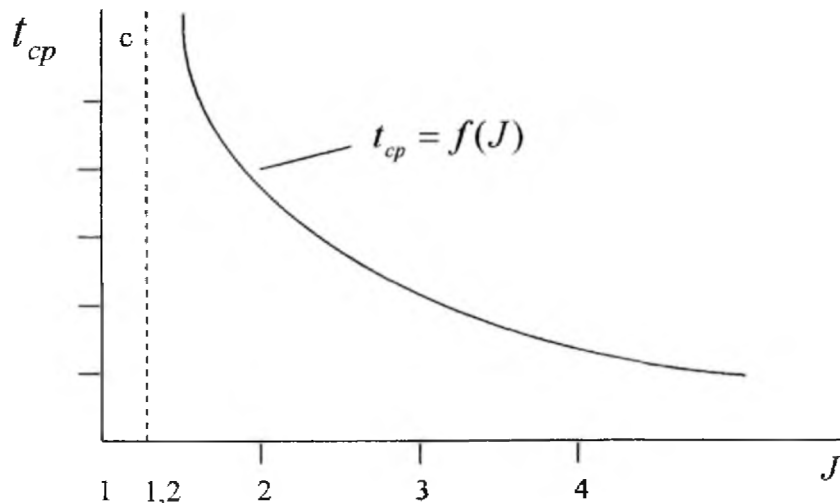


Рисунок 4 Время-токовая характеристика теплового реле

Для защиты электроустановок тепловое реле выбирают по его номинальному току, который должен быть равен номинальному току защищаемого объекта. Срабатывание реле происходит при токе $I_{cp} = (1,2 \div 1,3) * I_H$ примерно через 20 минут. С увеличением тока нагрузки время срабатывания t_{cp} сокращается (рисунок 4).

Нагрев биметаллической пластины зависит от температуры окружающей среды, с ростом которой ток срабатывания уменьшается.

$$J_{cp} = J_{cp.n} \sqrt{\frac{\theta_{cp} - \theta}{\theta_{cp} - \theta_n}},$$

где $I_{cp.n}$ - номинальный ток срабатывания, соответствующий номинальной окружающей температуре θ_n (обычно $+40^\circ \text{C}$);

θ_{cp} - температура срабатывания реле;

θ - фактическая температура окружающей среды.

Для правильной работы тепловой защиты желательно реле располагать в том же помещении, что и защищаемый объект.

Методика проведения эксперимента

Выполнение лабораторной работы следует начинать с изучения

принципа действия и конструкции тепловых реле серии ТРП и ТРН, обратив особое внимание на работу "прыгающего" контакта. Изучение конструкции производить путем частичной разборки реле. Хорошо разобратся в физической сущности время-токовой характеристики, в выборе и настройке реле для защиты электроустановок от перегрузки по току.

Для снятия время-токовой характеристики используется лабораторная установка, схема которой изображена на рисунке 1, и секундомер. Снятие характеристики производится в следующей последовательности:

1. Убедиться, что автоматический выключатель сети F и выключатель цепи управления S1 поставлены в положение "выключено", а ручка регулятора автотрансформатора T1 выведена влево до упора.

2. Автоматическим выключателем F подключить лабораторную установку к сети и выключателем S1 включить цепь питания контактора К.

3. Нажатием кнопки S3 "Пуск" подключить нагрузку к питающей сети.

4. Контролируя по амперметру А, ручкой регулятора автотрансформатора T1 установить ток перегрузки и пустить секундомер. При колебаниях тока необходимо регулятором поддерживать установленное его значение.

Опытные значения токов перегрузки следует брать равными, указанным в таблице.

Таблица Протокол записи результатов измерений

Номер опыта	Значение токов перегрузки, А / Значение номи-	36	38	40	42	44	46	48	50
1	$I_H=19, A$								
2	$I_H=22, A$								
3	$I_H=25, A$								

5 В момент срабатывания теплового реле, что отмечается выключением

контактора К, остановить секундомер и записать значение времени срабатывания t_{cp} в таблицу против величины тока перегрузки I .

6 Как только тепловое реле автоматически вернется в исходное положение, что проявляется легким щелчком «прыгающего» контакта, снова повторить действия, описанные в п.п.5.3-5.5 для следующего значения тока перегрузки.

Всего необходимо снять 3 время-токовые характеристики, задаваясь значениями тока перегрузки, при трех положениях регулировочного поводка теплового реле, соответствующих номинальным токам $I_H = 19A$, $I_n = 22A$ и $I_n = 25A$. Полученные данные записать в таблицу и по результатам построить время-токовые характеристики $t_{cp} = f(I)$ для трех значений настройки реле.

7 Сравнить опытные данные с теоретическими и сделать выводы по результатам исследований.

Отчет о выполненной работе

Отчет должен содержать:

1. Цель, программу исследований и краткое описание работы.
2. Принципиальную схему лабораторной установки.
3. Таблицу с результатами измерений.
4. Построенные на одном графике время-токовые характеристики по результатам опытов.
5. Выводы по работе.

Вопросы для самопроверки

- 1 Почему необходимо защищать электроустановки от длительно протекающих токов, превышающих номинальный?
- 2 Как устроен чувствительный элемент теплового реле?
- 3 Устройство «прыгающего» контакта и его назначение.
- 4 В какую цепь электроустановки включается чувствительный эле-

мент реле и в какую цепь разрывающие контакты?

5 Почему тепловое реле не может применяться для защиты электроустановок от короткого замыкания?

6 От каких факторов зависит усилие изгиба биметаллической пластины при ее нагреве?

7 Какие факторы влияют на величину тока срабатывания теплового реле?

8 Что представляет собой время-токовая характеристика теплового реле?

9 Как устранить влияние температуры окружающей среды на ток срабатывания реле?

10 Как правильно выбрать тепловое реле для защиты электроустановки от перегрузки по току?

Лабораторная работа № 3

Исследование электрических аппаратов в схеме управления многосекционным конвейером

Общие сведения о применении электрических аппаратов в схеме управления механизмов непрерывного транспорта

Целью исследования электрических аппаратов в схеме управления многосекционным конвейером является выявление возможных неисправностей блокировочных цепей, цепей пуска и последовательности включения и отключения при работе в автоматическом режиме. Согласованная и исправная работа электрических аппаратов в схеме управления должна осуществлять:

- включение и отключение конвейеров в определенной последовательности;
- обеспечение требуемой скорости транспортировки грузов и при необходимости согласование скорости различных конвейеров;
- контроль исправного состояния тягового элемента (ленты, канаты, цепи) и отключение установки при обрыве или слабом натяжении ленты или

сходе ее с опорных роликов, приводных барабанов;

- предупреждение завалов бункеров загрузочных и перегрузочных устройств;

- отключение установки при затянувшемся пуске.

Чтобы избежать завалов перегрузочных устройств в многосекционном ленточном конвейере применяются электрические блокировочные связи между двигателями совместных секций. Это может быть осуществлено как через блок-контакты, так и через главные (силовые) контакты контакторов, управляющих двигателями.

На рисунке 1 показана схема управления двигателями многосекционного конвейера, перемещающего насыпные грузы из правого бункера в левый. Электропривод конвейеров осуществляется от асинхронных короткозамкнутых двигателей М1, М2, М3. Во избежание завалов грузов пуск конвейеров начинается с включения двигателя М1 нажатием кнопки КнШ. Контакт К1, срабатывая, своими главными контактами подает питание на двигатель М1. Затем последовательно, нажатием кнопок КнП2 и КнП3, включающих контакторы К2 и К3, пускаются двигатели М2 и М3. Пуск двигателя М2 раньше пуска М1 невозможен, так как в цепи катушки контактора К2 находится замыкающий блок-контакт контактора К1. Аналогично заблокированы двигатели М2 и М3.

Общая остановка всех конвейеров производится нажатием кнопки КнС1. Отключаясь, контактор К1 своими блок-контактами разрывает цепи питания катушек К2 и К3. При необходимости остановить только конвейер 3 следует нажать кнопку КнС3. Двигатель М3 отключается, а М1 и М2 продолжают работать.

Схема управления обеспечивает защиту: от токов к.з. в силовой цепи и в цепях управления предохранителями Пр1, Пр2, Пр3 и от длительных перегрузок двигателей тепловыми реле РТ1, РТ2, РТ3; сигнализацию, указывающую какой из двигателей находится в рабочем состоянии, какой в отключенном (горит красная лампа ЛК - двигатель работает, зеленая ЛЗ - отключен).

В схеме не предусмотрен контроль за состоянием ленты и другими повреждениями в конвейерах.

Схема включения конвейера может быть реализована на программируемом реле. На рисунке 4 показано подключение ко входам программируемого реле: S0 – аварийная остановка, S1 – кнопка «Пуск», S2 – кнопка «Стоп», S3 – кнопка «Быстрый останов», QF1 – QF3 – вспомогательные контакты автоматических выключателей двигателей M1 – M3, KM1 – KM3 – контакторы включения двигателей M1 – M3, H1 – лампочка индикации, SF1 – автоматический выключатель.

Цель и программа работы

1 Целью работы является приобретение начальных навыков в исследовании и нахождении характерных неисправностей электрических аппаратов в типовых системах автоматизированного управления многосекционными ленточными конвейерами.

2 Программа работы:

- ознакомление с защитными блокировками, применяемыми в схемах автоматизации конвейерных установок;
- изучение схемы узла контроля пуска конвейера и управления пуском многосекционного ленточного конвейера;
- изучение схемы защиты перегрузочных бункеров от завалов с помощью электродного датчика уровня загрузки бункера;
- исследование признаков неисправностей в системе управления многосекционным конвейером и установление возможных причин их возникновения.

Описание установки

- Лабораторная работа выполняется на установке, схема которой приведена на рисунке 3. Она представляет собой автоматизированную схему управления электроприводами многосекционного конвейера, аналогичную описан-

ной ранее (рисунок 1). Монтаж установки выполнен на стенде, на котором размещены электродвигатели М1, М2, М3; кнопочные станции управления тремя электроприводами КнШ и КнС1, КнП2 и КнС2, КнП3 и КнС3; сигнальные лампочки Л1, Л2, Л3, указывающие рабочее состояние двигателей (лампочка горит - двигатель включен); автоматический выключатель сети В1; выключатели В2 и В3, предназначенные для проведения наладочных операций в схеме управления и позволяющие независимое включение двигателей М2 и М3. Верхнее положение этих выключателей соответствует замкнутому состоянию цепей, показанных на рис.3; вольтметр для контроля напряжения сети.

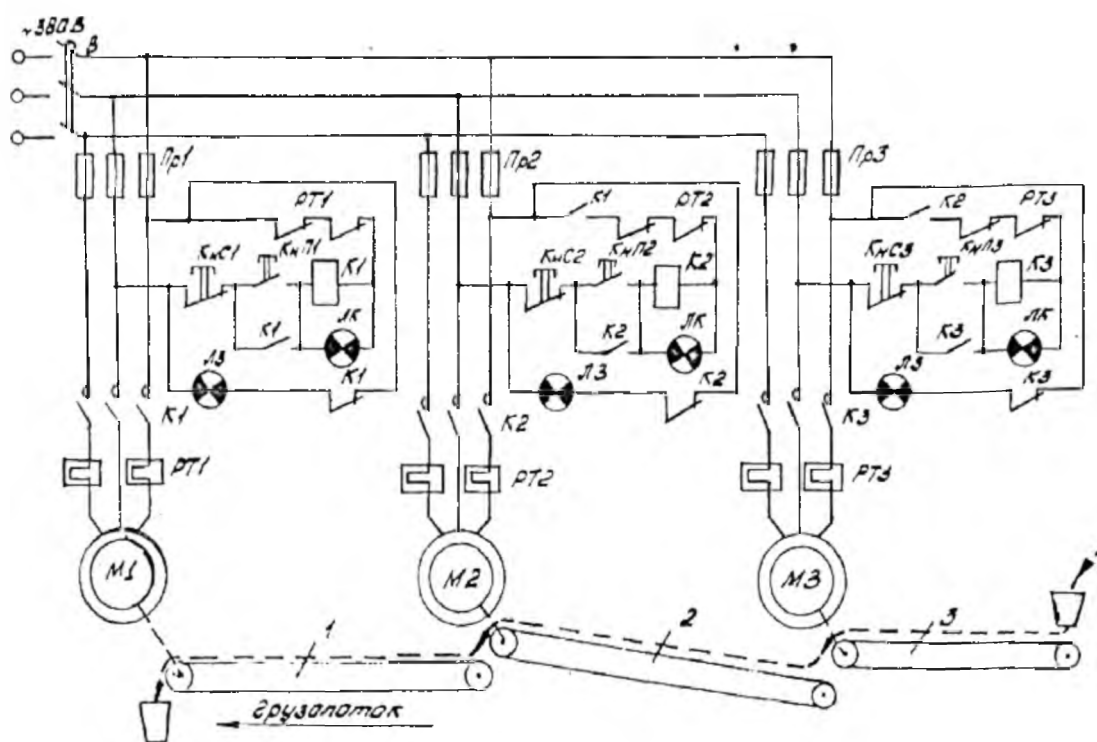


Рисунок 1 Схема управления многосекционным конвейером

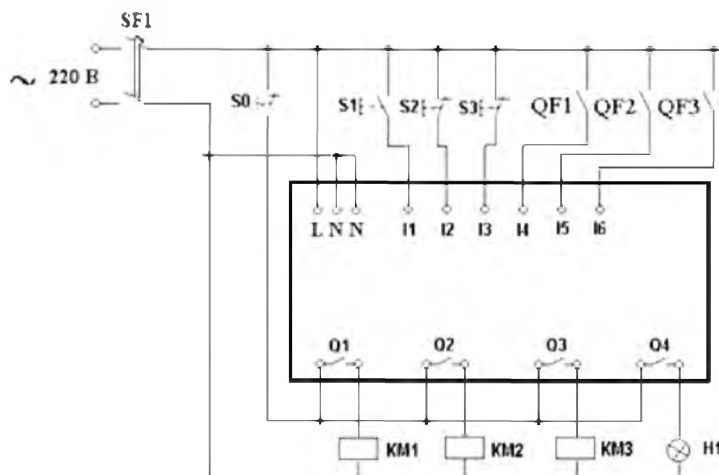


Рисунок 2 Цепи управления программируемого реле

В качестве электродвигателей использованы маломощные трехфазные асинхронные двигатели с короткозамкнутым ротором серии АВ - 042. Для наглядного представления работы двигателя при демонстрации опытов на вал насажен легкий барабан, имеющий красные полосы.

Для имитации характерных неисправностей, возникающих в системе управления, на стенде расположены три выключателя, обозначенные В4, В5, В6, Однако, на схеме (рисунок 3) места их выключений не показаны. В процессе выполнения лабораторной работы студентам предлагается установить возможные причины неисправностей по признакам их проявления при включении установки.

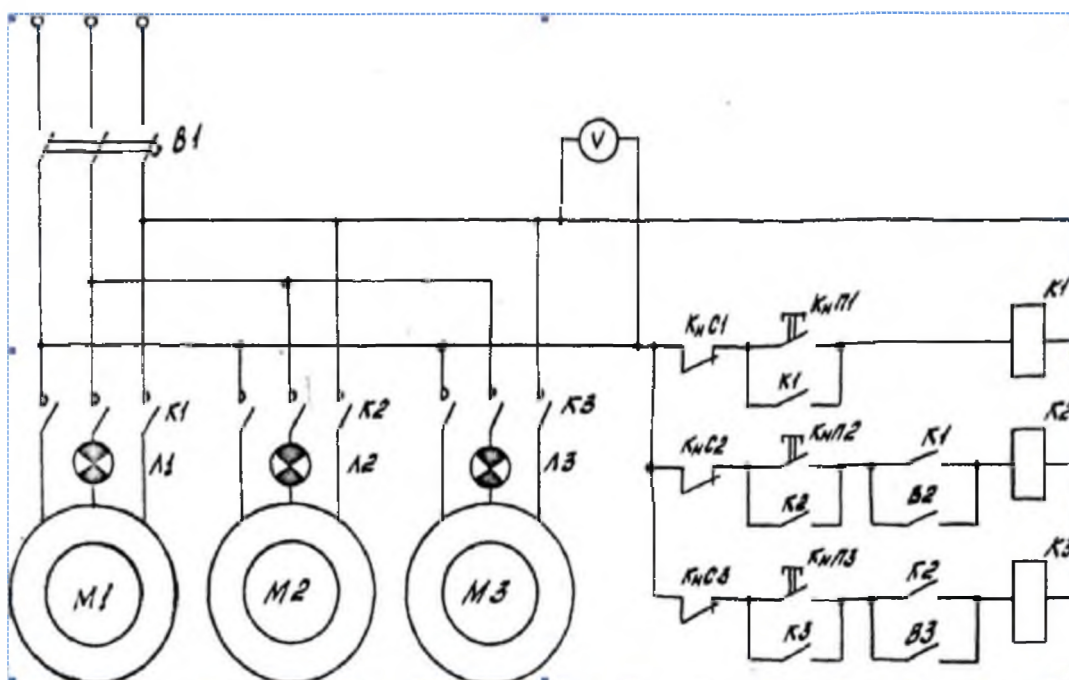


Рисунок 3 Схема лабораторной установки

Методические указания

Прежде чем приступить к выполнению работы необходимо изучить схемы автоматизированного управления электроприводами конвейеров (рис.1, рис.2, рис.3). Разобраться в назначении и работе блокировочных связей, последовательности включения отдельных элементов схемы, обеспечивающих нормальное, функционирование системы, также твердо знать принцип действия и устройство электрических аппаратов, примененных в установке. Изучить лабораторную установку, пользуясь схемой рис.3 и монтажной схемой на лицевой панели стенда.

С разрешения преподавателя включить установку. Для чего автоматический выключатель В1 перевести в положение "Вкл". и убедиться в наличии напряжения, записать его значение. Выключатели В2 и В3 наладочных работ и выключатели В4, В5 и В6 установить в положение "Выкл." Затем опробовать схему, осуществляя пуск и остановку электродвигателей привода в последовательности, указанной ранее для многосекционного конвейера (рисунок 3). При опробовании схемы следует сделать попытки

осуществить неправильную последовательность работы ее элементов. Например, попытаться включить электродвигатель М2 или М3 при отключенном электродвигателе М1. Убедиться также в возможности осуществления наладочных операций на конвейерах 2 и 3 при неработающем конвейере 1, для чего включить выключатели В2 и В3 и произвести пуск двигателей М2иМ3.

После опробования схемы приступить к исследованию возможных неисправностей и заполнить заготовленную таблицу по форме:

Обозначение выключателя на схеме, имитирующего неисправность	Признаки неисправности	Возможные причины неисправности и порядок ее устранения
В4 В5 В6 В2 В3	Напряжение питания =..... В.	

Последовательно устанавливая выключатели В4, В5, В6, имитирующие неисправность в положение "Вкл.", снова опробовать схему. Заметить и записать проявляющиеся при этом признаки неисправности в работе установки и, пользуясь схемой, указать возможные причины неполадок и пути их устранения. Для поиска причин неисправности разрешается производить повторное включение схемы в различной последовательности, если в этом есть необходимость. Приступать к исследованию неисправности необходимо только после установления причин и устранения данной неисправности, о чем поставить в известность преподавателя.

Затем, устанавливая поочередно выключатели В2 и В3 в положение «Вкл.», опробовать работу схемы и, исследуя в такой же последовательности, указать какие неисправности сопровождаются аналогичными признаками. Результаты исследований записать в таблицу и по окончании работы показать преподавателю. Выключить установку.

Содержание отчета

1. Схема установки.
2. Перечень оборудования.
3. Таблица с результатами исследований.

Контрольные вопросы

1. Для каких целей применяется автоматизация схем управления механизмами непрерывного транспорта?
2. Рассказать, как осуществляется защита перегрузочных бункеров от завалов.
3. Объяснить, пользуясь схемой, работу узла контроля пуска.
4. Какая должна быть последовательность пуска и остановки приводов многосекционного конвейера?
5. Пользуясь схемой, рассказать, как осуществляется блокировка для обеспечения определенной последовательности пуска и остановки приводов многосекционного конвейера.
6. Как производятся наладочные операции в схеме управления конвейерной установкой?
7. Как проявится неисправность при «подгорании» блок-контактов К1 в цепи управления двигателем М1 и М2 (рисунок 3)?
8. Как проявится неисправность при «подгорании» блок-контакта К2 (рисунок 3)?
9. Как проявится неисправность при «залипании» блок-контакта К1 в цепи управления двигателем М1? То же самое в цепи управления двигателем М2?
10. Какую неисправность вызовет «залипание» блок-контакта К2?

Указания по технике безопасности

1. Изучение лабораторной установки производить только при обесточенной схеме.

2. Запрещается садиться, опираться на установку, если даже она не включена.

3. Включение установки производить только в присутствии и с разрешения преподавателя.

4. Запрещается открывать доступ к внутреннему монтажу установки.

5. После окончания работ установка должна быть обесточена.

Лабораторная работа № 4

«Исследование электрических аппаратов в блокировочных цепях станка»

1 Цель и программа работы

1.1 Целью работы является приобретение навыков в исследовании и нахождении характерных неисправностей электрических аппаратов в типовых системах взаимосвязи между отдельными механизмами станка.

1.2 Программа работы:

- изучение схем взаимосвязи наладочного и рабочего режимов станка;
- изучение схемы ограничения перемещений механизма станка;
- изучение схемы согласования работы главного привода и привода подачи станка;
- исследование признаков неисправностей в схеме согласования работы и установление возможных причин их возникновения.

2 Общие сведения о типовых взаимосвязях в схемах управления станками

В схемах автоматизированного управления станками встречаются некоторые типовые взаимосвязи, призванные осуществлять следующие режимы.

2.1 Наладочный и рабочий режимы станка.

В рабочем режиме привод станка работает длительно или повторно-кратковременно. Наладочные операции производятся для опробования отдельных узлов станка, для проверки правильности установки заготовки и инструмента. Этот режим характеризуется кратковременными включениями ненагруженного привода при малых оборотах двигателя. На рисунке 1 приведена схема согласования наладочного и рабочего режимов привода. Для длительного режима нажимается кнопка КНП. При этом получает питание контактор К, который главными контактами включает двигатель М и блок-контактом блокирует кнопку КНП.

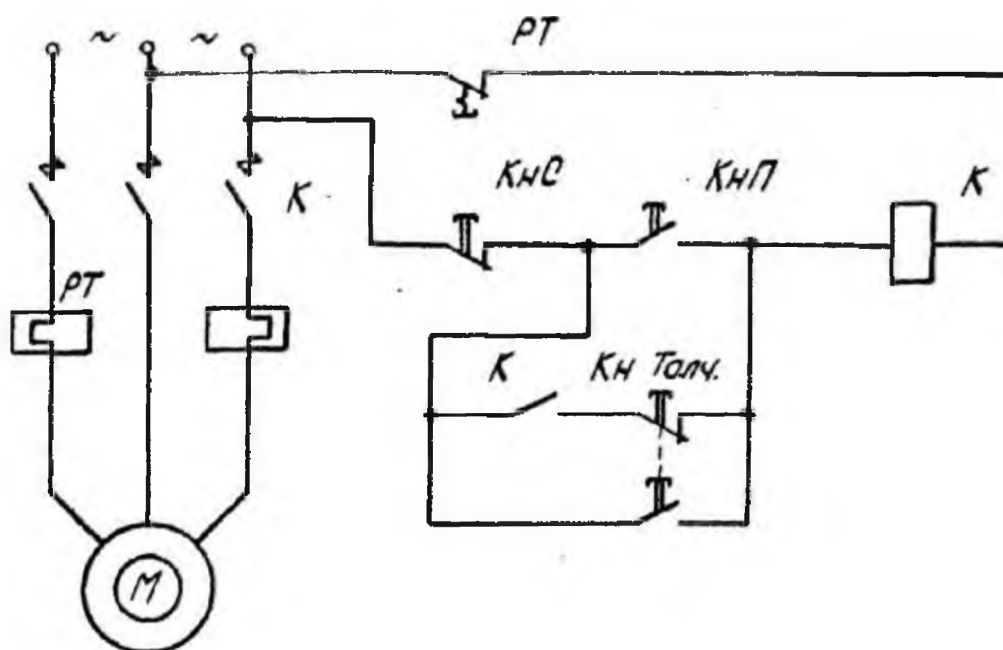


Рисунок 1 - Схема взаимосвязи наладочного и рабочего режимов станка

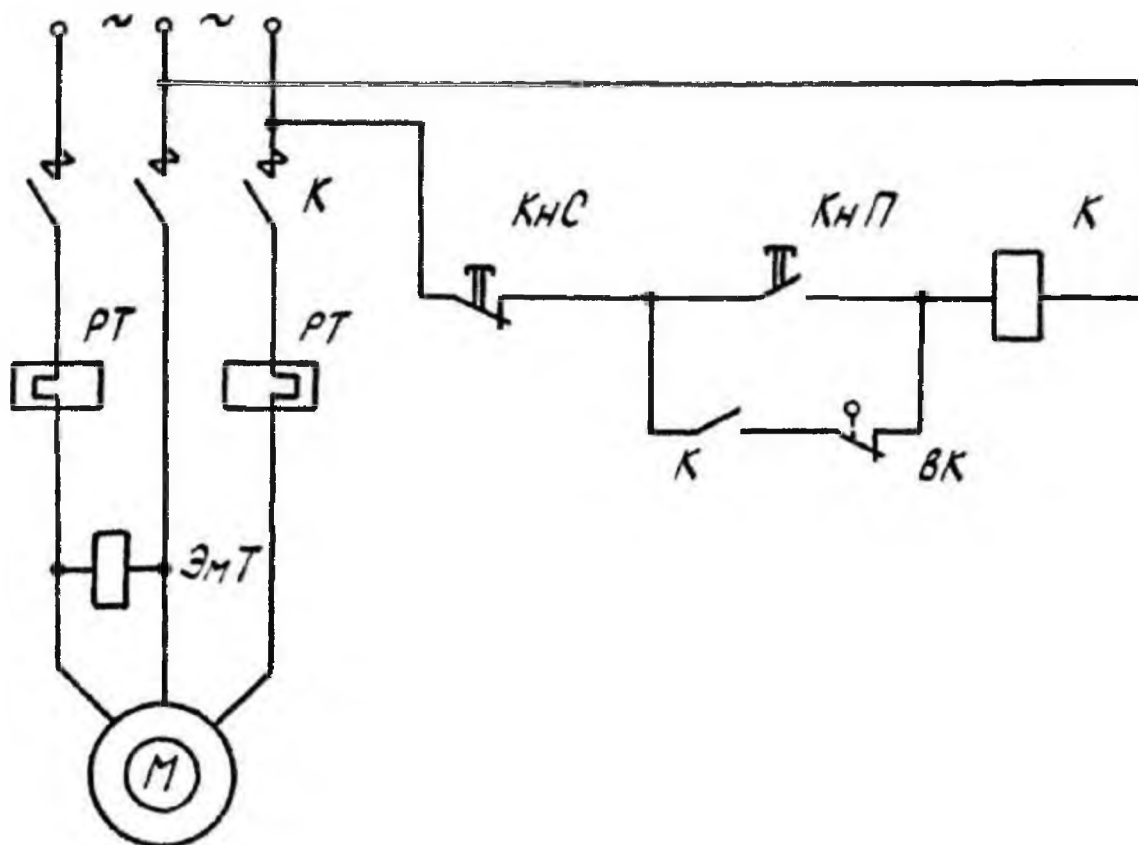


Рисунок 2 Схема отключения двигателя при ограничении перемещения механизма

2.2 Ограничение перемещений и точная остановка механизмов станка применяется для исключения столкновения между отдельными подвижными элементами или для предупреждения выхода узлов станка из нормального зацепления с ведущим звеном кинематической цепи.

2.3 Согласование работы отдельных приводов

Первоочередность включения главного привода здесь обеспечивается введением в цепь катушки контактора КП замыкающего контакта контактора КГ. При неработающем приводе подачи контактор главного привода КГ отключается без выдержки времени после нажатия кнопки КНС 1.

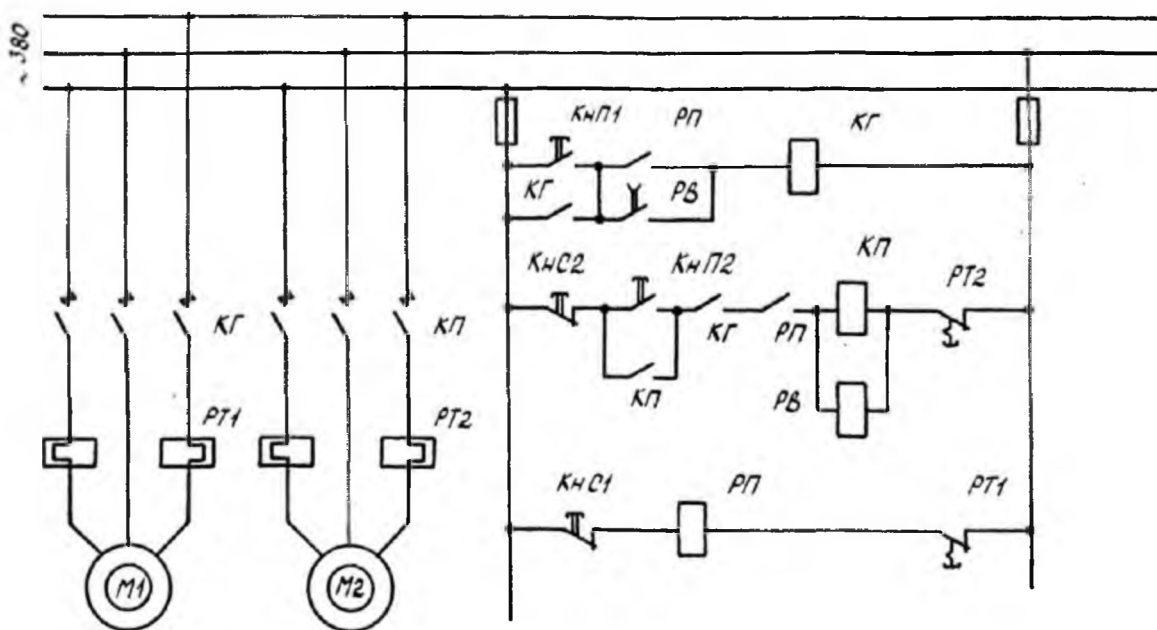


Рисунок 3 Схема согласования работы главного привода и привода

Для отключения подачи станка главного привода при работающем приводе подачи следует длительно нажать на кнопку КнС 1. При этом теряет питание промежуточное реле РП, обеспечивается контактор КП и отключается двигатель подачи М2. Отключение главного привода М2 произойдет через некоторое время, обусловленное установкой реле времени РВ, катушка которого подключена параллельно катушке контактора КП. При кратковременном нажатии на кнопку КнС1 вновь включится реле РП, и если к этому моменту реле РВ не сработало, то главный привод не отключится после отключения привода подачи.

3 Описание установки

Лабораторная установка, схема которой показана на рисунке 4, представляет собой действующий макет автоматизированной системы, обеспечивающей согласованную работу главного привода и привода подачи металло-режущего станка. По выполняемым функциям она аналогична схеме, приве-

денной на рисунке 3.

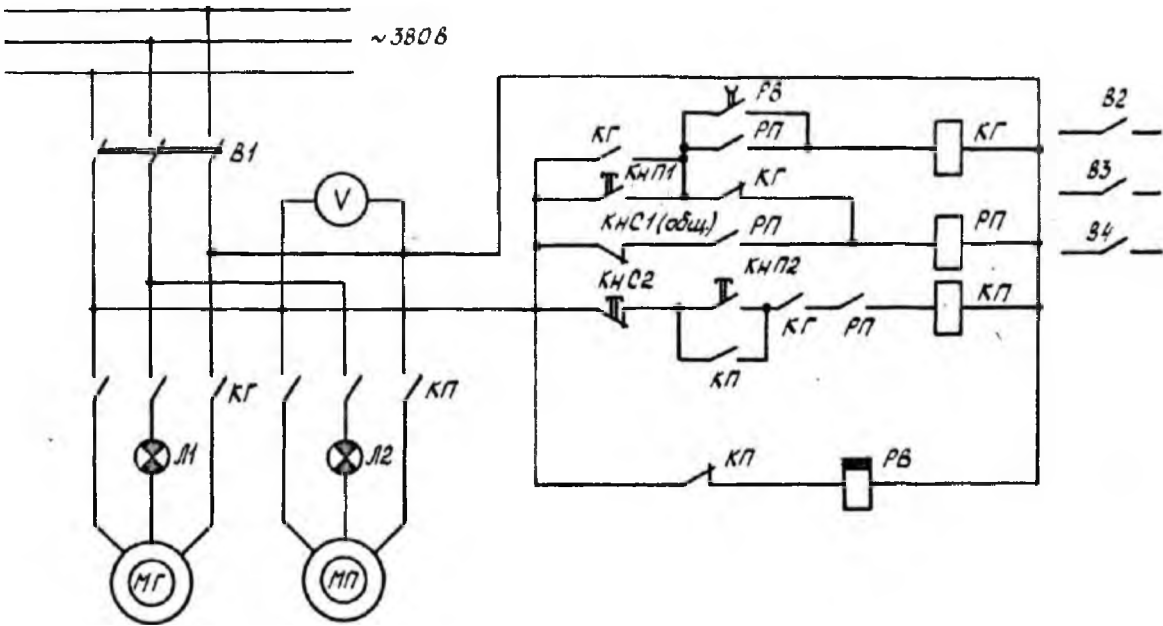


Рисунок 4 - Схема лабораторной установки

4 Методические указания

Пользуясь ее схемой, указать возможные причины и пути устранения неисправности. Результаты исследований записать в таблицу 2.

Таблица 2 Результаты исследований

Обозначение выключа - теля на схеме, имити- рующего неисправность	Признаки неисправности	Возможные причины неисправности и порядок ее устранения
B2 B3 B4	Напряжение пи- тока =... В	

Приступать к исследованию следующей неисправности следует только после установления и устранения причин данной неисправности.

После окончания исследований обесточить установку выключателем

В1 и показать результаты преподавателю.

5 Содержание отчета

5.1 Схема установки.

5.2 Таблица с результатами исследований.

5.3 Перечень оборудования.

6 Контрольные вопросы

6.1 Рассказать, пользуясь схемой, как осуществляется работа привода станка в наладочном режиме.

6.2 Рассказать работу схемы, обеспечивающей ограничение перемещений и точную остановку механизмов станка.

6.3 Как осуществляется согласованная работа отдельных приводов станка и для чего это необходимо?

6.4 Как проявится неисправность при «подгорании» контактов реле времени РВ на рисунке 4?

6.5 Какими признаками неисправной работы схемы (рисунок 4) сопровождается «подгорание» контактов промежуточного реле РП в цепи катушки КГ? В цепи катушки РП? В цепи катушки КП?

7 Указания по технике безопасности

7.1 Изучение лабораторной установки производить только при обесточенной схеме.

7.2 Запрещается садиться, опираться на установку, если она даже не включена.

7.3 Включение установки производить только в присутствии и с разрешения преподавателя.

7.4 Запрещается открывать доступ к внутреннему монтажу установки.

7.5 Не допускать соприкосновения с вращающимися частями установки.

7.6 После окончания работ установку обесточить.

Лабораторная работа № 5

«Исследование устройства защитного отключения»

Общие сведения об устройствах защитного отключения (УЗО)

УЗО предназначено для защиты человека от поражения электрическим током в групповых линиях, питающих розетки.

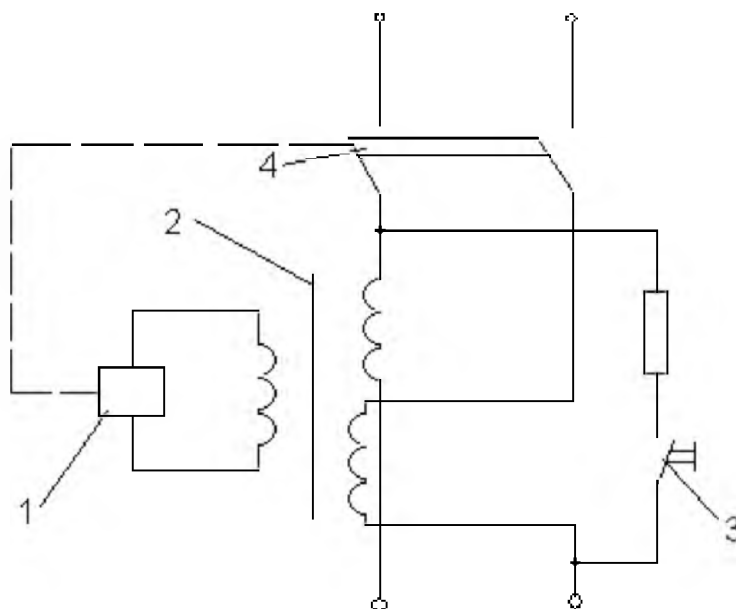


Рисунок 5 - Устройство защитного отключения

1 Поляризованное реле

2 Трансформатор

3 Тестовая кнопка

4 Контактная группа

Ток, от которого происходит защита, может быть вызван прикосновением к токоведущим частям электрооборудования или частям, которые могут

быть под напряжением, например металлический корпус прибора. УЗО обеспечивает защиту при непосредственном контакте человека с оборудованием или проводником, кроме этого УЗО предупреждает опасность возникновения пожара при замыкании электропроводки.

Цель и программа работы

Целью работы является изучение устройства защиты отключения в схеме защиты.

Программа работы включает:

4. Ознакомление с конструкцией устройства защиты отключения.
5. Изучение схемы включения устройства защиты отключения.
6. Определение параметров устройства защиты отключения, при котором происходит размыкание контактов УЗО.

Метод испытаний

1.1 Предварительные работы по проведению испытания УЗО проводятся в следующей последовательности:

- ознакомление с проектной документацией и другими электрическими схемами, в которые включены устройства защитного отключения;
- ознакомление с паспортом и инструкцией по эксплуатации УЗО;
- визуальный осмотр правильности подключения УЗО в электрическую схему потребителя;
- визуальный осмотр на предмет целостности УЗО, надежности крепления выводов.

1.2 Величина сопротивления изоляции УЗО должна соответствовать паспортным данным.

1.3 Измерение времени отключения (t_A).

Чтобы измерить время отключения, должны быть выполнены следующие

щие операции:

1. L, N и PE провода должны быть подключены к сети, как показано на рис.1 (подключение нулевого провода не обязательно).

2. Установить поворотный переключатель 8 (см. приложение А, рисунок 2) к требуемой измерительной функции tA, UB.

3. Задать, является ли выключатель селективным, для чего нажать соответственно на кнопку 12.

4. Установить безопасный уровень напряжения при помощи кнопки 11.

5. Установить величину диапазона отключающего дифференциального тока при помощи кнопки 10.

Если величина напряжения прикосновения, измеряемая при помощи тестового тока, равного $40\% I_N$, и вычисленного для заданного диапазона отключающего дифференциального тока, будет превышать предварительно установленную величину безопасного напряжения UL измерение времени отключения автоматически тормозится и символ 28 появляется на дисплее.

В случае, если УЗО не будет реагировать, на дисплее появится изображение символа 21. Отсутствие реакции УЗО может быть вызвано неправильным выбором диапазона величины отключающего дифференциального тока I_N , поломкой УЗО или его неправильной работой.

1.4 Измерение отключающего тока (IA).

Чтобы измерить отключающий ток следует проделать следующие операции:

1 L, N и PE провода должны быть подключены к сети, как показано на рис.1 (подсоединение нулевого провода не обязательно).

2 Установить поворотный переключатель 8 к требуемой измерительной функции IA, RE.

3 Задать, является ли УЗО селективным, и нажать соответственно на кнопку 12.

4 Установить безопасный уровень напряжения при помощи кнопки 11.

5 Установить предел отключающего дифференциального тока при по-

мощи кнопки 10.

6 Установить начальную фазу дифференциального тока при помощи кнопки 9.

7 Начать измерение при помощи кнопки 7.

После срабатывания УЗО действительная величина тока срабатывания будет показана на дисплее 16. Показ результата измерения сопровождается показом на дополнительном поле дисплея 17 предварительно установленного для заданного типа УЗО величина диапазона дифференциального тока.

Отсутствие реакции УЗО может быть вызвано неправильным выбором диапазона величины отключающего дифференциального тока $I_{\Delta N}$, дефектом УЗО или его неправильной настройкой.

Если величина напряжения прикосновения, измеряемая при помощи тестового тока, равного $40\% I_N$, и вычисленного для заданного диапазона отключающего дифференциального тока, будет превышать предварительно установленную величину безопасного напряжения U_L измерение времени отключения автоматически тормозится.

Полученные данные записать в таблицу.

Таблица 3 Результаты измерения

Тип УЗО	Основные параметры УЗО					Ток срабатывания,			Время срабатывания,		
	Число полюсов	Номинал, откл. дифф.	Время срабат.	U ном В	I ном А						
						0,8	Uh	1,	0,8	Uh	1,

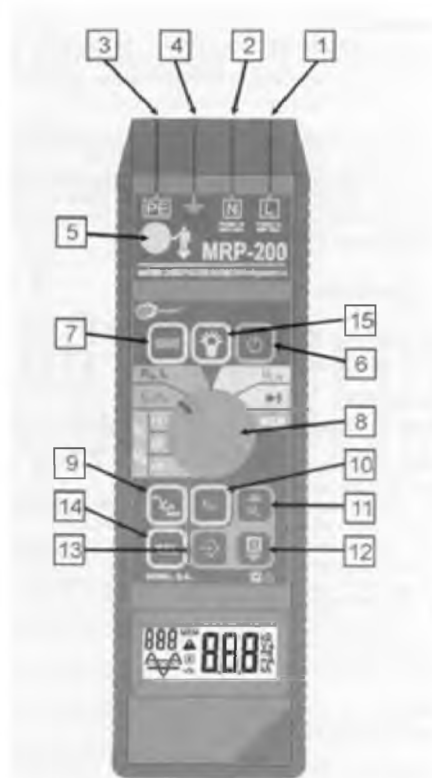
Содержание отчета

В отчете необходимо:

- указать цель работы;
- привести схему УЗО;
- записать измеренные значения параметров устройства защитного отключения;

– устно ответить на все контрольные вопросы.

Панель управления



Контрольные вопросы

13. Рассказать назначение и устройство устройства защитного отключения, показав его условное обозначение на электрической схеме.

14. Как устройство защитного отключения включается в электрических схемах?

15. Рассказать об особенностях условий работы устройства защитного отключения.

16. Какими номинальными параметрами характеризуется устройство защитного отключения?

17. Назовите наиболее распространенные серии устройства защитного отключения, их отличие по условиям применения.

Контроллер ОВЕН ПЛК

Контроллер ОВЕН ПЛК150 соответствует требованиям ГОСТ Р 51841-2001. Контроллер ПЛК150 выпускается в различных модификациях, отличающихся типом установленных в него дискретных выходных элементов, напряжением питания и различными лицензионными ограничениями размера памяти ввода/вывода программы ПЛК.

Условное обозначение контроллера:



Напряжение питания:

220 В переменного тока

24 В постоянного тока;

Тип встроенных аналоговых выходных элементов:

И – цифроаналоговый преобразователь «параметр – ток 4...20 мА»

У – цифроаналоговый преобразователь «параметр – напряжение 0...10 В»

А – цифроаналоговый преобразователь «параметр – ток 4...20 мА или напряжение 0...10 В»

Размер лицензионного ограничения на область ввода-вывода:

L – ограничение в 360 байт

M – без ограничения

Примеры условного обозначения контроллера при заказе:

ПЛК150-24.И-L – контроллер с номинальным напряжением питания 24 В постоянного тока, оснащенный встроенными цифроаналоговыми преобра-

зователями «параметр – ток 4...20 мА» и имеющий лицензионное ограничение на размер области ввода-вывода в 360 байт.

ПЛК150-220.А-М – контроллер с номинальным напряжением питания 220 В переменного тока, оснащенный встроенными цифроаналоговыми преобразователями «параметр – ток 4...20 мА или напряжение 0...10 В» и не имеющий лицензионного ограничения на размер области ввода-вывода.

1. Используемые термины и сокращения

ПЛК – программируемый логический контроллер.

CODESYS – специализированная среда программирования логических контроллеров. Торговая марка компании 3S-Software.

Modbus – открытый протокол обмена по сети RS-485. Разработан компанией Mod- iCon, в настоящий момент поддерживается независимой организацией Modbus-IDA (www.modbus.org).

Modbus-TCP – версия протокола Modbus, адаптированная к работе в сети TCP/IP. DCON – открытый протокол обмена по сети RS-485. Разработан компанией Advantech, применяется в модулях ввода/вывода Adam, модулях компании IPC DAS и некоторых других.

Retain-переменные – переменные пользовательской программы, значение которых сохраняется при выключении питания контроллера или при его перезагрузке. Target-файл – файл или набор файлов, поставляемых производителем, содержащие информацию о ресурсах контроллера, количестве входов и выходов, интерфейсах и т. д. Инсталлируются в систему CODESYS для сообщения ей данной информации.

2. Назначение

Программируемый логический контроллер ОВЕН ПЛК150 предназначен для создания систем автоматизированного управления технологическим оборудованием в различных областях промышленности, жилищно-коммунального и сельского хозяйства.

Логика работы ПЛК150 определяется потребителем в процессе программирования контроллера. Программирование осуществляется с помощью системы программирования CODESYS 2.31.

Контроллер ОВЕН ПЛК150 выпускается в корпусе, предназначенном для крепления на DIN-рейке 35 мм. Подключение всех внешних связей осуществляется через разъемные соединения, расположенные по двум боковым и передней (лицевой) сторонам контроллера. Открытие корпуса для подключения внешних связей не требуется.

Схематический внешний вид контроллера показан на рис. 1.

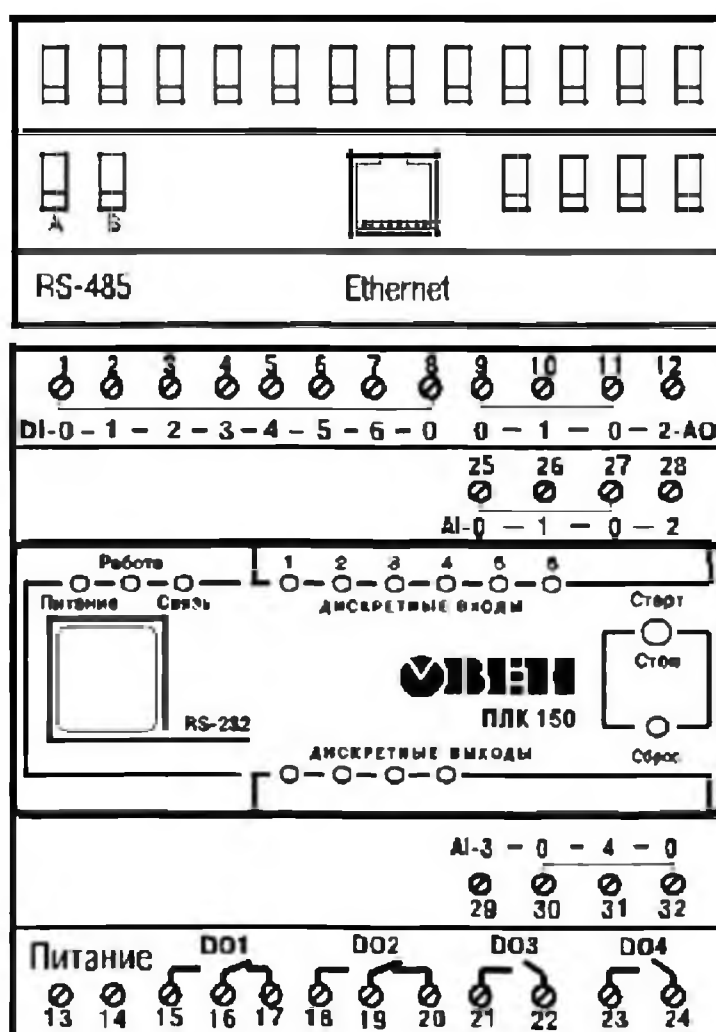


Рисунок 1 – Внешний вид ПЛК150

На боковой стороне расположены разъемы интерфейсов Ethernet и RS-485.

На лицевой панели расположен порт Debug RS-232, предназначенный для связи со средой программирования, загрузки программы и отладки. Подключение к этому порту осуществляется кабелем, входящим в комплект поставки. Также порт Debug RS-232 может быть использован для подключения устройств, работающих по протоколам Modbus, OBEH и DCON.

По обеим боковым сторонам контроллера расположены клеммы для подключения дискретных датчиков и исполнительных механизмов. На переднюю панель контроллера выведена светодиодная индикация о состоянии дискретных входов и выходов, о наличии питания и о наличии связи со средой программирования CODESYS. Также на передней панели имеются две кнопки: кнопка, предназначенная для запуска и остановки программы в контроллере и скрытая кнопка, предназначенная для перезагрузки контроллера. Нажать кнопку возможно только тонким заостренным предметом. В корпусе контроллера расположен маломощный звуковой излучатель, управляемый из пользовательской программы как дополнительный дискретный выход. Звуковой излучатель может быть использован для функций аварийной или иной сигнализации или для отладочных нужд. Частота звукового сигнала излучателя фиксированная и не поддается настройке.

Контроллер ПЛК150 оснащен встроенными часами реального времени, имеющими собственный аккумуляторный источник питания. Энергии полностью заряженного аккумулятора хватает на непрерывную работу часов реального времени в течение 6 месяцев (при температуре 15-35°C). В случае износа аккумулятора, не полной его зарядки, а также при работе при более низких температурах время работы часов реального времени может сократиться.

Аккумулятор, используемый для питания часов реального времени, дополнительно используется как источник аварийного питания микропроцессора контроллера. При случайном отключении основного питания контроллер переходит на аварийное питание и сохраняет промежуточные результаты вычислений и работоспособность интерфейсов Ethernet в течение 10 минут. Свето-

диодная индикация и выходные элементы контроллера при этом не запитываются и не функционируют. При включении основного питания во время работы на аварийном питании контроллер сразу приступает к выполнению пользовательской программы, не тратя время на загрузку и сохраняя все промежуточные результаты вычислений. После 10 мин. работы на аварийном питании контроллер записывает Retain-переменные в энергонезависимую память и отключается. Часы реального времени остаются в рабочем состоянии. После включения основного питания контроллер загружается и запускает программу пользователя (если установлена опция автозапуска). Время работы от аварийного источника питания может быть автоматически скорректировано самим контроллером в зависимости от степени зарядки аккумулятора и температуры окружающей среды.

Во время загрузки контроллера его выходы переводятся в заранее заданное «безопасное состояние», в которых находятся до полной загрузки контроллера и запуска пользовательской программы.

«Безопасное состояние» – это состояние выходов контроллера, при котором подключенные к ним исполнительные механизмы находятся в состоянии, наиболее безопасном для объекта управления, не приводящим к его поломке. Значение «безопасного состояния» выходов задается при конфигурировании области ввода/вывода в PLC-Configuration.

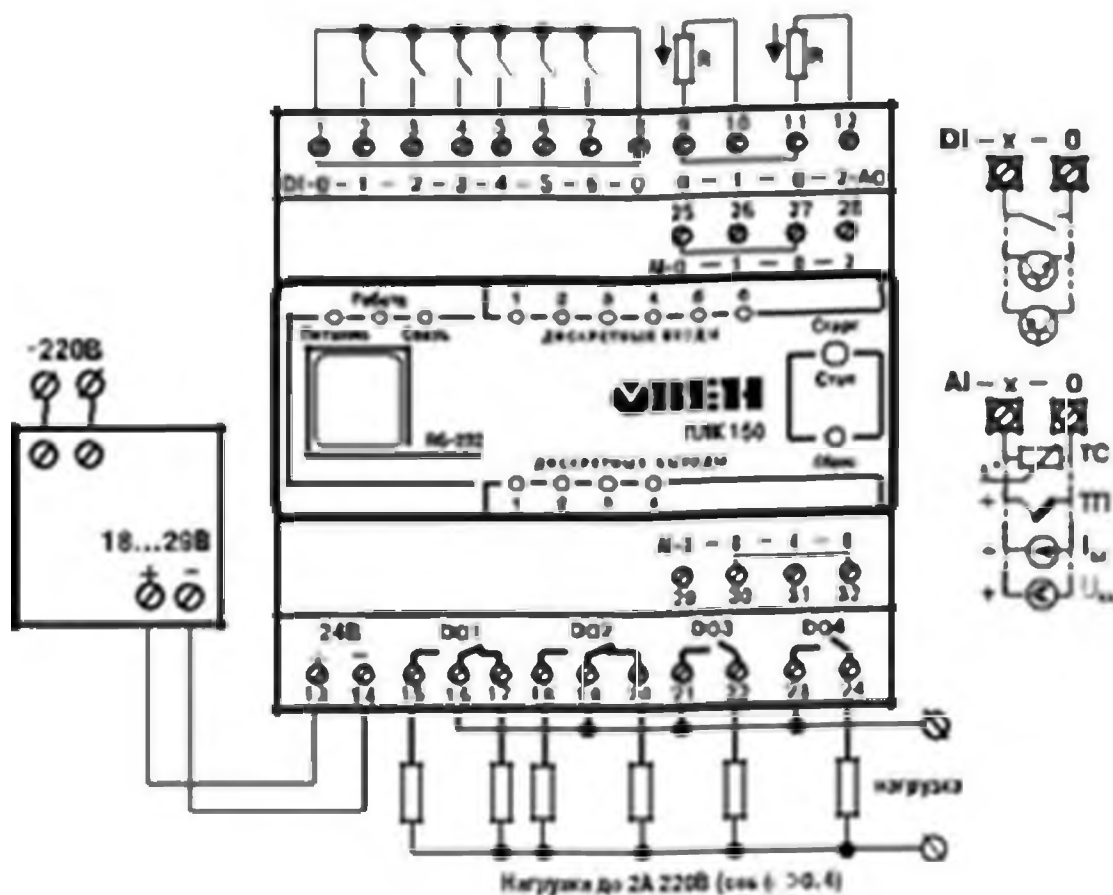


Рисунок 2 – Схема подключения питания, входов и выходов к ПЛК150-24.X-X

Примечания.

1. Клеммы 1,8 и 13 электрически объединены внутри контролера, подключение датчиков к дискретным входам может осуществляться относительно любой из этих клемм.

2. Нагрузочное сопротивление аналогового выхода (R) составляет до 900 Ом при выходном сигнале «ток 4...20 мА» и более 2 кОм при выходном сигнале «напряжение 0...10 В». Подключение внешнего блока питания для аналоговых выходов не требуется, блок питания встроен в контроллер.

Arduino. Список распространенных ошибок и решений

Таблица 1. Список распространенных ошибок и решений /50/

Ошибка	Причина	Решение
Загрузка		
Программа не загружается на плату	Неправильно выбран COM-порт	Установите правильный порт в меню Сервис I Последовательный порт . Если вы не знаете, какой порт правильный, посмотрите его значение в Диспетчере устройств
Сообщение: avr-dude: stk500_getsync(): not in sync: resp=0x00	Неправильно выбран тип платы	Выберите правильно тип платы в меню Сервис I Плата
Невозможно выбрать правильный порт COM	Возможно, плата не была подключена к ПК во время запуска среды разработки Arduino IDE	Перезапустите Arduino IDE
Монитор порта и среда разработки больше не функционируют	Возможно, вы посылаете очень много данных через последовательный интерфейс	Отключите контроллер и закройте программу (в крайнем случае с помощью Диспетчера задач). Внесите изменения в программу - например, установите паузу с помощью функции <code>delay ()</code> . Загрузите программу заново
Плата Arduino после	Изменился COM-	Проверьте установку

кратковременного отключения больше не программируется	порт	COM-порта
Меню Сервис открывается очень медленно, работа Arduino IDE завершается аварийно	Устройства Bluetooth могут блокировать последовательный порт, который сканируется программным обеспечением Arduino	Деактивируйте устройство Bluetooth (хотя бы на время)
Меню Сервис открывается очень медленно, работа Arduino IDE завершается аварийно	Какое-то приложение на ПК блокирует порты COM в фоновом режиме. Этим приложением могут быть сканеры и брандмауэры	Проверьте установки в брандмауэре
Загрузка программ обрывается	Высокое потребление электроэнергии платой	Подключите дополнительно внешнее электропитание или удалите подключенные к плате периферийные устройства
Ошибка при загрузке	Соединены контакты 0 и 1, которые блокируют процесс загрузки	Удалите шилдт или кабели во время процесса загрузки
Среда разработки Arduino IDE «зависает» при загрузке	Возможны проблемы с программой LVPrsSrv.exe (утилитой Logitech для Windows)	Завершите процесс LVPrsSev в Диспетчере задач
Ошибки при компиляции		
error: expected “,”	Пропущена точка с	Исправьте программ-

before '}' token	запятой	ный код
error: expected “}” at of input	Пропущена фигурная скобка	Исправьте программный код
error: “Имя переменной” was not declared in this scope	Переменная не декларирована	Исправьте программный код
error: integer constant is too large for 'long' type	Целое число выходит за границы диапазона целых чисел	Исправьте программный код
java.lang. StackOverflowError [...]	Ошибка переполнения стека. Часто возникает при проблемах с описанием строк	Ищите в тексте программы ошибку, связанную с неправильным использованием двойных (""") или одинарных (") кавычек, косой черты (\), комментариев и т. д. (например, вместо ' \»' вы использовали '»' и т. п.)
Неожиданное поведение программы		
Программа не запускается снова после сброса кнопкой Reset на плате	Вы постоянно посылаете данные на последовательный порт платы и «новая» программа ожидает окончания этого процесса	Убедитесь, что приложение не отправляет непрерывно последовательные данные плате. При необходимости исправьте программный код
Программа успешно загрузилась, но не функционирует	К плате присоединено слишком много аппаратных средств	Подключите внешнее электроснабжение
Программа успешно загрузилась, но не	Выбран ошибочный тип платы	Проверьте установку платы (Сервис 1 Пла-

функционирует		та)
Скетч слишком большой	Программа слишком большая для выбранной платы	Оптимизируйте программный код или используйте внешние модули для хранения данных (например, SD-Card)
ШИМ (PWM) не функционирует	На данном контакте ШИМ не используется	Проверьте, имеется ли обозначение «~» на контакте
Ошибки инсталляции		
Java Virtual Machine Launcher: Could not find the main class. Program will exit	Возможно неправильно распаковался архив Arduino.zip	Распакуйте повторно архив и проверьте наличие файла pde.jar
Неправильно указан COM-порт	Возможно, что неправильно установлен драйвер	Проверьте установку драйвера в Диспетчере устройств и при необходимости переустановите его
Windows не определяет подключенную к USB плату Arduino	Плата подключена к разъему USB со спецификацией 3.0	Подключите плату к разъему USB 2.0
Драйвер не находится	Плата подключена к разъему USB со спецификацией 3.0	Подключите плату к разъему USB 2.0

Доработка схемы управления шаговым двигателем

Шаговый электродвигатель — это синхронный бесщёточный электродвигатель с несколькими обмотками, в котором ток, подаваемый в одну из обмоток статора, вызывает фиксацию ротора. Последовательная активация обмоток двигателя вызывает дискретные угловые перемещения (шаги) ротора.

Конструктивно шаговые электродвигатели состоят из статора, на котором расположены обмотки возбуждения, и ротора, выполненного из магнито-мягкого (ферромагнитного) материала или из магнито-твёрдого (магнитного) материала. Шаговые двигатели с магнитным ротором позволяют получать большой крутящий момент и обеспечивают фиксацию ротора при обесточенных обмотках.

Гибридные двигатели сочетают в себе лучшие черты двигателей с переменным магнитным сопротивлением и двигателей с постоянными магнитами.

Статор гибридного двигателя также имеет зубцы, обеспечивая большое количество эквивалентных полюсов, в отличие от основных полюсов, на которых расположены обмотки. Обычно используются 4 основных полюса для 3.6 град. двигателей и 8 основных полюсов для 1.8 — 0.9 град. двигателей. Зубцы ротора обеспечивают меньшее сопротивление магнитной цепи в определенных положениях ротора, что улучшает статический и динамический момент. Это обеспечивается соответствующим расположением зубцов, когда часть зубцов ротора находится строго напротив зубцов статора, а часть между ними.

Ротор гибридного двигателя имеет зубцы, расположенные в осевом направлении. Ротор разделен на две части, между которыми расположен цилиндрический постоянный магнит. Таким образом, зубцы верхней половины ротора являются северными полюсами, а зубцы нижней половины — южными. Кроме того, верхняя и нижняя половины ротора повернуты друг от-

носителем друга на половину угла шага зубцов. Число пар полюсов ротора равно количеству зубцов на одной из его половинок. Зубчатые полюсные наконечники ротора, как и статор, набраны из отдельных пластин для уменьшения потерь на вихревые токи.

Шаговые двигатели стандартизованы по посадочным размерам и размеру фланца: NEMA 17, NEMA 23, NEMA 34, ... — размер фланца 42 мм, 57 мм, 86 мм, 110 мм соответственно. Шаговые электродвигатели NEMA 23 могут создавать крутящий момент до 30 кгс*см, NEMA 34 до 120 кгс*см и до 210 кгс*см для двигателей с фланцем 110 мм.

1. Рассмотрение схемы управления шаговым двигателем

Схема и программа к PIC-контроллеру были разработаны и представлены на веб-сайте http://www.piclist.com/images/www/hobby_elec/index.htm японского радиолюбителя Seiichi Inoue. Позже они распространились по сети интернет и в данный момент доступны на многих сайтах.

В России статья под названием «PIC-контроллер управляет электродвигателем» впервые появилась в 6 номере журнала «Радио» в 2002 году, где описание к ней уже было переведено на русский язык. Принципиальная схема устройства изображена на рис. 1.

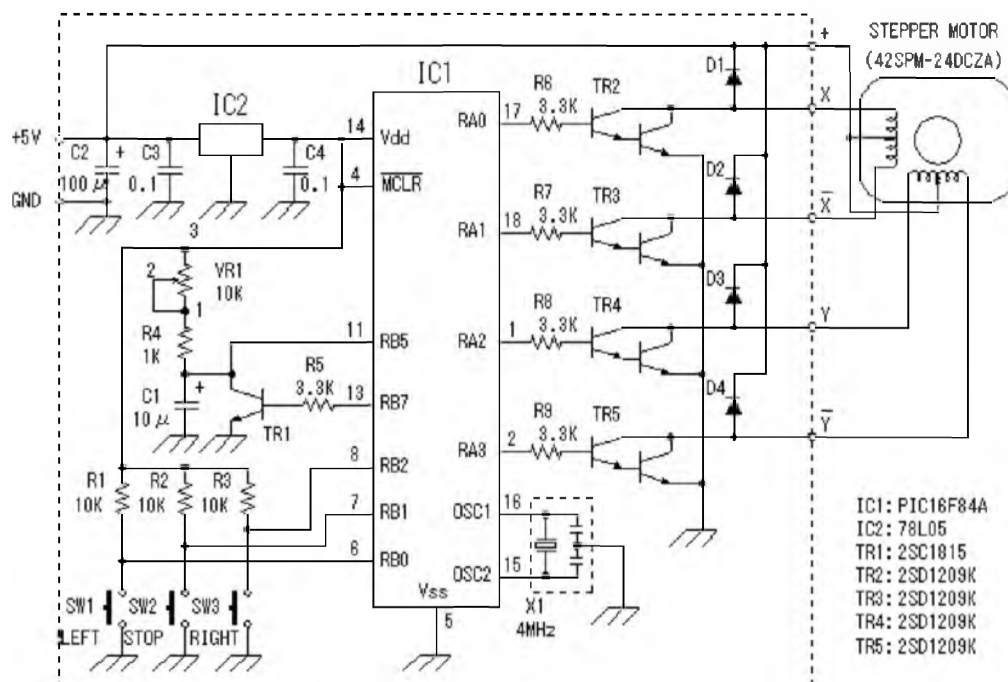


Рисунок 1 – Принципиальная электрическая схема управления шаговым двигателем

Основной служит МК PIC16F84A. Тактовую частоту (4 МГц) задает кварцевый резонатор X1. На элементах R4, VR1, C1 и транзисторе TR1 собран генератор, частоту которого можно плавно изменять переменным резистором VR1. Напряжение с конденсатора C1 поступает на вход RB5 МК IC1. После того как оно превысит пороговое, на выходе RB7 появляется напряжение высокого уровня. Открывшийся транзистор TR1 разряжает конденсатор, после чего цикл повторяется.

При перемещении движка резистора VR1 из одного крайнего положения в другое частота вращения двигателя M1 изменяется от 27 до 128 мин⁻¹. Следует учесть, что при увеличении частоты вращения снижается момент на валу двигателя. Устройство не имеет обратной связи, поэтому частота вращения зависит как от сопротивления введенной части резистора VR1, так и от нагрузки на валу.

Выходы RA0...RA3 МК через ключи, выполненные на составных транзисторах TR2...TR5, коммутируют напряжения на обмотках шагового

двигателя. Диоды D1...D4 защищают транзисторы от пробоя импульсами напряжения, возникающими в момент их закрывания.

К первым трем разрядам порта RB (RB0...RB2) МК подключены кнопки SW1...SW3, с помощью которых изменяют направление вращения вала двигателя и останавливают его.

2. Переработанная электрическая схема

В исходную электрическую схему были внесены изменения, улучшившие ее характеристики, а также были составлены три вспомогательные схемы. Одна из которых предназначена для индикации режима работы, а две другие для изменения режима работы основной схемы дистанционно.

Ниже представлены конечная схема для управления шаговым двигателем и схема передатчика (для переключения режимов работы на расстоянии), смоделированные в программе «Proteus 7.7».

На схемах не обозначены выводы питания микроконтроллеров (особенность программной среды). Подразумевается, что «+» питания соединен с 14 выходом микросхемы U2, 5 выходом U3 и U5. А «-» питания соединен с 5 выходом микросхемы U2, 2 выходом U3 и U5.

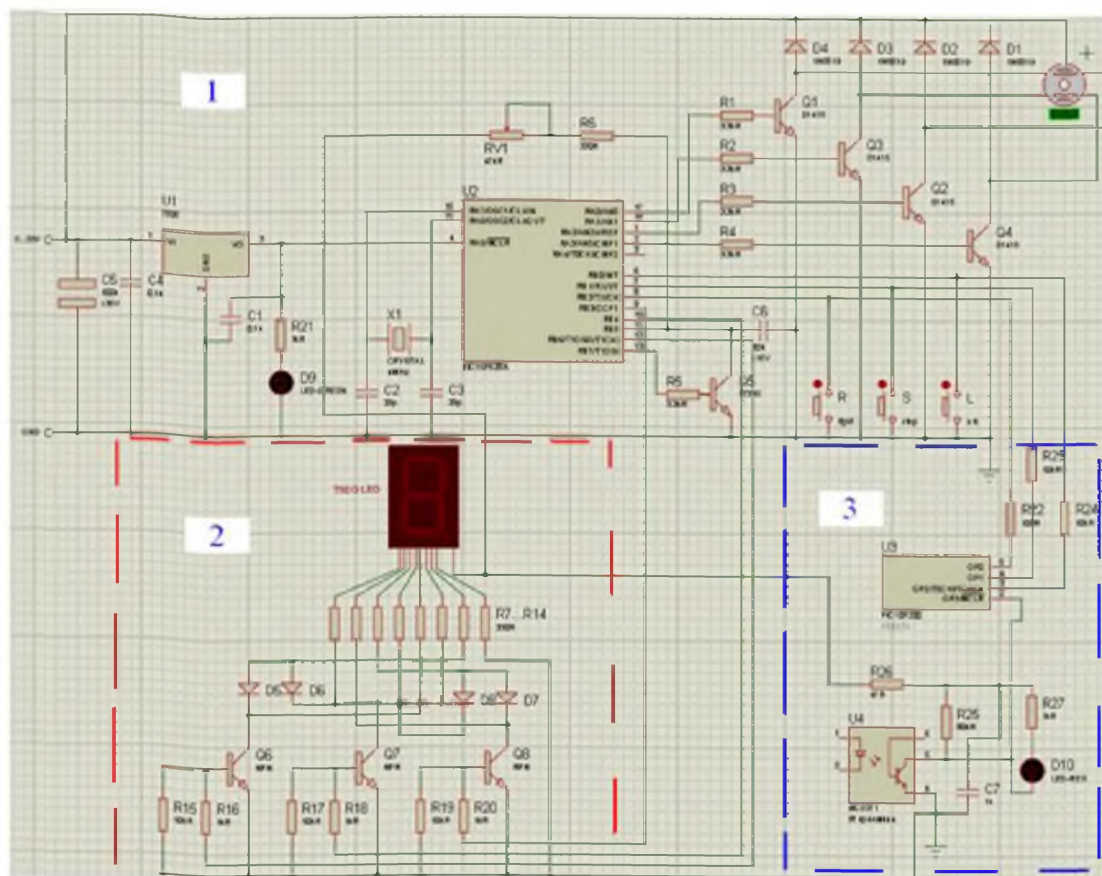


Рисунок 2 – Принципиальная электрическая схема основной части

Номерами 2 и 3 обозначены дополнительные схемы, разработанные специально для основной (под номером 1). Схема №2 предназначена для индикации режимов работы, а схема №3 для дистанционного приема команд схемы передатчика (рисунок 3).

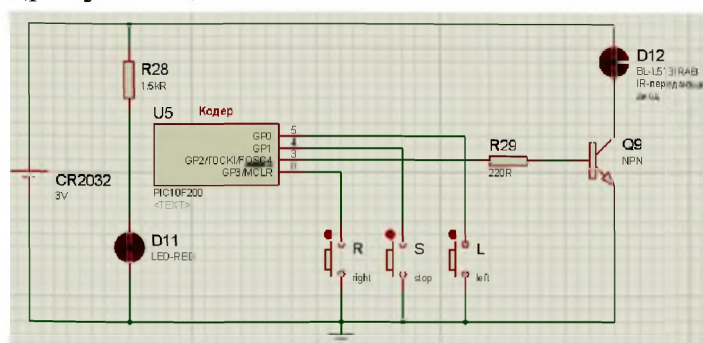


Рисунок 3 – Схема передатчика (вспомогательная часть)

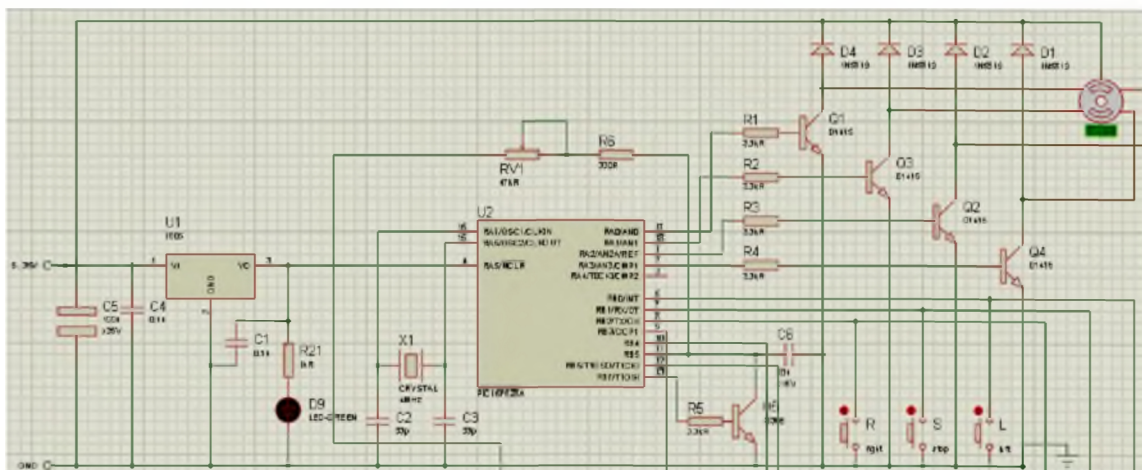


Рисунок 4 – Измененная основная схема

В исходную схему внесены следующие изменения (сравнение рисунков 1 и 4):

- 1) TR1 2SC1815 заменен на отечественный транзистор КТ315;
- 2) TR2 - TR4 2SD1209K заменены на D1415, имеющие лучшие характеристики (напряжение до 120 В, ток до 10 А);
- 3) R4 1кОм заменен на 330Ом (что увеличило максимальную скорость вращения шагового двигателя);
- 4) VR1 10кОм заменен на 47кОм (что уменьшило минимальную скорость вращения шагового двигателя);
- 5) Подача питания на R1 – R3 не от источника, а от контроллера U3 (рисунок 2). Объяснение в пункте 3.1;
- 6) IC2 78L05 заменена на 7805 (для работы с большими токами);
- 7) IC1 Pic16f84a заменена на Pic16f628a, имеющую меньшую стоимость и большую распространенность.

Основными элементами вспомогательных схем являются:

- для индикации режима работы (2 схема на рисунке 2) семисегментный индикатор с общим анодом;
- для схем передатчика и приемника два микроконтроллера Pic10f200, ИК-диод типа BL-L513IRAB и интегральная микросхема приемника MC3374.

Принцип работы основной части схемы остался неизменным, но из-за добавленной схемы №3 (рисунок 2) появилась возможность переключения режимов как с помощью кнопок (расположенных на основной схеме) так и дистанционно посредством передатчика (рисунок 3). Для этого, сопротивления R1, R2 и R3 соединены не с «+» питания, а с выходами U3.

Также для схемы №2 (рисунок 2) были задействованы ранее не используемые выходы микроконтроллера (RB3 – 9 выход, RB4 – 10 выход и RB6 – 12 выход). В этой части изменение коснулось только программы.

Схема под номером 2 предназначена для индикации режима работы.

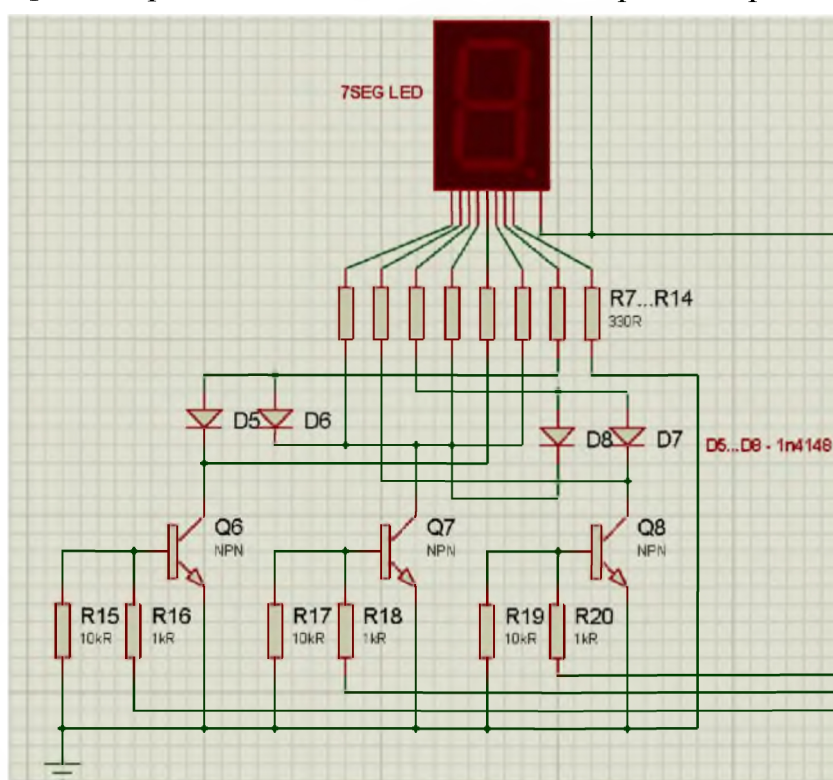


Рисунок 5 – Схема индикации

Возможны следующие режимы:

- отсутствие вращения двигателя (режим “stop” – на индикаторе знак S);
- вращение двигателя влево (режим “left” - на индикаторе знак l);
- вращение двигателя вправо (режим “right” - на индикаторе знак r).

Принцип работы данной схемы следующий: главным элементом является семисегментный индикатор с общим анодом и «+» питания подается на общий анод сразу. Катоды сегментов через токоограничивающие сопротивления коммутируются тремя транзисторами Q6 – Q8 (рисунок 2) в определенной последовательности, образуя нужные символы. Управляющие сигналы для открытия транзисторов подает микроконтроллер Pic16f628a посредством управляющей программы.

Диоды D5 – D8 используются для коммутации сегментов, образующих более одного знака. Так для образования буквы S используется 5 сегментов индикатора, буквы г – два, буквы l – два сегмента.

Схема на рисунке 3 предназначена для кодирования сигнала о нажатой кнопке и передачи его в виде излучения ИК светодиода. Кодирование происходит по времени. Каждая команда состоит из стартового бита длиной 0,5мс и бита команды (от 1мс до 3мс). На рисунке 6 изображены осциллограммы кодированных команд. Для того чтоб приемник их «увидел», они должны передаваться на частоте 33кГц (т.е. быть разбиты на прямоугольные импульсы соответствующей длины). В соответствии с этим, команды будут выглядеть так, как показано на рисунке 7.

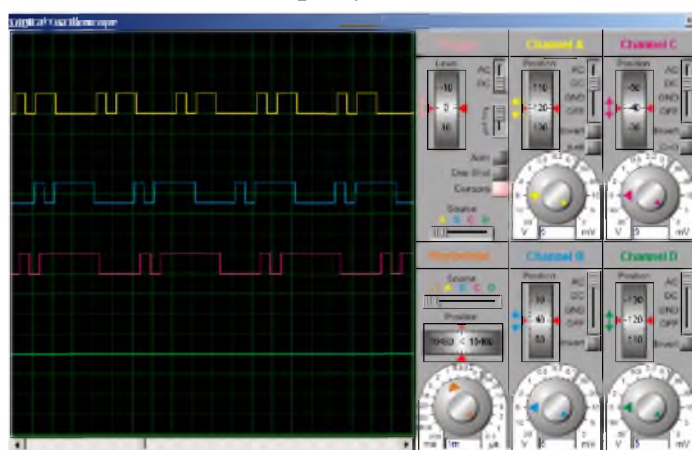


Рисунок 6 – Осциллограммы кодированных команд

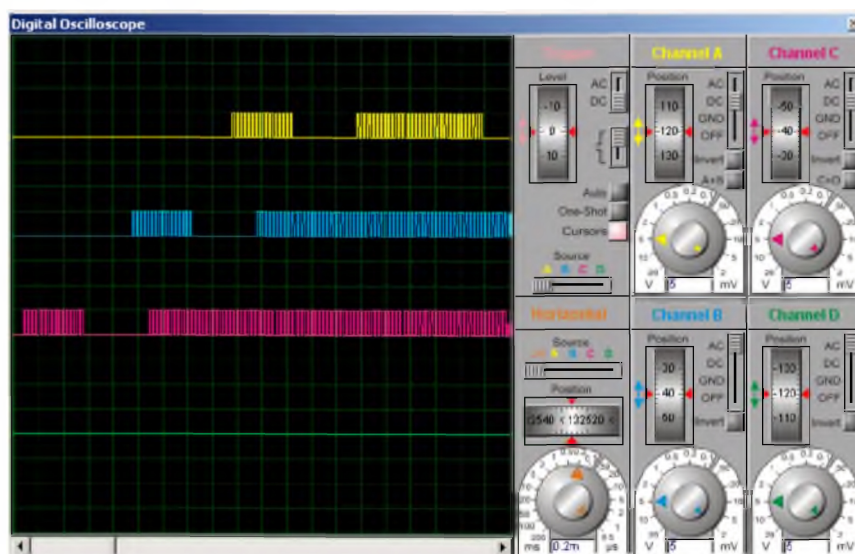


Рисунок 7 – Осциллограммы кодированных команд на выходе передатчика

Таким образом, при нажатии на одну из кнопок, на базу транзистора подаются пачки импульсов, соответствующие той или иной команде. Усиленные транзистором, они излучаются ИК светодиодом. Излучение происходит до момента, пока нажата кнопка.

Схема под номером 3 предназначена для приема и декодирования команд от передатчика (рисунок 3). Прием ведется на интегральную схему типа MC3374. Она работает с частотой 33кГц принимаемого ИК сигнала и содержит принимающий диод, фильтр и усилитель (на рисунке 8 U4 является не точным обозначением MC3374 и подразумевается, что R25 находится в ее корпусе). Входящие сигналы усиливаются и инвертируются (рисунок 9), после чего поступают на микроконтроллер, где идет проверка, какая из команд «пришла». Определение команды происходит по длительности рабочего бита, а распознавание очередной команды по стартовому биту.

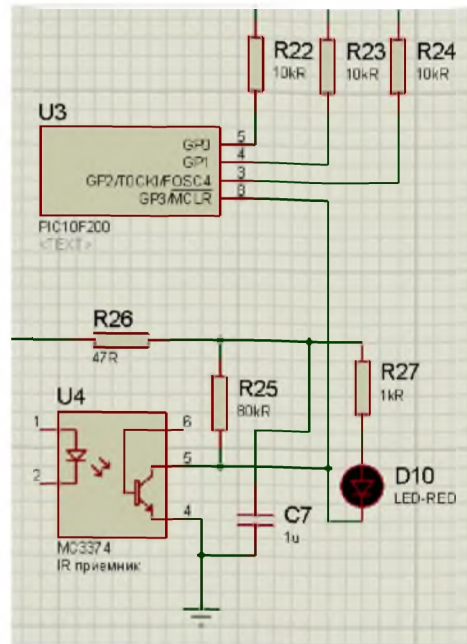


Рисунок 8 – Схема приемника и декодера

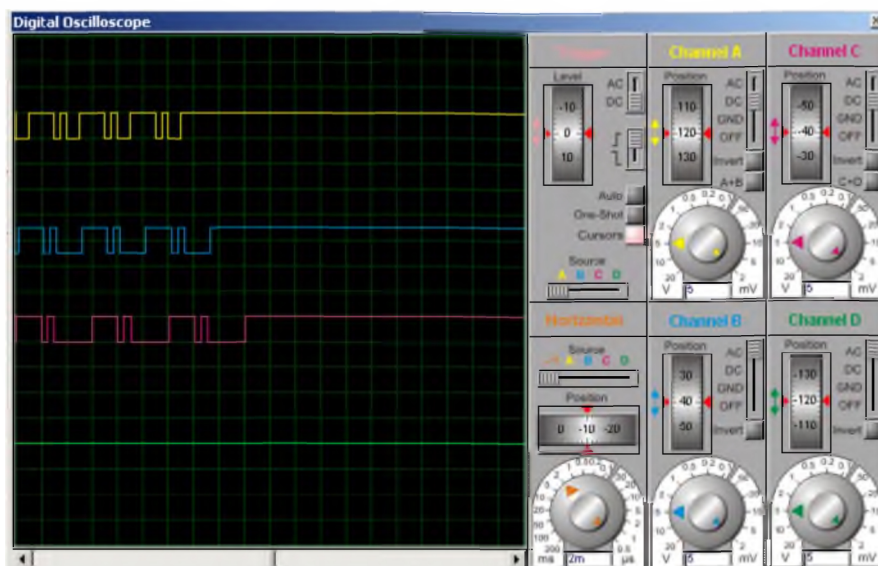


Рисунок 9 - Осциллограммы команд на выходе MC3374

На выходах микроконтроллера приемника установлена логическая “1”. Если происходит распознавание команды, то на одном из трех выходов логическая “1” сменяется логическим “0” (в зависимости от того какая из команд пришла), что является знаком для переключения режима работы двигателя и аналогично нажатию кнопки на схеме 1 (рисунок 2).

Ниже представлен исходный код программы Seiichi Inoue для микроконтроллера Pic16f84a (рисунок 1) на языке “Assembler” для программной среды MPLab IDE (интегрированная среда разработки для микроконтроллеров PIC microchip).

```

;
; *****
;
;               stepper Motor controller
;
;               Author : seiichi Inoue
; *****
;
;               list          p=pic16f84a
;               include       p16f84a.inc
;               __config _hs_osc & _wdt_off & _pwrtc_on & _cp_off
;
; ***** Label Definition *****
;
; cblock h'0c'
;
; mode                      ;Operation mode
;                          ;0=stop 1=right 2=left
; count1                    ;wait counter
; count2                    ;wait counter(for 1msec)
;
; endc
;
; rb0 equ 0                ;RB0 of PORTB
; rb1 equ 1                ;RB1 of PORTB
; rb2 equ 2                ;RB2 of PORTB
; rb5 equ 5                ;RB5 of PORTB
; rb7 equ 7                ;RB7 of PORTB
;
; ***** Program Start *****
;
; org 0                    ;Reset Vector
; goto init
; org 4                    ;Interrupt Vector
; clrf intcon               ;Clear Interruption reg
;
; ***** Initial Process *****
;
; init
;     bsf status,rp0        ;Change to Bank1
;     clrf trisa             ;Set PORTA all OUT
;     movlw b'00100111'     ;RB0,1,2,5=IN RB7=OUT
;     movwf trisb           ;Set PORTB
;     movlw b'10000000'     ;RBPU=1 Pull up not use
;     movwf option_reg      ;Set OPTION_REG
;     bcf status,rp0        ;Change to Bank0
;     clrf mode             ;Set mode = stop
;     clrf count1           ;Clear counter
;     clrf count2           ;Clear counter
;     movlw b'00000101'     ;Set PORTA initial value
;     movwf porta           ;Write PORTA
;     bsf portb,rb7         ;Set RB7 = 1
;     btfsc portb,rb5       ;RB5 = 0 ?
;     goto $-1              ;No. wait
;
; start
; ***** Check switch condition *****
;
;     btfsc portb,rb1       ;RB1(stop key) = ON ?
;     goto check1          ;No. Next
;     clrf mode             ;Yes. Set stop mode
;     goto drive           ;No. Jump to motor drive
;
; check1
;     btfsc portb,rb2       ;RB2(right key) = ON ?
;     goto check2          ;No. Next
;     movlw d'1'           ;Yes. Set right mode
;     movwf mode           ;Save mode
;     goto drive           ;No. Jump to motor drive
;
; check2
;     btfsc portb,rb0       ;RB0(left key) = ON ?
;     goto drive           ;No. Jump to motor drive
;     movlw d'2'           ;Yes. Set left mode
;     movwf mode           ;Save mode
;
; ***** Motor drive *****
;
; drive
;     movf mode,w           ;Read mode
;     bz start             ;mode = stop
;     bsf portb,rb7         ;Set RB7 = 1
;     btfsc portb,rb5       ;RB5 = 0 ?

```

```

        goto    $-1                ;No. Wait
        movlw   d'5'              ;Set loop count(5msec)
        movwf   count1            ;Save loop count
loop     call    timer             ;wait 1msec
        decfsz  count1,f          ;count - 1 = 0 ?
        goto    loop             ;No. Continue
        bcf     portb,rb7         ;Set RB7 = 0
        btfss  portb,rb5         ;RB5 = 1 ?
        goto    $-1              ;No. Wait
        movf    porta,w           ;Read PORTA
        sublw   b'000000101'     ;Check motor position
        bnz     drive2           ;Unmatch
        movf    mode,w           ;Read mode
        sublw   d'1'             ;Right ?
        bz      drive1           ;Yes. Right
        movlw   b'00001001'     ;No. Set Left data
        goto    drive_end        ;Jump to PORTA write
drive1   movlw   b'00000110'     ;Set Right data
        goto    drive_end        ;Jump to PORTA write
;-----
drive2   movf    porta,w         ;Read PORTA
        sublw   b'000000110'     ;Check motor position
        bnz     drive4           ;Unmatch
        movf    mode,w         ;Read mode
        sublw   d'1'             ;Right ?
        bz      drive3           ;Yes. Right
        movlw   b'00000101'     ;No. Set Left data
        goto    drive_end        ;Jump to PORTA write
drive3   movlw   b'00001010'     ;Set Right data
        goto    drive_end        ;Jump to PORTA write
;-----
drive4   movf    porta,w         ;Read PORTA
        sublw   b'000001010'     ;Check motor position
        bnz     drive6           ;Unmatch
        movf    mode,w         ;Read mode
        sublw   d'1'             ;Right ?
        bz      drive5           ;Yes. Right
        movlw   b'00000110'     ;No. Set Left data
        goto    drive_end        ;Jump to PORTA write
drive5   movlw   b'00001001'     ;Set Right data
        goto    drive_end        ;Jump to PORTA write
;-----
drive6   movf    porta,w         ;Read PORTA
        sublw   b'000001001'     ;Check motor position
        bnz     drive8           ;Unmatch
        movf    mode,w         ;Read mode
        sublw   d'1'             ;Right ?
        bz      drive7           ;Yes. Right
        movlw   b'00001010'     ;No. Set Left data
        goto    drive_end        ;Jump to PORTA write
drive7   movlw   b'00000101'     ;Set Right data
        goto    drive_end        ;Jump to PORTA write
;-----
drive8   movlw   b'00000101'     ;compulsion setting

drive_end
        movwf   porta            ;write PORTA
        goto    start           ;Jump to start

;***** 1msec Timer Subroutine *****
timer    movlw   d'200'          ;Set loop count
        movwf   count2          ;Save loop count
tm1p     nop                    ;Time adjust
        nop                    ;Time adjust
        decfsz  count2,f        ;count - 1 = 0 ?
        goto    tm1p           ;No. Continue
        return                  ;Yes. Contine end

;*****
;***** END of Stepper Motor controller *****
;*****
end

```

Ниже представлен текст измененной программы для работы с контроллером Pic16f628a (с учетом подачи управляющих сигналов на схему индикации режима). Текст программы также содержит комментарии, облегчающие его понимание.

```

    list      p=pic16f628a ;выбор контроллера
    config 3f22h          ;запись конфигураций

    cblock 20h           ;блок констант
mode      ;регистр режима
count1    ;0=стоп 1=вращение вправо 2=вращение влево
count2    ;регистр задержки
           ;регистр задержки на 1мс
    endc
;прописывание используемых регистров специального назначения
    intcon equ 0bh
    status equ 03h
    trisa equ 05h
    trisb equ 06h
    option_reg equ 01h
    porta equ 05h
    portb equ 06h
    cmcon equ 1fh

; ***** Начало программы *****
    org 0 ;первый адрес счетчика команд
    goto init
    org 4 ;вектор прерывания
    clrf intcon ;очистка регистра прерываний

; ***** Предварительные установки *****
init
    bcf status,7 ;переход в первый банк
    bsf status,5
    bcf status,6
    clrf trisa ;определение работы портаА на выход
    movlw b'00100111' ;определение работы портаВ RB0,1,2,5 - работа на вход RB3,4,6,7 - на выход
    movwf trisb ;запись в портВ
    movlw b'10000000' ;RVRU=1 подтягивающие резисторы не используются
    movwf option_reg ;установка регистра OPTION_REG
    bcf status,5 ;переход в нулевой банк
    clrf mode ;установка режима Стоп
    clrf count1 ;подготовка регистра задержки 1
    clrf count2 ;подготовка регистра задержки 2
    movlw 0x07 ;отключение компараторов
    movwf cmcon
    movlw b'00000101' ;запись в порта начального значения
    movwf porta
    bsf portb,7 ;установка 1 на 7 выходе портаВ
    bsf portb,3 ;установка 1 на 3 выходе портаВ
    btfsc portb,5 ;проверка 5 выхода портаВ (разряд RC цепочки через транзистор)
    goto $-1 ;зацикливание программы до окончания разряда

start
; ***** Проверка состояния портов (режимов вправо,стоп,влево) *****
    btfsc portb,1 ;проверка нажатия кнопки "стоп" (RB1)
    goto check1
    bcf portb,4 ;если кнопка не нажата (RB1=1) уходим для проверки остальных кнопок
    bcf portb,6 ;если кнопка нажата (RB1=0)
    bsf portb,3 ;то устанавливаем на выходах индикации
    clrf mode ;значения RB3=1, RB6=0, RB4=0
    goto drive ;записываем 0 в регистр режима
    ;и переходим в подпрограмму drive

check1
    btfsc portb,2 ;проверка нажатия кнопки "вправо" (RB2)
    goto check2
    bcf portb,3 ;если кнопка не нажата (RB2=1) уходим для проверки последней кнопки
    bcf portb,6 ;если кнопка нажата (RB2=0)
    bsf portb,4 ;то устанавливаем на выходах индикации
    movlw d'1' ;значения RB4=1, RB3=0, RB6=0
    movwf mode ;записываем 1 в регистр режима
    goto drive ;и переходим в подпрограмму drive

check2
    btfsc portb,0 ;проверка нажатия кнопки "влево" (RB0)
    goto drive
    bcf portb,3 ;если кнопка не нажата (RB0=1) уходим в подпрограмму drive
    bcf portb,4 ;если кнопка нажата (RB0=0)
    bsf portb,6 ;то устанавливаем на выходах индикации
    ;значения RB6=1, RB3=0, RB4=0

```

```

movlw d'2'      ;записываем 2 в регистр режима
movwf mode     ;и переходим в подпрограмму drive
;***** подпрограмма drive *****
drive
movf mode,0    ;проверка содержимого регистра режима копированием его содержимого в регистр аккумулятора
bz start      ;если содержимое регистра =0, то переходим на метку start в начало программы
bsf portb,7    ;если содержимое регистра <0, то повторяем разряд RS цепочки открыванием транзистора (RB7=1)
btfsc portb,5  ;ждем полного разряда
goto $-1       ;зацикливание программы до окончания процедуры разряда
movlw d'5'     ;запись числа 5 в регистр задержки
movwf count1   ;***
loop
call timer     ;переход по стеку в подпрограмму задержки на 1мс
decfsz count1,1 ;вычитание 1 из регистра задержки count1 и зацикливание пока count1 не станет =0
goto loop      ;безусловный переход до окончания задержки (5мс)
bcf portb,7    ;установка 0 на 7 выходе портаB (RB7)
btfss portb,5  ;проверка состояния входа RB5 (заряд конденсатора до напряжения питания)
goto $-1       ;зацикливание до окончания процесса заряда
movf porta,0   ;копирование значения портаA в регистр аккумулятора
sublw b'000000101' ;вычитание из аккумулятора значение константы (проверка позиции ротора двигателя)
bnz drive2     ;если положение ротора не совпадает с необходимым, то переходим в подпрограмму drive2
movf mode,0    ;чтение регистра mode (копирование значения регистра в регистр аккумулятора)
sublw d'1'     ;вычитание из аккумулятора 1 (проверка режима)
bz drive1      ;если в регистре режима содержалась 1 (что соответствует нажатию кнопки "вправо" (RB2)
;то переходим в подпрограмму drive1
movlw b'00001001' ;если в регистре режима не содержалась 1 копируем число 9 в аккумулятор
goto drive_end   ;переход в подпрограмму drive_end для записи числа из аккумулятора в регистр порта
drive1
movlw b'00000110' ;копирование в регистр аккумулятора числа 6
goto drive_end   ;переход в подпрограмму drive_end для записи числа из аккумулятора в регистр порта
;-----
drive2
movf porta,0    ;копирование значения портаA в регистр аккумулятора
sublw b'000000110' ;вычитание из аккумулятора значение константы (проверка позиции ротора двигателя)
bnz drive4     ;если положение ротора не совпадает с необходимым, то переходим в подпрограмму drive4
movf mode,0    ;чтение регистра mode (копирование значения регистра в регистр аккумулятора)
sublw d'1'     ;вычитание из аккумулятора значение константы(проверка позиции ротора двигателя)
bz drive3      ;если в регистре режима содержалась 1 (что соответствует нажатию кнопки "вправо" (RB2)
;то переходим в подпрограмму drive3
movlw b'00000101' ;если разность <>0 то записываем в регистр аккумулятора данные для режима работы "влево"
goto drive_end   ;переход в подпрограмму drive_end для записи числа из аккумулятора в регистр порта
drive3
movlw b'00001010' ;если разность =0 то записываем в регистр аккумулятора данные для режима работы "вправо"
goto drive_end   ;переход в подпрограмму drive_end для записи числа из аккумулятора в регистр порта
;-----
drive4
movf porta,0    ;копирование значения портаA в регистр аккумулятора
sublw b'000001010' ;вычитание из аккумулятора значение константы (проверка позиции ротора двигателя)
bnz drive6     ;если положение ротора не совпадает с необходимым, то переходим в подпрограмму drive6
movf mode,0    ;чтение регистра mode (копирование значения регистра в регистр аккумулятора)
sublw d'1'     ;вычитание из аккумулятора значение константы(проверка позиции ротора двигателя)
bz drive5      ;если в регистре режима содержалась 1 (что соответствует нажатию кнопки "вправо" (RB2)
;то переходим в подпрограмму drive5
movlw b'00000110' ;если разность <>0 то записываем в регистр аккумулятора данные для режима работы "влево"
goto drive_end   ;переход в подпрограмму drive_end для записи числа из аккумулятора в регистр порта
drive5
movlw b'00001001' ;если разность =0 то записываем в регистр аккумулятора данные для режима работы "вправо"
goto drive_end   ;переход в подпрограмму drive_end для записи числа из аккумулятора в регистр порта
;-----
drive6
movf porta,0    ;копирование значения портаA в регистр аккумулятора
sublw b'000001001' ;вычитание из аккумулятора значение константы (проверка позиции ротора двигателя)
bnz drive8     ;если положение ротора не совпадает с необходимым, то переходим в подпрограмму drive8
movf mode,0    ;чтение регистра mode (копирование значения регистра в регистр аккумулятора)
sublw d'1'     ;вычитание из аккумулятора значение константы(проверка позиции ротора двигателя)
bz drive7      ;если в регистре режима содержалась 1 (что соответствует нажатию кнопки "вправо" (RB2)
;то переходим в подпрограмму drive7
movlw b'00001010' ;если разность <>0 то записываем в регистр аккумулятора данные для режима работы "влево"
goto drive_end   ;переход в подпрограмму drive_end для записи числа из аккумулятора в регистр порта
drive7
movlw b'00000101' ;если разность =0 то записываем в регистр аккумулятора данные для режима работы "вправо"
goto drive_end   ;переход в подпрограмму drive_end для записи числа из аккумулятора в регистр порта
;-----
drive8
movlw b'00000101' ;если положение ротора не совпадает не с одним из возможных, то копируем в регистр
;аккумулятор значение для выходов портаA соответствующее начальному значению
drive_end
movwf porta     ;копирование числа из регистра аккумулятора в регистр порта
goto start     ;переход к началу программы
;***** Подпрограмма задержки на 1мс *****
timer
movlw d'200'    ;копирование числа 200 в регистр аккумулятора
movwf count2    ;копирование числа 200 из регистра аккумулятора в регистр задержки count2
tmpr            ;два пустых цикла для корректировки значения задержки
;***
decfsz count2,1 ;вычитание 1 из регистра задержки
goto tmpr       ;переход на метку tmpr если результат не 0 (зацикливание программы)
return          ;возвращение по стеку из подпрограммы задержки
end             ;команда окончания программы

```

Текст, выделенный подчеркиваниями, необходим для работы программы на контроллере Pic16f628a. Часть, подчеркнутая двумя чертами, определяет тип контроллера и его конфигурации (внутренние установки).

```

list      p=pic16f628a
__config 3f22h

```

Части, подчеркнутые одной чертой, вносят изменения в настройки, которых нет у контроллера Pic16f84a:

`cmcon equ 1fh` (прописывание регистра специального назначения `cmcon`, для работы с компараторами).

`bcf status,7` и `bcf status,6` (выбор группы «банков». В контроллере `Pic16f84a` всего 2 «банка», а в `Pic16f628a` – 4. Эти две команды говорят о использовании только первых двух).

`movlw 0x07` (копирование и запись числа 7 в регистр `cmcon`)

`movwf cmcon` (это позволяет отключить компараторы, находящиеся на входах `RA0...RA3` и использовать их в соответствии с программой).

Части кода программы, выделенные рамкой, используются для создания управляющего напряжения на `RB3`, `RB6` или `RB4` (в соответствии с текущим режимом работы логическая “1” устанавливается на одном из этих выходов. Наличие логического уровня открывает или закрывает транзисторы `Q6 – Q8` схемы индикации на рисунке 5).

Ниже представлена программа для передатчика.


```

List      p=p1c10f200
_config   h'0f09'

INDF      EQU    H'0000'
TMR0      EQU    H'0001'
PCL       EQU    H'0002'
STATUS    EQU    H'0003'
FSR       EQU    H'0004'
OSCCAL    EQU    H'0005'
GPIO      EQU    H'0006'
Reg_1     equ    H'0010' ;регистр задержки 1
Reg_2     equ    H'0011' ;регистр задержки 2
Reg       equ    H'0012' ;регистр счета

org 0
goto start
org 4
goto start

start     movlw    b'00000000'
          option
          movlw    b'1011'
          tris     GPIO
          clrf     GPIO
mm1       btfss    GPIO,0
          goto     ma
mm2       btfss    GPIO,1
          goto     mb
mm3       btfss    GPIO,3
          goto     mc
          goto     mm1
;////////////////////////////////////////
ma        call     startbit
          call     Pause0
          call     osnbit
          call     Pause1
          goto     mm1 ;нажатие не более одной кнопки за раз
mb        call     startbit
          call     Pause0
          call     osnbit
          call     Pause1
          goto     mm2 ;нажатие не более одной кнопки за раз
mc        call     startbit
          call     Pause0
          call     osnbit
          call     Pause1
          goto     mm3 ;нажатие не более одной кнопки за раз
;////////////////////////////////////////
startbit  movlw    .17
          movwf    Reg
to        decfsz   Reg,1
          goto     for
          return
;формирование 0,5 мс
for       bsf      GPIO,2
          call     Pause
          bcf      GPIO,2
          call     Pause
          goto     to
osnbit    movlw    .34
          movwf    Reg
tol       decfsz   Reg,1
          goto     for1
          return
;формирование 1 мс
for1      bsf      GPIO,2
          call     Pause
          bcf      GPIO,2
          call     Pause
          goto     tol
;////////////////////////////////////////

;задержка 15 мкс
Pause     movlw    .2
          movwf    Reg_1
wr        decfsz   Reg_1, F
          goto     wr
          nop
          return
;задержка 0,5 мс
Pause0    movlw    .166
          movwf    Reg_1
wr0       decfsz   Reg_1, F
          goto     wr0
          nop
          return
;задержка 2 мс
Pause1    movlw    .151
          movwf    Reg_1
          movlw    .3
          movwf    Reg_2
wr1       decfsz   Reg_1, F
          goto     wr1
          decfsz   Reg_2, F
          goto     wr1
          nop
          nop
          return
end

```

Ниже представлена программа для приемника.

```

List      3=pic10f200
_config   0'0F09'

INDF      EQU    H'0000
TMR0      EQU    H'0001
PCL        EQU    H'0002
STATUS     EQU    H'0003
FSR        EQU    H'0004
GPIO       EQU    H'0006
Reg_1      equ    H'0010 ;регистр задержки 1
Reg_2      equ    H'0011 ;регистр задержки 2
Reg_us1    equ    H'0012 ;регистр условия 1
Reg_us11   equ    H'0013 ;регистр условия 2

org 0
goto start
org 4
goto start

start     movlw    b'01000000'
          option
          movlw    b'1000'
          tris     GPIO

n1        call     one ;присвоение 1 RP0, RP1, RP2

;////////////////////////////////////////
n1        btfsc    GPIO,3 ;начало программы
          goto     n1 ;защипливание до прихода команды
;задержка 570мкс с проверкой окончания импульса
          movlw    .116
          movwf    Reg_1
a1        btfsc    GPIO,3
          goto     n2
          decfsz   Reg_1, F
          goto     a1
          goto     n1 ;возвращение в начало программы если не пришел стартовый бит
;задержка 660мкс с проверкой окончания импульса
n2        movlw    .133
          movwf    Reg_1
a2        btfss    GPIO,3
          goto     n3
          decfsz   Reg_1, F
          goto     a2
          goto     n1 ;возвращение в начало программы если не пришел сигнальный бит
n3        call     Pause ;задержка для перехода к середине импульса
          btfsc    GPIO,3
          goto     n1 ;если бит короче 1мс то ошибка и переход в начало программы
          call     Pause ;выход за значение первой команды
          call     Pause
          btfsc    GPIO,3 ;если бит закончился, то пришла первая команда
          goto     m1
          call     Pause ;выход за значение второй команды
          call     Pause
          btfsc    GPIO,3 ;если бит закончился, то пришла вторая команда
          goto     m2
          call     Pause ;выход за значение третьей команды
          call     Pause
          btfsc    GPIO,3 ;если бит закончился, то пришла третья команда
          goto     m3
          goto     n1 ;если бит не закончился, то ошибка и переход в начало программы
;////////////////////////////////////////
m1        btfss    Reg_us1,0 ;проверка повторного прихода команды
          call     one ;установка 1 на выходных портах контроллера
          clrf     Reg_us1 ;обнуление регистра условия
          movlw    h'1'
          addwf    Reg_us11
          bsf      Reg_us1,0 ;запись признака первой команды
          btfsc    Reg_us1,3
          bcf      GPIO,0 ;установка 0 на выход RP0
          goto     prov
;////////////////////////////////////////
m2        btfss    Reg_us1,1 ;проверка повторного прихода команды
          call     one ;установка 1 на выходных портах контроллера
          clrf     Reg_us1 ;обнуление регистра условия
          movlw    h'1'
          addwf    Reg_us11
          bsf      Reg_us1,1 ;запись признака второй команды
          btfsc    Reg_us1,3
          bcf      GPIO,1 ;установка 0 на выход RP1
          goto     prov
;////////////////////////////////////////
m3        btfss    Reg_us1,2 ;проверка повторного прихода команды
          call     one ;установка 1 на выходных портах контроллера
          clrf     Reg_us1 ;обнуление регистра условия
          movlw    h'1'
          addwf    Reg_us11
          bsf      Reg_us1,2 ;запись признака третьей команды
          btfsc    Reg_us1,3
          bcf      GPIO,2 ;установка 0 на выход RP2
;////////////////////////////////////////
;задержка около 10мс с проверкой прихода нового импульса
prov      movlw    .98
          movwf    Reg_1
          movlw    .11
          movwf    Reg_2
a3        btfss    GPIO,3 ;начало программы
          goto     n1
          decfsz   Reg_1, F
          goto     a3
          decfsz   Reg_2, F
          goto     a3
          goto     n1
;////////////////////////////////////////
one        bsf      GPIO,0 ;запись 1 на все выходы
          bsf      GPIO,1
          bsf      GPIO,2
          clrf     Reg_us11
          return
;////////////////////////////////////////
;задержка 0,5мс
Pause     movlw    .166
          movwf    Reg_1
wr        decfsz   Reg_1, F
          goto     wr
          nop
          return
end

```

ПРОГРАММИРОВАНИЕ МЕХАТРОННЫХ СИСТЕМ. CODESYS И ARDUINO

Автор-составитель –
Моногаров Сергей Иванович

*Учебное пособие по изучению дисциплины
«Основы микропроцессорной техники»
для студентов всех форм обучения направления
13.03.02 – Электроэнергетика и электротехника*

Редакционно-издательский отдел
Зав. отделом: А.О. Белоусова
Компьютерная вёрстка: Л.В. Зданевич
Печать и послепечатная обработка: С.В. Татаренко
Подписано к печати 27.09.2018. Формат 60×90/16.
Усл. печ. л. 7. Уч.-изд. л. 2,88. Тираж 550 экз.
Заказ № 89/18.

ФГБОУ ВО «Армавирский государственный педагогический университет»
Редакционно-издательский отдел
© АГПУ, 352900, Армавир, ул. Ефремова, 35
☎fax 8(86137)32739, e-mail: rits_agpu@mail.ru, сайт: rits.agpu.net