

Моделирование алгоритма управления двигателем для микроконтроллера STM32 с использованием метода модельно-ориентированного проектирования

П.С. Зубков, Н.А. Кулаев, В.Ю. Артамонов, Н.В. Дубов

Мордовский государственный университет им. Н.П. Огарёва, Саранск

Аннотация: В данной работе рассмотрены этапы моделирования алгоритма управления состояниями синхронного двигателя с постоянными магнитами. В его основу заложен принцип векторного частотно-токового регулирования. Модель управления создана в пакете Matlab/Simulink, подсистемы которой описаны, и приведена классификация сигналов для микроконтроллера STM32. На последнем этапе промоделированы и визуализированы входные/выходные сигналы системы при разных исходных значениях скорости и тока инвертора, на основании которых сделан вывод о работоспособности модели для её дальнейшей генерации как С-кода и интеграции в микроконтроллер.

Ключевые слова: Модельно-ориентированное проектирование, Matlab/Simulink, STM32, синхронный двигатель с постоянными магнитами, векторное управление.

Метод модельно-ориентированного проектирования подразумевает использование графических инструментов Matlab/Simulink для моделирования и отладки системы управления, например, блок-схемы и конечные автоматы. Блок-схемы избавляют от необходимости формирования дифференциальных уравнений, а конечные автоматы – способ справиться с операторами ветвления [1].

Алгоритм управления – это набор инструкций, которые выполняются в зависимости от входных показаний и текущего состояния внутренних переменных, внося изменения в эти внутренние переменные и обновляя выходные значения. Это происходит один раз в каждом цикле генератора тактового сигнала или таймера, вызванного его передним фронтом [2]. Цель алгоритма управления, представленного в этой работе, состоит в том, чтобы управлять состоянием синхронного двигателя с постоянными магнитами (PMSM) с помощью микроконтроллера, примером которого является в данном случае частотно-токовое управление (I-Hz управление регулирует

положение и амплитуду вектора тока статора во вращающихся осях), показанное на рисунке 1 в виде блок-схемы. Однако алгоритм должен быть организован как конечный автомат, чтобы правильно планировать выполнение задач. Это реализуется при помощи Stateflow Simulink [3, 4].

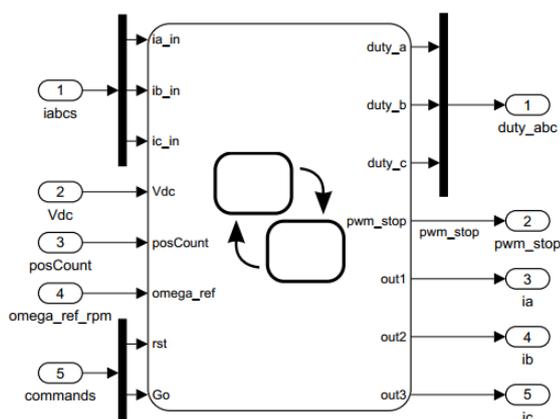


Рисунок 1 – Алгоритм управления в Simulink-модели: I-Hz_model.slx

Блок Chart из Stateflow, доступный в Simulink Library Browser, является своего рода подсистемой, в которой можно изобразить конечные автоматы. На рисунке 2 показаны три параллельных конечных автомата.

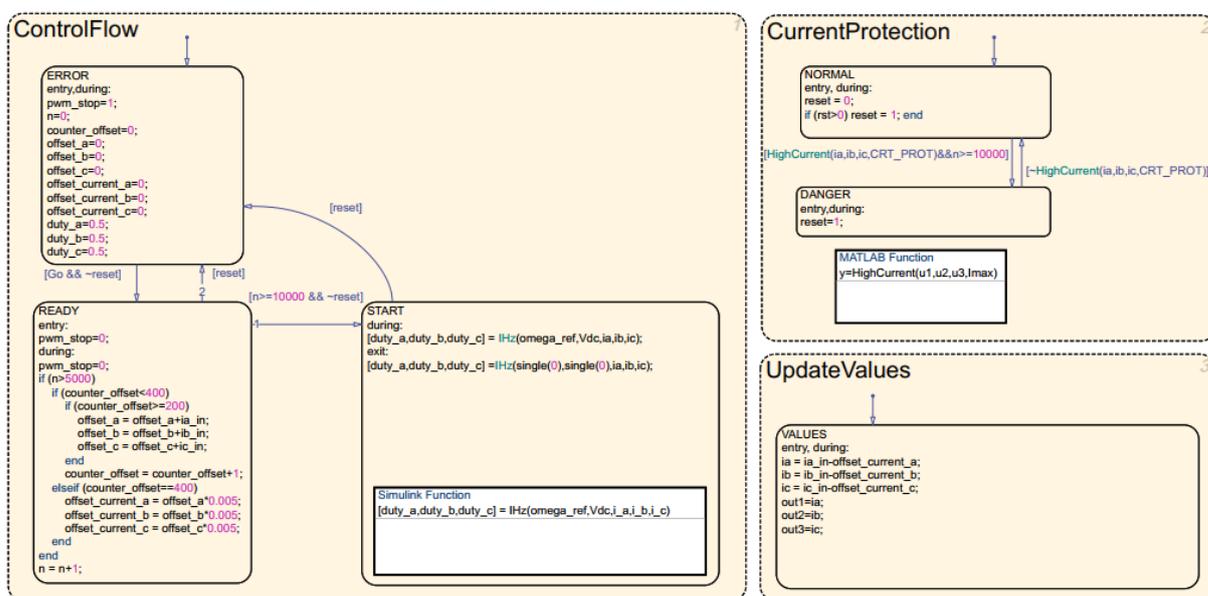


Рисунок 2 – Конечные автоматы в Stateflow

1. Control Flow: присутствует три внутренних состояния. Первое, состояние ОШИБКА (ERROR), в которой инициализируются внутренние переменные, а порты ШИМ установлены на логику с тремя состояниями (выключено). Когда команда Go устанавливается на уровень логической 1 (HIGH), она переходит в состояние ГОТОВ (READY), когда порты ШИМ активны, конденсаторы в цепи последовательной положительной обратной связи инвертора заряжаются, и измеряются смещения АЦП. После некоторых циклов состояние START становится активным, и частотно-токовое управление выполняется в функции Simulink. Всякий раз, когда команда сброса на уровне логической 1 (HIGH), состояние ОШИБКА (ERROR) становится активным, а переменные функции Simulink удаляются.

2. Current Protection: это конечный автомат проверяет интенсивность нарастания токов, чтобы решить, находится ли операция в безопасных пределах или ее следует отключить.

3. Update Values: в этой части смещение вычитается из входных токов. Кроме того, выходные значения диаграммы обновляются.

Компоненты алгоритма управления, называемые INz_model.slx на рисунке 1, должны быть классифицированы как входы, выходы, локальные переменные или глобальные переменные в окне Model Explorer.

- В модели INz входные и выходные сигналы названы и заданы как один тип данных. Входными сигналами блока являются *iabcs* для измеренных токов статора, V_{dc} – уровень напряжения звена постоянного тока, *posCount* – счетчик положения от энкодера, *omega_ref_rpm* – опорный сигнал угловой скорости и *commands* – команды запуска и сброса. Выходными сигналами являются *duty_abc* – коэффициенты заполнения 3-фазного ШИМ-инвертора, *rwm_stop* – отключение вентиля инвертора и три общих выхода для контроля других переменных.

- На диаграмме Stateflow рисунка 2, переменные сконфигурированы как входы, выходы или локальные переменные с их начальными условиями. Все они рассматриваются как один тип данных.

- В функции Simulink, соответствующие сигналы названы и сконфигурированы для ExportedGlobal для обработки в качестве глобальных переменных во время генерации кода. Значения некоторых блоков Constant и Gain вызываются из рабочей области Matlab. Важно помнить, что все переменные и операции должны работать как один тип, который эквивалентен типу MCU float32.

Последний этап тестирования заключается в запуске моделирования алгоритма управления и моделируемой установки (тестирование модели в контуре управления – MIL). Как инвертор, так и двигатель считаются компонентом системы, где сигналы ШИМ являются входами, а токи статора – выходами. Положение ротора, заданное датчиком, не будет использоваться для I-Hz управления.

В имитационной модели Simulink (см. рисунок 3) инвертор представлен усилителем сигналов ШИМ, а двигатель представлен блоком синхронной машины с постоянными магнитами [5], в котором установлены данные изготовителя реального двигателя. ШИМ-сигналы выдаются периферийным устройством таймера MCU, сравнивая коэффициент заполнения с восходящей-нисходящей пилообразной несущей на частоте дискретизации 4 [кГц] [6, 7]. Это периферийное устройство изображается при моделировании компараторами и повторяющейся последовательной волной. Инвертор усиливает сигнал до уровня $0-V_{dc}$ для подачи напряжения на двигатель. После того, как установка смоделирована, алгоритм управления тестируется в контуре управления (MIL) с использованием Model Reference block, вызывающий внешнюю модель Simulink и запускается с частотой дискретизации генератора тактовых сигналов 4 [кГц]. Важно, чтобы частота

дискретизации моделирования была выше, чем у генератора тактовых сигналов, чтобы обеспечить подход к непрерывному управлению установкой.

В таблице 1 приведены данные для моделирования установки.

Таблица 1 – Параметры моделирования

Параметры модели PMSM:	Константы системы управления:
1. 4 пары полюсов в каждой из 3 фаз 2. Константа напряжения (K_E): 3,15 [U_{RMS}/K_{rpm}] 3. Момент инерции (J_R): 0,06 [$кг \times см^2$] 4. Активное сопротивление обмотки (R_{u-v}): 0,5 [$Ом$] 5. Индуктивное сопротивление обмотки (L_{u-v}): 2,2 [$мГн$]	1. $K_{p_d}=0,4$ 2. $K_{i_q}=80$ 3. $K_{p_d}=0,4$ 4. $K_{i_q}=80$ 5. $V_{DC}=24$ [B]

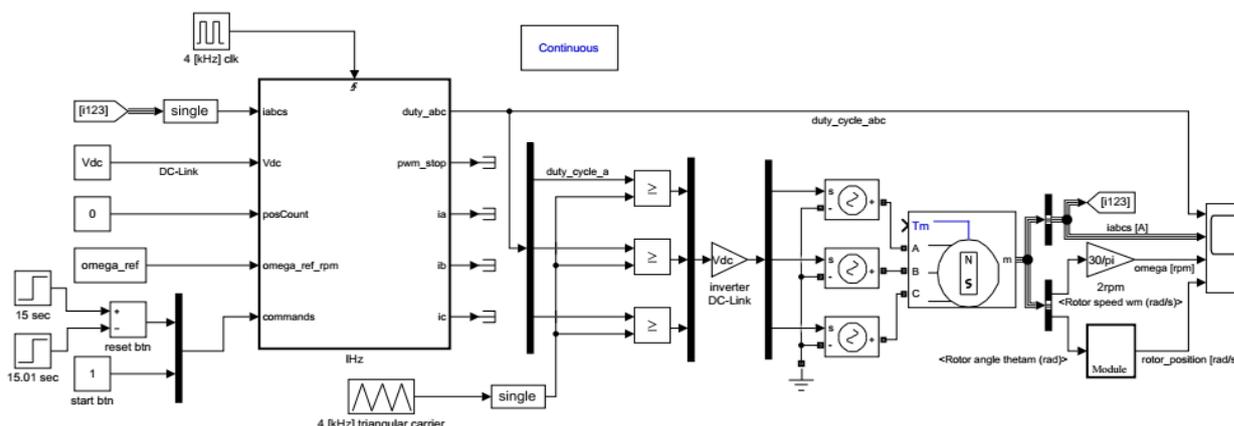


Рисунок 3 – Тестирование в контуре (MIL)

Как только среда моделирования установлена, соответствующие сигналы выбираются для визуализации в области видимости. Это коэффициент заполнения, токи статора, скорость вращения и положение ротора. Процесс моделирования проводится дважды или трижды, чтобы проверить, соответствуют ли показания двигателя заданным исходным. Визуализация сигнала показана на рисунках 4-5 ниже для набора исходных сигналов: $i_d = 0,8$ [A] и $w_{mec} = 400$ [$об/мин$]; $i_d = 1,2$ [A] и $w_{mec} = 600$ [$об/мин$].

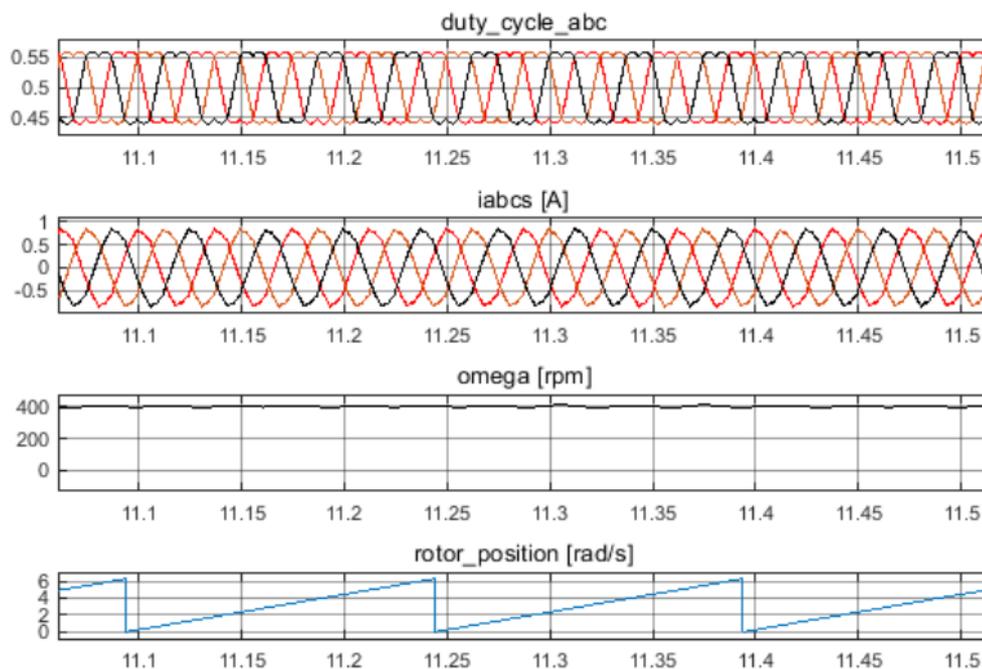


Рисунок 4 – Сигналы в установившемся режиме $i_d = 0,8$ [A] и $w_{\text{mec}} = 400$ [об/мин]

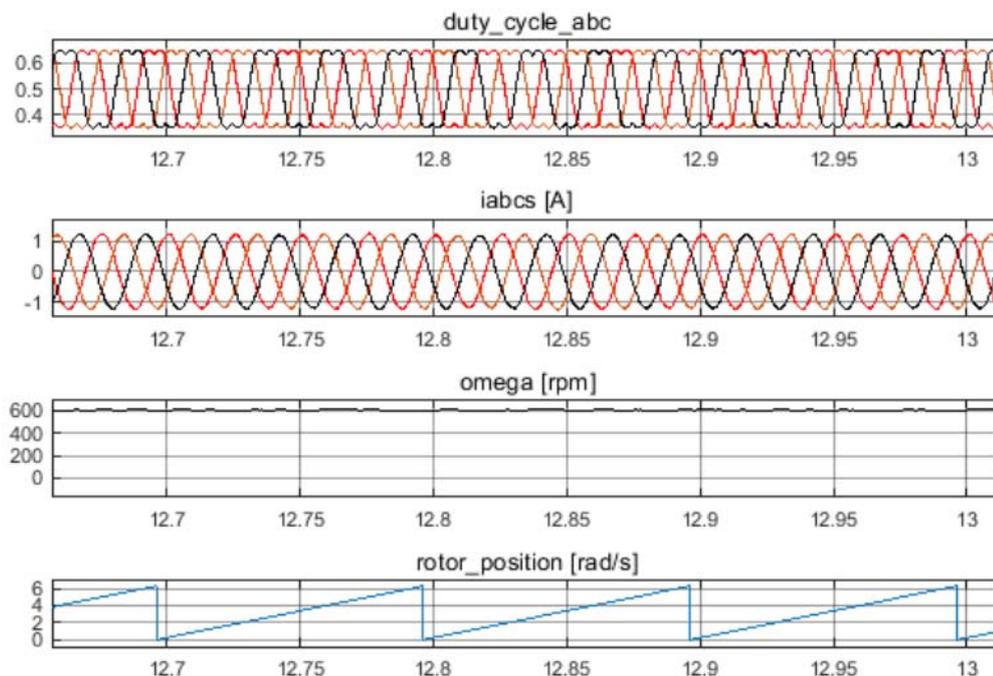


Рисунок 5 – Сигналы в установившемся режиме $i_d = 1,2$ [A] и $w_{\text{mec}} = 600$ [об/мин]

Результаты моделирования показывают, как текущие значения амплитуды и скорости вращения ротора в установившемся режиме соответствуют исходным i_d и w_{mec} . Теперь, когда обеспечена правильная работа логики, стали возможными дальнейшая генерация и интеграция в микроконтроллер алгоритма управления двигателя с помощью инструментов STM32 Embedded Target. Данный пакет использует блоки Simulink для представления периферийных устройств MCU [8-10].

Литература

1. Model-Based Design // MathWorks URL: mathworks.com/help/simulink/gs/model-based-design.html.
2. Терехов В.М., Осипов О.И. Системы управления электроприводов. М.: Издательский центр Академия, 2006. 304 с.
3. Stateflow: Model and simulate decision logic using state machines and flow charts // MathWorks URL: uk.mathworks.com/products/simpower.html.
4. Model and simulate electrical power systems // MathWorks. URL: mathworks.com/help/physmod/sps/powersys/ref/permanentmagnetsynchronousmachine.html.
5. Кабышев О.А., Кабышев А.М. Разработка и компьютерное моделирование регулятора переменного напряжения // Инженерный вестник Дона. 2019. №1. URL: ivdon.ru/ru/magazine/archive/n1y2019/5683
6. Permanent Magnet Synchronous Machine // MathWorks. URL: mathworks.com/help/physmod/sps/powersys/ref/permanentmagnetsynchronousmachine.html.
7. Кулаев Н.А., Зубков П.С. Математическое описание управления синхронным двигателем с постоянными магнитами для аппаратной реализации на ПЛИС // Инженерный вестник Дона. 2019. №6. URL: ivdon.ru/ru/magazine/archive/n6y2019/6039



8. Wei Wu Model-Based Design for Effective Control System Development. IGI Global, 2017. 299 p.

9. Gabriela Nicolescu, Pieter J. Mosterman. Model-Based Design for Embedded Systems. CRC Press, 2009. 766 p.

10. Moreira C.O. Rapid Control Prototyping Using an STM32 Microcontroller: Bachelor's degree Clausthal-Zellerfeld, Germany, 2015. 85 p.

References

1. Model-Based Design. URL: mathworks.com/help/simulink/gs/model-based-design.html.

2. Terekhov V.M., Osipov O.I. Sistemy upravleniya elektroprivodov. [Electric drive control systems]. M.: Izdatel'skiy tsentr Akademiya, 2006. 304 p.

3. Stateflow: Model and simulate decision logic using state machines and flow charts. URL: uk.mathworks.com/products/simpower.html.

4. Model and simulate electrical power systems. URL: mathworks.com/help/physmod/sps/powersys/ref/permanentmagnetsynchronousmachine.html.

5. Kabyshev O.A., Kabyshev A.M. Inzenernyj vestnik Dona, 2019. №1. URL: ivdon.ru/ru/magazine/archive/n1y2019/5683

6. Permanent Magnet Synchronous Machine. URL: mathworks.com/help/physmod/sps/powersys/ref/permanentmagnetsynchronousmachine.html.

7. Kulayev N.A., Zubkov P.S. Inzenernyj vestnik Dona, 2019. №6. URL: ivdon.ru/ru/magazine/archive/n6y2019/6039

8. Wei Wu Model-Based Design for Effective Control System Development. IGI Global, 2017. 299 p.

9. Gabriela Nicolescu, Pieter J. Mosterman. Model-Based Design for Embedded Systems. CRC Press, 2009. 766 p.



10. Moreira C.O. Rapid Control Prototyping Using an STM32 Microcontroller: Bachelor's degree Clausthal-Zellerfeld, Germany, 2015. 85 p.