Лабораторная работа №3

Цель работы:

Моделирование управления трехфазного электродвигателя с использованием автоматного подхода при помощи микроконтроллера.

Программное обеспечение:

Proteus, STM32CubeIDE.

Общие сведения:

Схема управления трёхфазным двигателем при помощи микроконтроллера приведена на рисунке 1.

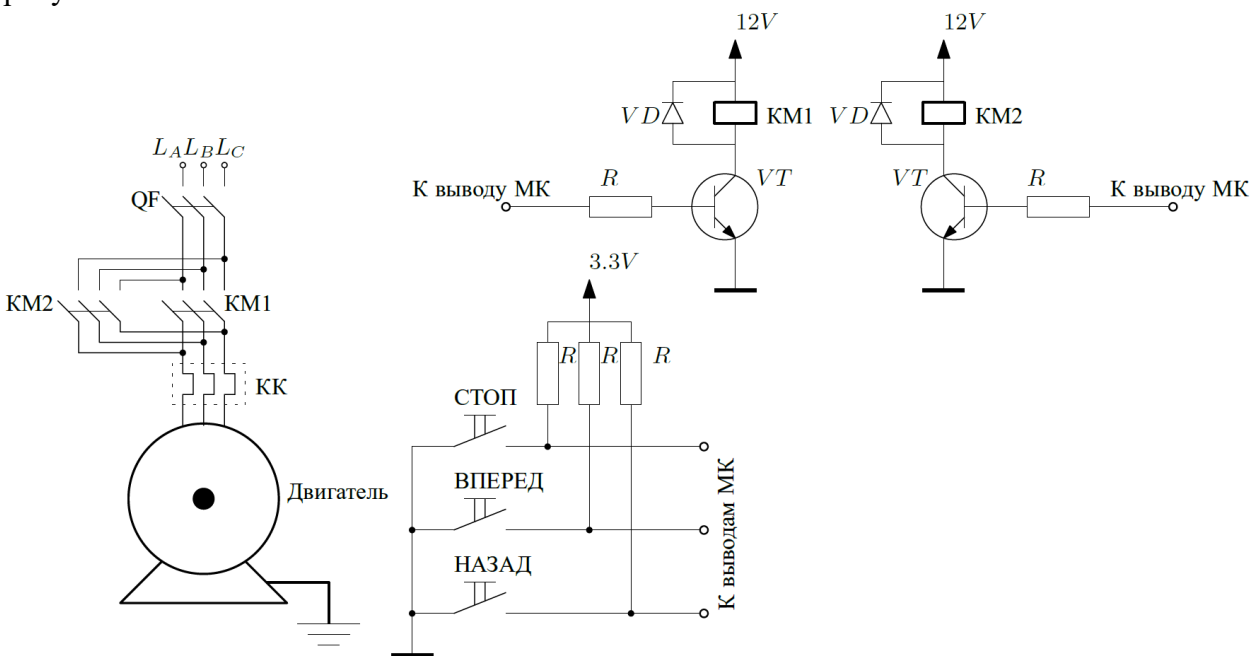


Рисунок 1 – Схема управления двигателем

Основные обозначения совпадают с лабораторной 1.

Схема работает следующим образом:

Фазы А, В и С питающего напряжения подводятся к клеммам

асинхронного двигателя через:

- трехполюсный автоматический выключатель, который защищает всю схему и позволяет отключать питающее напряжение (QF);
- поочередно через три пары силовых контактов реле КМ1 и КМ2;
- тепловое реле КК, которое служит для защиты от перегрузок.

Для изменения направления вращения трехфазного электродвигателя необходимо поменять местами подключение любых двух фаз. Для этого в цепь обмотки двигателя включены силовые

контакты от двух пускателей, которые подключаются поочередно, меняя чередование фаз. В схеме при вращении вперед последовательность фаз А, В, С, при вращении назад - С, В, А, т.е. чередование фаз А и С.

Схема управления включает в себя три кнопки, которые подключаются к микроконтроллеру:

- нормально разомкнутая ВПЕРЕД;
- нормально разомкнутая НАЗАД;
- нормально разомкнутая СТОП.

Силовые реле управляются микроконтроллером через транзисторы, это необходимо по двум причинам – реле большая нагрузка для выводов микроконтроллера, и напряжение питания реле превышает допустимое напряжение на выводах микроконтроллера. Кроме того, в цепь реле параллельно устанавливают диод, он защищает схему от импульса от катушки реле при выключении. Нельзя допускать одновременного замыкания 2х реле!

Работа схемы

Переводим рычаг трехполюсного автоматического выключателя во включенное положение, его контакты замыкаются, схема готова к работе. На микроконтроллер в начальном состоянии на линиях управления реле КМ1 и КМ2 установлено логическое 0.

1. Запуск вперед

Нажимаем кнопку ВПЕРЕД. Микроконтроллер считывает состояние кнопки и проверяет какое значение установлено на каждой из управляющих реле линий. Если на обоих линиях уставлено значение логического 0, это означает, что двигатель остановлен и можно его запустить для вращения вперед. Для этого на линии управления КМ1 устанавливается логическая 1. Если на линии КМ1 уже установлена логическая 1, то двигатель уже вращается вперед и ничего предпринимать не требуется. Если на линии КМ2 установлена логическая 1, то двигатель вращается назад и его невозможно переключить на вращение вперед.

2. Останов двигателя.

Нажимаем кнопку СТОП. Микроконтроллер считывает состояние кнопки и устанавливает на обоих линиях управления реле логические 0.

3. Реверс двигателя

Чтобы запустить двигатель в обратном направлении, нажимаем кнопку НАЗАД. Микроконтроллер считывает состояние кнопки и проверяет какое значение установлено на каждой из управляющих реле линий. Если на обоих линиях уставлено значение логического 0, это означает, что двигатель остановлен и можно его запустить для вращения в обратно направлении. Для этого на линии управления КМ2 устанавливается логическая 1. Если на линии КМ2 уже установлена логическая 1, то двигатель уже вращается в нужном направлении и ничего предпринимать не требуется. Если на линии КМ1 установлена логическая 1, то двигатель вращается вперед и его невозможно переключить на вращение вперед.

4. Защита от перегрузок

Защита от перегрузок выполняется тепловым реле КК.

Моделирование системы.

Создание схемы в Proteus.

1. Запустите Proteus. Создайте новый проект, задайте имя проекта и расположение (удобнее всего сделать отдельную папку для проекта Proteus и STM IDE), далее следует выбрать: «Create a schematic from selected template – DEFAULT», «Do not create a PCB layout», «No Firmware Project».

2. Для добавления элементов нажмите Library->Pick parts на панели инструментов (или пр.кн.мыши на области схемы Place->Component->From library). В поле поиска введите

название необходимого компонента, затем выберите нужный компонент из списка и нажмите кнопку “OK”.

3. Соберите схему согласно рисунку 1.

Для моделирования двигателя используйте «MOTOR-3PH», в настройках установите момент инерции = 0.0000000005.

Для моделирования 3х-фазной сети используйте 3 генератора сигнала, со следующими параметрами:

- тип выходного сигнала: sin;
- амплитуда: 10V;
- частота: 10 Гц;
- фаза: для первого источника – 0, для второго источника – 120, для третьего источника 240.

Для моделирования реле KM1 и KM2 используйте 3 однополюсных реле G2RL-1A-CF-DC12 с соединёнными катушками управления.

В качестве транзисторов можно использовать компонент – BC547, резисторы – RES, кнопки – BUTTON, диоды – DIODE.

Итоговая схема управления двигателем представлена на рисунке 2.

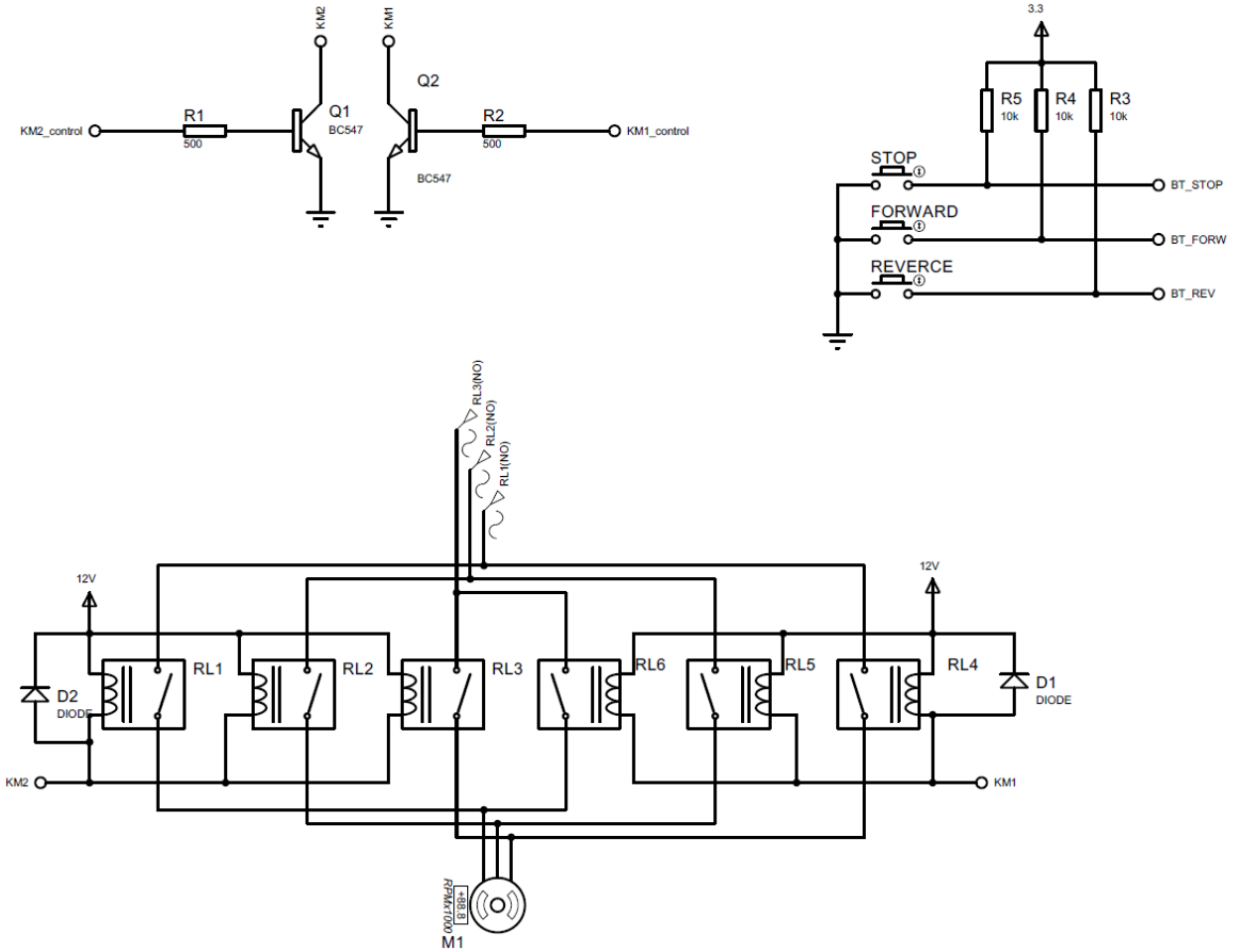


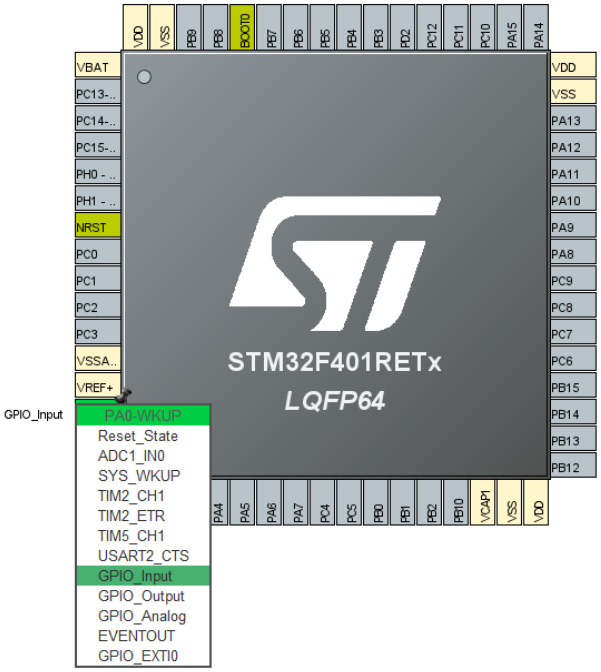
Рисунок 2 – Схема управления двигателем в Proteus

4. После сборки схемы необходимо настроить линии питания, для этого необходимо перейти в «Design->Configure Power Rails». Выбрать имя линии – VDD/VDD и добавить в нее линии 3.3V, и указать напряжение 3.3. Далее создать новую линию питания – 12V, установить напряжение 12 и добавить в нее силовую линию 12V.

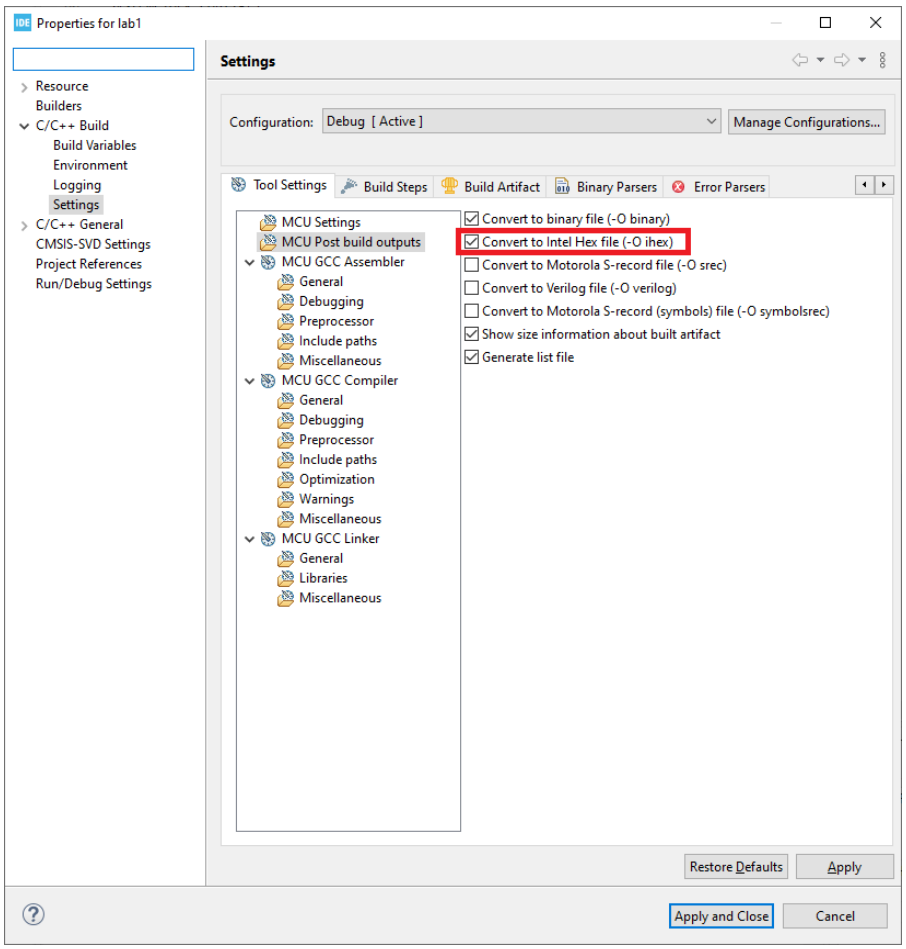
5. Добавьте на схему микроконтроллер STM32F401RE.
Подключите сигналы от кнопок и реле к выводам микроконтроллера.
Запомните, к каким выводам подключены какие сигналы.

Создание прошивки для микроконтроллера.

1. Запустите STM32CubeIDE, в качестве workspace выберете папку для лабораторной работы. Создайте новый проект «Start new STM32 project», в поле выбора микроконтроллера выберете STM32F401RE. Нажмите Next, задайте имя проекта, выберете язык – C, типы выходного файла – Executable, тип проекта – STM32Cube; нажмите Finish. В пути и названии проекта не должно быть русских букв.
2. В открывшемся окне появится схематичное изображение микроконтроллера. Для того что бы задать режим работы пина, необходимо нажать на него и выбрать режим. Задайте выводам режим работы, для сигналов от кнопок – GPIO_Input, для сигналов к реле – GPIO_Output.



3. Сгенерируйте код, для этого нажмите Project->Generate Code на панели инструментов. Откроется основной файл программы – “main.c”. Если “main.c” не открылся, то его необходимо открыть вручную, он находится в Project Explorer по пути Core->Src->main.c. Откройте свойства проекта, для этого нажмите Project->Properties, далее перейдите C/C++ Build->Settings->Tool settings->MCU Post build options и установите галочку напротив пункта Convert to Intel Hex file.



Примените и закройте свойства.

4. Используйте область “USER CODE BEGIN 2” для задания переменных и область бесконечного цикла “USER CODE BEGIN 3” для написания кода управления двигателем. Чтобы получить информацию о состоянии пина необходимо написать: «HAL_GPIO_ReadPin(GPIOx, GPIO_PIN_#);», где x – имя порта, # – номер пина к которому подлечена кнопка. Например если кнопка подключена к пину 2 порта C то команда будет выглядеть следующим образом: “HAL_GPIO_ReadPin(GPIOC, GPIO_PIN_2);”.
Для того, что бы задать выводному порту логическую 1 необходимо написать «HAL_GPIO_WritePin(GPIOx, GPIO_PIN_#, GPIO_PIN_SET);», а для установки логического 0: «HAL_GPIO_WritePin(GPIOx, GPIO_PIN_#, GPIO_PIN_RESET);».
5. После написания кода нажмите Project->Build all для компиляции кода. Если в программе нет ошибок то в Build Console будет надпись: “Build Finished”.

Запуск программы в Proteus.
1. Откройте окно со схемой в Proteus. Перейдите в свойства микроконтроллера дважды на него нажав, в области Program File выберете скомпилированный STM32 IDE файл. Этот файл находится в папке Debug проекта и имеет расширение «ИМЯ ПРОЕКТА.hex».
2. Используйте кнопки в левом нижнем углу окна Proteus для запуска и остановки симуляции.

Ход работы

1. Разработать схему в Proteus.
2. Разработать программу управления двигателем.
3. Провести моделирование для следующих ситуаций:

- Двигатель стартует в прямом направлении из положения покоя, затем останавливается.
- Двигатель стартует в обратном направлении из положения покоя, затем останавливается.
- Нажимается кнопка пуска двигателя в обратном направлении при вращении в прямом направлении.

Содержание отчета:

- Титульный лист.
- Цель работы.
- Схема Proteus.
- Код программы управления двигателем.
- Вывод.