УДК 62-83:004.4(045)
DOI: 10.24160/1993-6982-2017-4-83-91

Разработка программного обеспечения для управления электроприводом в технологической системе с применением метода модельно-ориентированного программирования

И.С. Полющенков

Рассмотрена разработка программного обеспечения для управления электроприводом в технологической системе с применением метода модельно-ориентированного программирования, базирующегося на использовании средств компьютерной математики и моделирования для разработки и отладки программного обеспечения микропроцессорных систем управления. Приведены компьютерные модели, которые позволяют осуществить прием и передачу данных по последовательным интерфейсам — шинам *I2C* и *CAN* и по интерфейсу последовательного асинхронного приемопередатчика. Разработанное программное обеспечение использовано в микропроцессорной системе управления электропривода для реализации интерактивного управления, обмена информацией с системой управления верхнего уровня, при настройке параметров корректирующих и управляющих элементов, а также для отслеживания параметров движения и контрольной информации. Показано, что при использовании метода модельно-ориентированного программирования при разработке программного обеспечения для реализации обмена данными требуется только формировать пакеты при передаче данных и интерпретировать их при приеме. Правила формирования и интерпретации сообщений устанавливаются протоколом обмена. Формирование пакетов осуществляется путем группировки параметров и переменных различных числовых форматов в заданной последовательности. Интерпретация принятых сообщений заключается в выделении из них полей заданных числовых форматов. При этом алгоритм использования сети и доступа устройств к ней, а также наблюдение за ее исправностью происходят путем использования средств модельно-ориентированного программирования. Указанные обстоятельства позволяют значительно упростить процесс разработки. Предложенная графическая форма архитектуры программного обеспечения может быть использована при проектировании различных технических систем.

Ключевые слова: микропроцессорное управление, математическое моделирование, модельно-ориентированное программирование, цифровые интерфейсы, интерактивное управление.

Для цитирования: Полющенков И.С. Разработка программного обеспечения для управления электроприводом в технологической системе с применением метода модельно-ориентированного программирования // Вестник МЭИ. 2017. № 4. С. 83—91. DOI: 10.24160/1993-6982-2017-4-83-91.

The Development of Software for Electric Drive Control in a Process System Using the Model-Oriented Programming Technique

I.S. Polyushchenkov

The article considers the development of software for control of electric drive in a process system by applying the model-oriented programming technique. The model-oriented programming technique is based on using computer mathematics and modeling tools to develop and debug the software of microprocessor control systems. Computer models featuring the ability of receiving and transmitting data via the digital *I2C* and *CAN* bus serial interfaces and via a serial asynchronous transceiver interface are described. The developed software is used in an electric drive's microprocessor control system for implementing interactive control, for data exchange with a higher-level control system, for adjusting the parameters of corrective and control elements, and for monitoring the motion parameters and control information. It is shown that the tasks that have to be solved for setting up data exchange in elaborating software through the use of the model-oriented programming boil down to shaping data packets in transmitting information and interpreting the data packets in receiving them. The message shaping and interpreting rules are set by the communication protocol. Data packets are shaped by grouping parameters and variables in different numerical formats in a predetermined sequence. The received messages are interpreted by identifying the fields of predetermined numerical formats contained in them. The algorithm governing the use of network resources and the access of devices to it, and the features monitoring the network serviceability are all set up by using the tools available in the model-oriented programming technique. Owing to the above-mentioned circumstances, the design process is simplified to a considerable extent. The proposed graphic form of the software architecture can be used in designing various technical systems.

Key words: microprocessor control, mathematical modeling, model-oriented programming, digital interfaces, interactive control.

For citation: Polyushchenkov I.S. The Development of Software for Electric Drive Control in a Process System Using the Model-Oriented Programming Technique. MPEI Vestnik. 2017; 4: 83—91. (in Russian). DOI: 10.24160/1993-6982-2017-4-83-91.

Введение

Современный электропривод имеет в своем составе электрическую машину, силовой преобразователь, электронную управляющую часть, различные передаточные механизмы и средства для сопряжения с управляющими и измерительными устройствами, устройствами индикации, которые могут быть разнесены в пространстве. Если электропривод работает в составе автоматизированной технологической системы, то обмен данными между внешними устройствами и электроприводом, а также между электроприводом и системой управления верхнего уровня обеспечивает целостность технологической системы, единство управления и сбора информации о ее состоянии и параметрах.

На рис. 1 показана функциональная схема управления электроприводом в составе анализируемой технологической системы. Особенности разработки и функционирования данного электропривода (ЭП) опубликованы в [1], там же представлены результаты проектирования микропроцессорной системы управления и ее программного обеспечения с применением средств модельно-ориентированного программирования. В настоящей работе подробно рассмотрена реализация коммуникационной подсистемы электропривода, а именно решение задачи реализации обмена данными по цифровым последовательным интерфейсам между электроприводом и периферийными устройствами. Также показано, что применение средств модельно-ориентированного программирования позволяет упростить решение указанных задач проектирования.

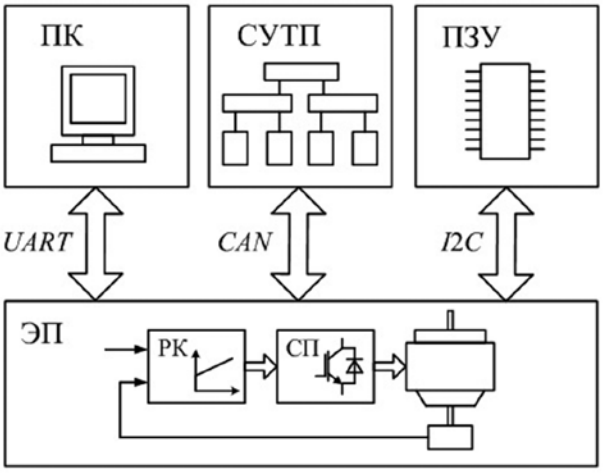


Рис. 1. Функциональная схема управления электроприводом в составе технологической системы

Содержание и методология разработки

Рассматриваемая система электропривода (см. рис. 1) имеет регулятор координат (РК), который управляет силовым преобразователем (СП) и электрическим двигателем в соответствии с задающим воздействием и сигналами обратных связей. В системе управления электропривода использован микроконтроллер STM32F407. В качестве системы управления верхнего уровня в зависимости от условий применения электропривода могут быть взяты персональный компьютер (ПК) и специальное управляющее устройство — схема управления технологическим процессом (СУТП). Также предусмотрен обмен данными с микросхемой постоянного запоминающего устройства для сохранения и восстановления параметров и настроек электропривода.

В устройствах на основе электронной техники обычно имеются встроенные модули сопряжения с промышленными сетями по различным интерфейсам. Наиболее простыми и распространенными из них считаются последовательные интерфейсы, по которым данные передаются и принимаются последовательно в виде структурированных пакетов. Соответствующими стандартами устанавливаются физические и электрические свойства сигналов, скорость обмена, правила доступа устройств к сети, алгоритмы передачи информации и использования сети, функции наблюдения за исправностью [2]. Приводимые в стандартах протоколы обмена устанавливают единообразный порядок преобразования сообщений.

Для выполнения указанных стандартов и алгоритмов используются аппаратные средства и программное обеспечение. При отсутствии достаточного опыта у разработчика могут возникнуть трудности с проектированием программного обеспечения при использовании традиционных средств разработки на основе языков программирования.

Метод модельно-ориентированного программирования помогает избежать подобных трудностей, так как в отличие от традиционных средств разработки он основан на использовании средств компьютерного моделирования, например системы компьютерной математики Matlab [3]. Метод позволяет разрабатывать программное обеспечение для микропроцессорных систем в виде графических моделей. Например, для программирования микроконтроллеров семейства STM32 в Matlab может быть интегрирована библиотека модельно-ориентированного программирования *Waijung Blockset* [4]. Модельные блоки этой библиотеки позволяют задействовать в компьютерной модели аппаратные средства микроконтроллера для его сопряжения с физическим объектом. Кроме библиотеки модельно-ориентированного программирования, при разработке программного обеспечения могут быть использованы другие средства Matlab, а именно: блоки *Simulink*, диаграммы состояний *StateFlow*, *S*-функции на языках программирования Matlab [5] и *C*-функции на языке программирования используемого микроконтроллера. Модель программного обеспечения, имеющая средства для взаимодействия с аппаратной частью, является исполняемой моделью, которая используется для генерирования программного кода при переносе в микропроцессорную систему.

Типы используемых цифровых интерфейсов, указанные на схеме рис. 1, выбраны с учетом топологий и про-

токолов соответствующих сетей, а также конструкции и технических характеристик сопрягаемых устройств.

Интерфейс асинхронного последовательного приемопередатчика *UART* (Universal Asynchronous Receive-Transmitter) [6] *RS232* распространен в системах управления электроприводами и автоматизации, хотя по одной линии связи он позволяет соединить только два устройства. По такому интерфейсу (см. рис. 1) осуществляется обмен данными между персональным компьютером и электроприводом. Специально разработанное приложение для Windows позволяет настраивать электропривод и анализировать его работу, отслеживая координаты движения и контрольную информацию. Подключение электропривода к ПК предусмотрено по последовательному или по USB-порту при использовании преобразователя интерфейсов *UART* – USB. Также интерфейс *UART* может быть использован для управления непосредственно в технологических системах, если это предусмотрено организацией сетей.

Схема управления технологическим процессом СУТП в общем случае взаимодействует с несколькими устройствами (электроприводами и датчиками), соединенными последовательно. Поэтому для реализации этой сети использована интерфейсная шина *CAN* (Controller Area Network). Кроме того, интерфейс типа *CAN* обеспечивает высокую надежность связи.

Хранение параметров и настроек электропривода в соответствии со схемой, показанной на рис. 1, осуществляется в памяти микросхемы ПЗУ *AT24C32* (последовательное *EEPROM* 4096 байт), имеющей интерфейс типа *I2C* (Inter-Integrated Circuit). Ведущим устройством на шине *I2C* является микроконтроллер в системе управления электроприводом ЭП, а ведомым устройством — микросхема ПЗУ. Также шина *I2C* кроме доступа к ПЗУ часто используется для подключения различных устройств, например датчиков и индикаторов.

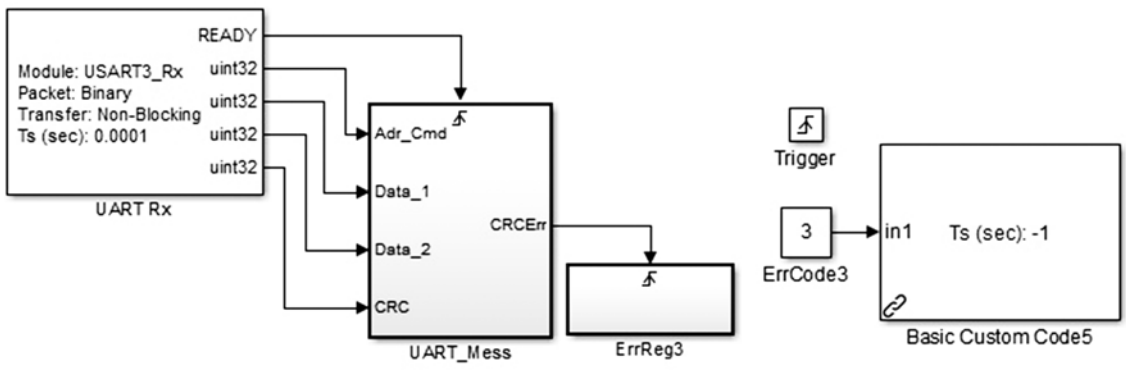
Рассмотрим разработку программного обеспечения информационной подсистемы электропривода с использованием библиотеки модельно-ориентированного программирования *Waijung Blockset*. Указанная библиотека имеет набор модельных блоков для реализации обмена данными по различным цифровым интерфейсам. Алгоритмы доступа к сетям, обнаружение и захват информационных пакетов при приеме и их передача обеспечиваются с помощью стандартных модельных блоков, которые настраиваются посредством меню. От программного обеспечения разработчика требуется обрабатывать принятые сообщения по правилам протокола и формировать исходящие сообщения заданного формата.

В исполняемой модели электропривода присутствуют подсистемы для обмена данными по интерфейсам, показанным на рис. 1.

Последовательный асинхронный приемопередатчик передает и принимает данные в виде пакетов по линиям *Tx* и *Rx* соответственно по одному биту в равные промежутки времени, что характеризуется скоростью соединения. Подсистема для приема данных с использованием *UART*, их обработки и интерпретации показана на рис. 2.

Обработчик приемника *UART Rx* предоставляет доступ к линии связи и отслеживает принятые данные, детектирует в них пакеты заданной длины по признакам их начала и конца, а также выделяет из пакетов блоки формата uint32. При обнаружении пакета устанавливается программный флаг *Ready* (выход блока *UART_Rx*), что приводит к вызову обработчика сообщений — триггерной подсистемы *UART_Mess*, схема которой приведена на рис. 3. Согласно протоколу обмена данными по асинхронному интерфейсу, информационный пакет имеет фиксированную длину и состоит из нескольких полей: идентификатора устройства *Adr*, кода операции *Cmd*, двух полей данных *Data_1* и *Data_2* и поля контрольной суммы, которая вычисляется по алгоритму циклического избыточного кода *CRC32* (Cyclic Redundancy Check). Обработчик сообщений выделяет поле кода операции и проверяет совпадение поля идентификатора устройства *Adr*, принятого в сообщении, с идентификатором *ID*, используя подсистему *IfActionSubsystem1*.

Также обработчик проверяет целостность данных в пакете, сравнивая контрольную сумму, принятую в сообщении, и вычисленную блоком *STM32F4XX CRC32*.



а *б*

Рис. 2. Подсистема исполняемой модели для приема данных по асинхронному последовательному интерфейсу: *а* — подсистема приемника данных; *б* — блок для регистрации нарушения целостности данных (подсистема *ErrReg3*)

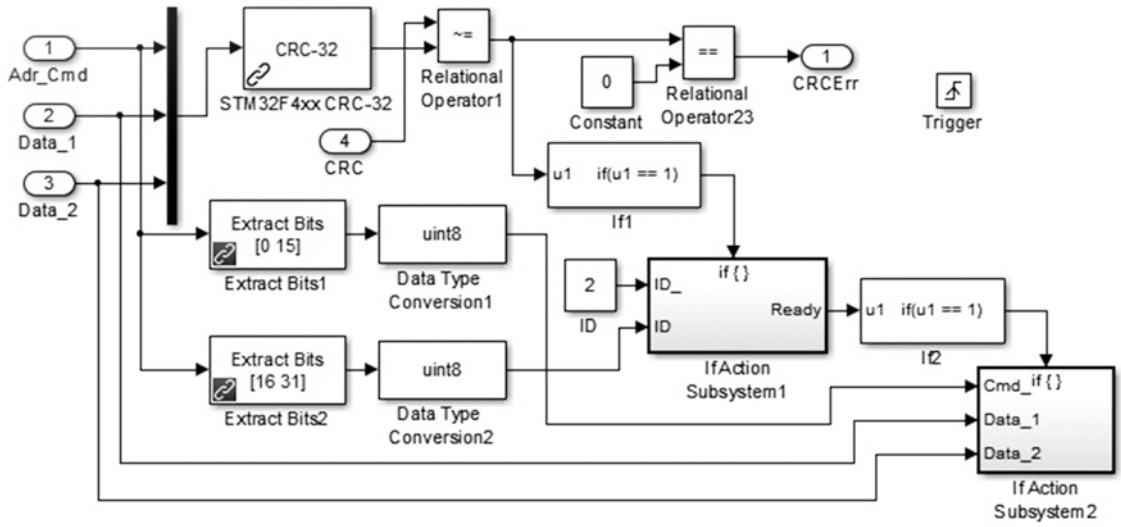


Рис. 3. Подсистема обработчика принятых данных (подсистема *UART_Mess*)

При нарушении целостности данных путем установки программного флага (выход *CRCErr*) вызывается триггерная подсистема *ErrReg3*, в которой с помощью блока *Basic Custom Code5* (см. рис. 2, *б*) в исполняемую модель интегрирована пользовательская функция журнала регистрации неисправностей. При совпадении идентификаторов устройства и целостности данных вызывается подсистема интерпретатора сообщений *IfActionSubsystem2*, присваиваивающая параметрам системы принятые в сообщении значения.

На рис. 4 изображены варианты подсистем исполняемой модели электропривода для передачи исходящих сообщений с использованием асинхронного приемопередатчика. В обоих вариантах формирование сообщений и доступ к линии связи осуществляется управляемыми подсистемами *UART_Transmit*. Для управления передачей данных в виде потока (рис. 4, *а*) используется таймер, прерывания от которого генерируются с временным интервалом, задаваемым с помощью блока таймера *Timer IRQ1*. Этот вариант подсистемы был использован при разработке и отладке системы регулирования координат электропривода. В схеме, показанной на рис. 4, *б*, подсистема *UART_Transmit* является триггерной и управляется программным флагом *UART Call*.

В этом случае реализуется передача данных в форме запросов и ответов. Указанный программный флаг устанавливается для ответа на принятые сообщения, а также для передачи сообщений при событиях, требующих реакции системы управления верхнего уровня или оператора.

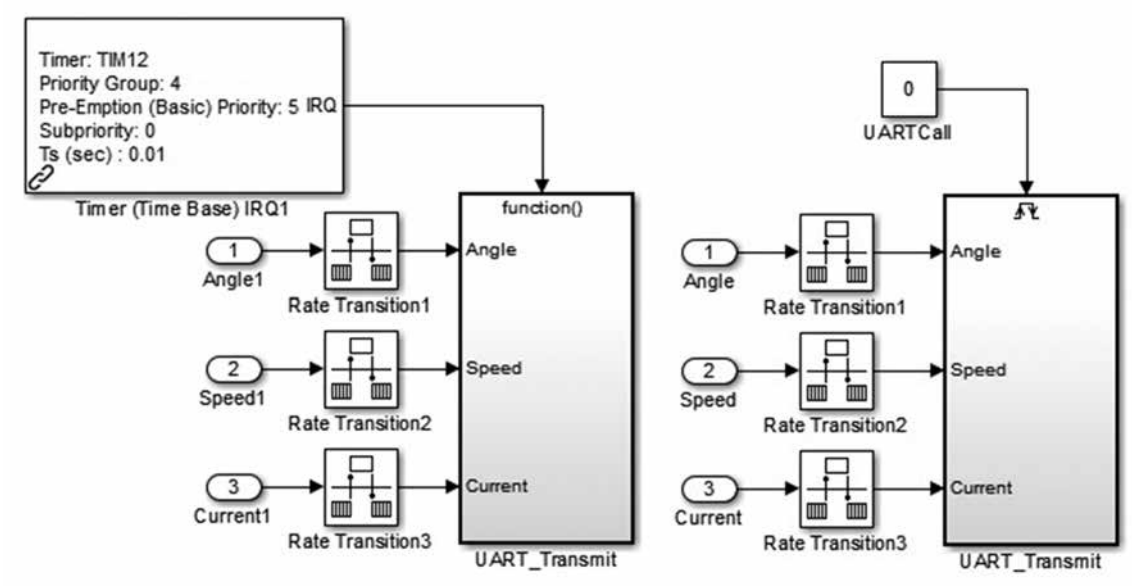
На рис. 5 продемонстрирована схема подсистемы *UART_Transmit* для передачи данных с использованием асинхронного интерфейса. Для доступа к линии связи использован модельный блок *UART Tx*. При формировании исходящих сообщений требуется группировать данные в поля и блоки, формат и последовательность которых установлены протоколом. Набор пересылаемых переменных (координаты движения и регуляторов, служебная информация) выбирается в зависимости от идентификатора данных *Status*, который задается при управлении электроприводом. Для доступа к координатам движения электропривода (перемещению, скорости, току электрического двигателя) использованы входы *Angle*, *Speed* и *Current* блока пользовательской *C*-функции *Basic Custom Code*, которая реализует протокол передачи данных. Доступ к другим параметрам и коэффициентам осуществляется непосредственно в *C*-функции.

Также в исходящем сообщении кроме полей данных имеются поля идентификатора адресуемого абонента (*ID1*), флагов состояния электропривода (*Sost*), идентификатора данных (*Status*) и контрольной суммы. Указанные поля и поля данных в заданной последовательности объединяются в информационный пакет, который передается назначенному абоненту модельным блоком *UART Tx* по асинхронному последовательному интерфейсу.

При разработке и отладке программного обеспечения для обмена данными по интерфейсу *UART* использовалось приложение Advanced Serial Port Monitor. Для интерактивного управления и настройки электропривода, а также отслеживания координат движения и анализа его работы создано специальное приложение, экранная форма которого показана на рис. 6.

Подсистемы исполняемой модели для приема сообщений по шине *CAN* и их обработки представлены на рис. 7. Доступ к сети обеспечивается блоком приемника *CAN Receive* (рис. 7, *а*).

При получении сообщения устанавливается программный флаг *Msg Pending* блока CAN *Receive*, что приводит к вызову подсистемы для обработки сообщений *CAN_Mess*. Проверка целостности данных выполняется блоком *CAN Receive* автоматически. Подсистема обработчика сообщений показана на рис. 7, *б*. Обработчик проверяет совпадение поля адреса в сообщении *ID* и адреса абонента *CAN_ID*, а также равенство длины сообщения *DLC* и заданной длины *CAN_DLC*, установленной прото-



а *б*

Рис. 4. Варианты подсистем исполняемой модели для передачи данных:
а — в виде потока данных; *б* — в виде ответов на запросы

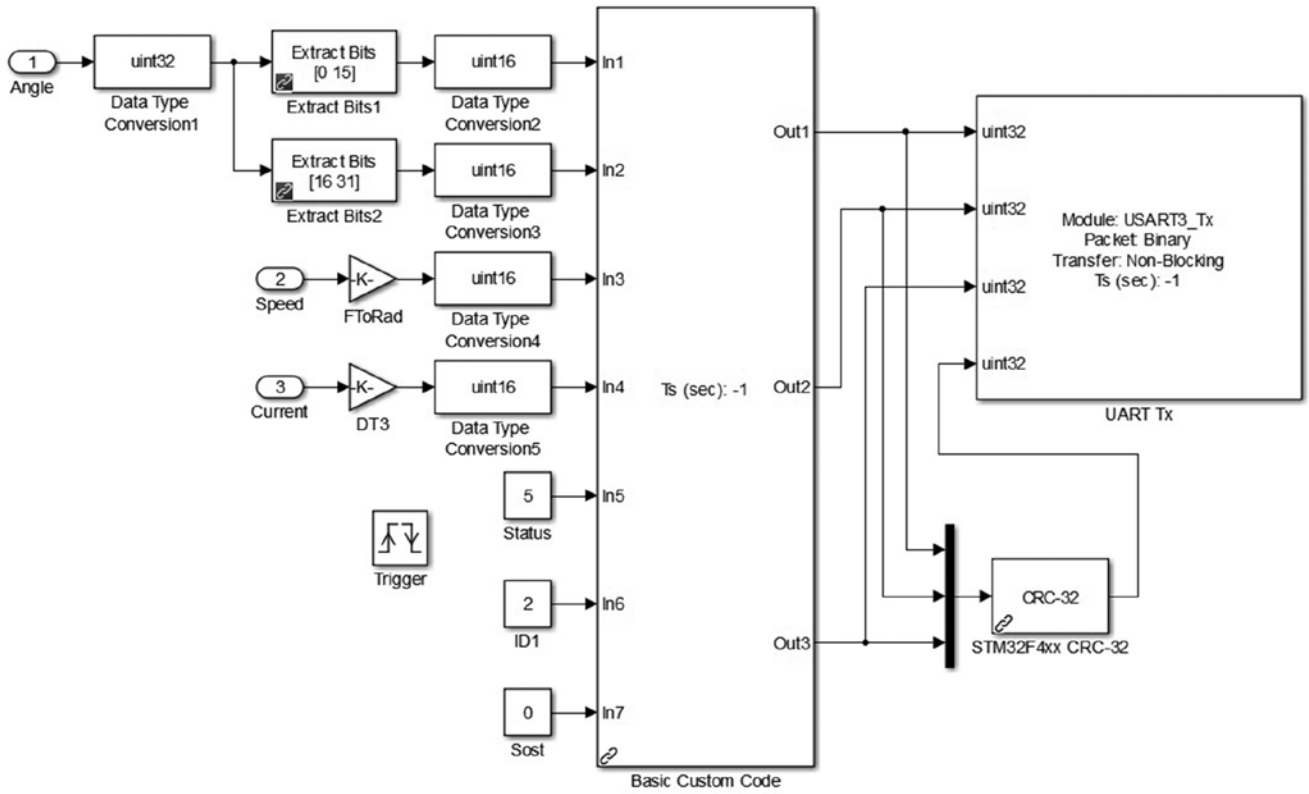


Рис. 5. Подсистема исполняемой модели для передачи данных по *UART*

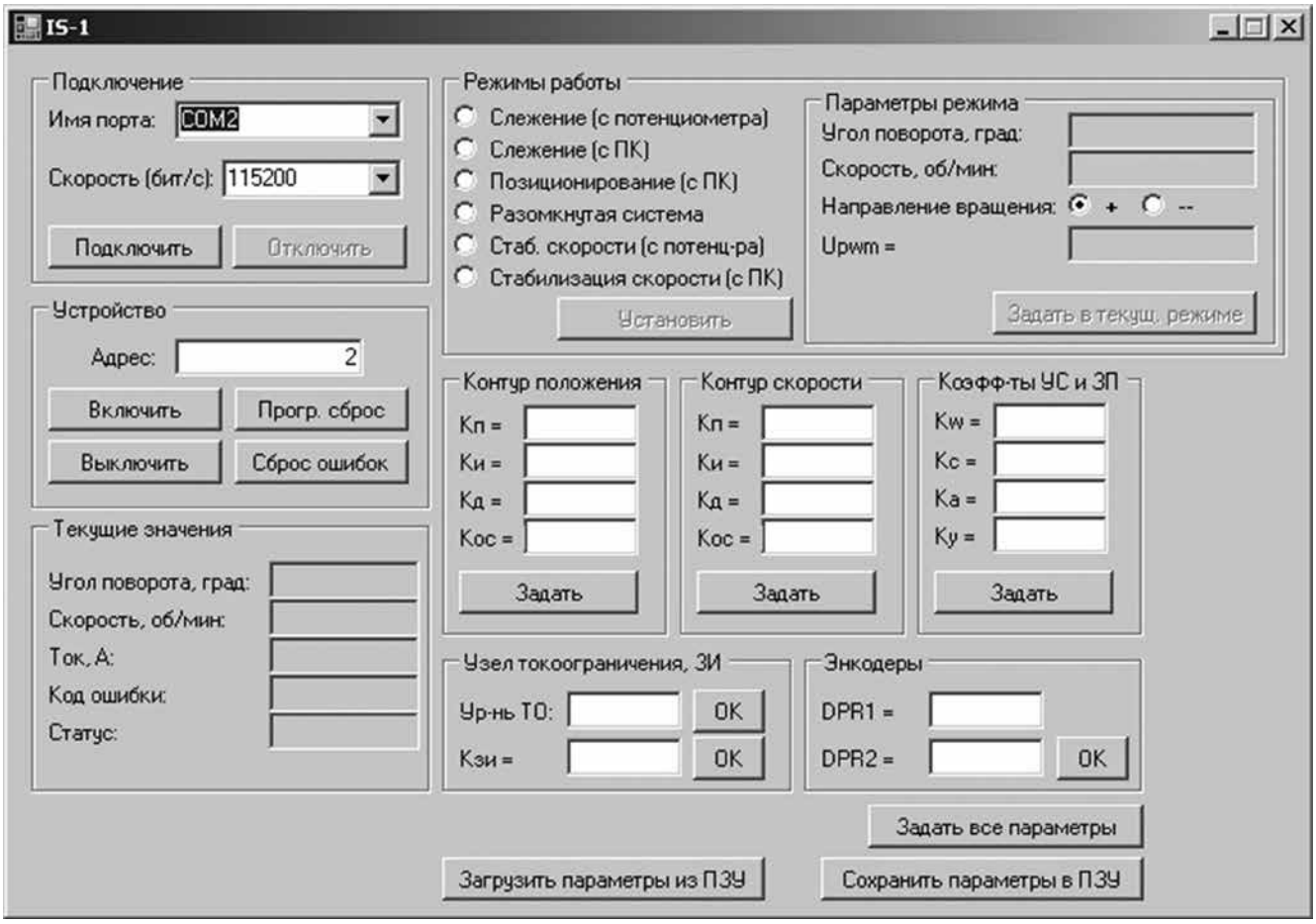
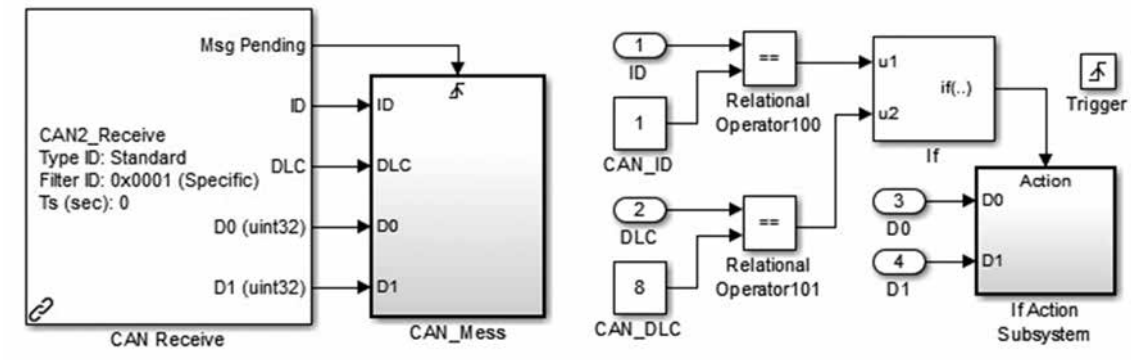


Рис. 6. Экранная форма приложения для управления электроприводом

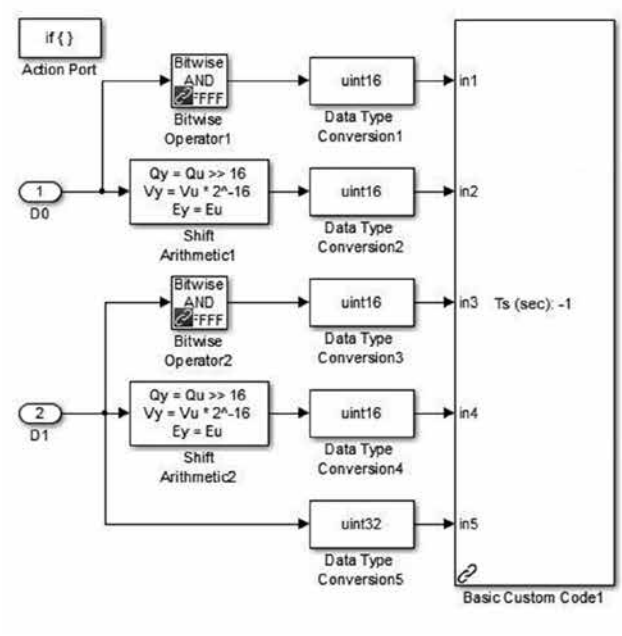


а *б*

Рис. 7. Подсистемы исполняемой модели для приема и обработки данных по шине *CAN*:
а — подсистема приемника данных; *б* — подсистема обработчика данных *CAN_Mess*

колом. При выполнении указанных условий вызывается подсистема *IfActionSubsystem* для интерпретации данных, схема которой показана на рис. 8, *а*.

Блок *Basic Custom Code 1* на рис. 8, *а* интегрирует в исполняемую модель пользовательскую функцию протокола для обработки принятых сообщений. Фрагмент указанной функции приведен на рис. 8, *б*. Из исходных блоков *D*0 и *D*1 при интерпретации выделяются поля кода операции (*in*1), адреса источника сообщения (*in*2) и данных разных числовых форматов (*in*3 — *in*5). Функция протокола по значению кода операции присваивает параметрам системы те значения, которые приняты в сообщении. Например, если в сообщении в соответствии с кодом операции приняты коэффициенты регулятора скорости, то используются два поля блока *D*1 (входы *in*3 и *in*4), содержащие пропорциональный и интегральный коэф-




```
void output_cancustomio1(const uint16_T in1, uint16_T in2,
                         uint16_T in3, uint16_T in4, uint32_T in5)
{
 switch (in1)
 {
  case 0:   // Код операции 0.Включение-выключение.
   {
    BLDC_37_P.StartStop_Value = !(BLDC_37_P.StartStop_Value);
    BLDC_37_P.CANCOP_Value = 0; // Код подтверждения.
    break;
   }
  case 1:  // Код операции 1.Задание скорости.
   {
    BLDC_37_P.Speed_Value = (int16_T)in4;
    BLDC_37_P.CANCOP_Value = 1; // Код подтверждения.
    break;
   }
  case 2:  // Код операции 2.Задание перемещения
   {
    BLDC_37_P.Position_Value = (int32_T)in5;
    BLDC_37_P.CANCOP_Value = 2; // Код подтверждения.
    break;
   }
  case 3:  // Код операции 3.Параметры регулятора скорости
   {
    BLDC_37_P.ProportionalGain_Gain = ((real_T)in3)*0.1;
    BLDC_37_P.IntegralGain_Gain = ((real_T)in4)*0.1;
    BLDC_37_P.CANCOP_Value = 3; // Код подтверждения.
    break;
   }
 }
   // Отправка подтверждения.
 BLDC_37_P.CAN_Mess_Value = !(BLDC_37_P.CAN_Mess_Value);
}
```

а *б*

Рис. 8. Интерпретатор сообщений:
а — подсистема исполняемой модели; *б* — фрагмент функции протокола

фициенты. Если в сообщении принято заданное перемещение, то функцией протокола используется одно поле — вход *in*5.

Подсистемы для формирования и передачи исходящих сообщений по сети *CAN* подобны подсистемам, показанным на рис. 4, 5, но в отличие от них не требуется блок для вычисления контрольной суммы, а для доступа к сети используется блок *CAN Transmit*.

При проектировании подсистемы для обмена данными по сети *CAN* и проверки ее работоспособности использовалось приложение *CANwise* при управлении электроприводом от ПК, а также специальный пульт управления, который конструктивно и программно подобен вычислительному устройству для управления электроприводами антенной установки. При этом обеспечивалась скорость соединения по сети до 1 Мбит/с и интенсивность потока данных до 250 сообщений в секунду.

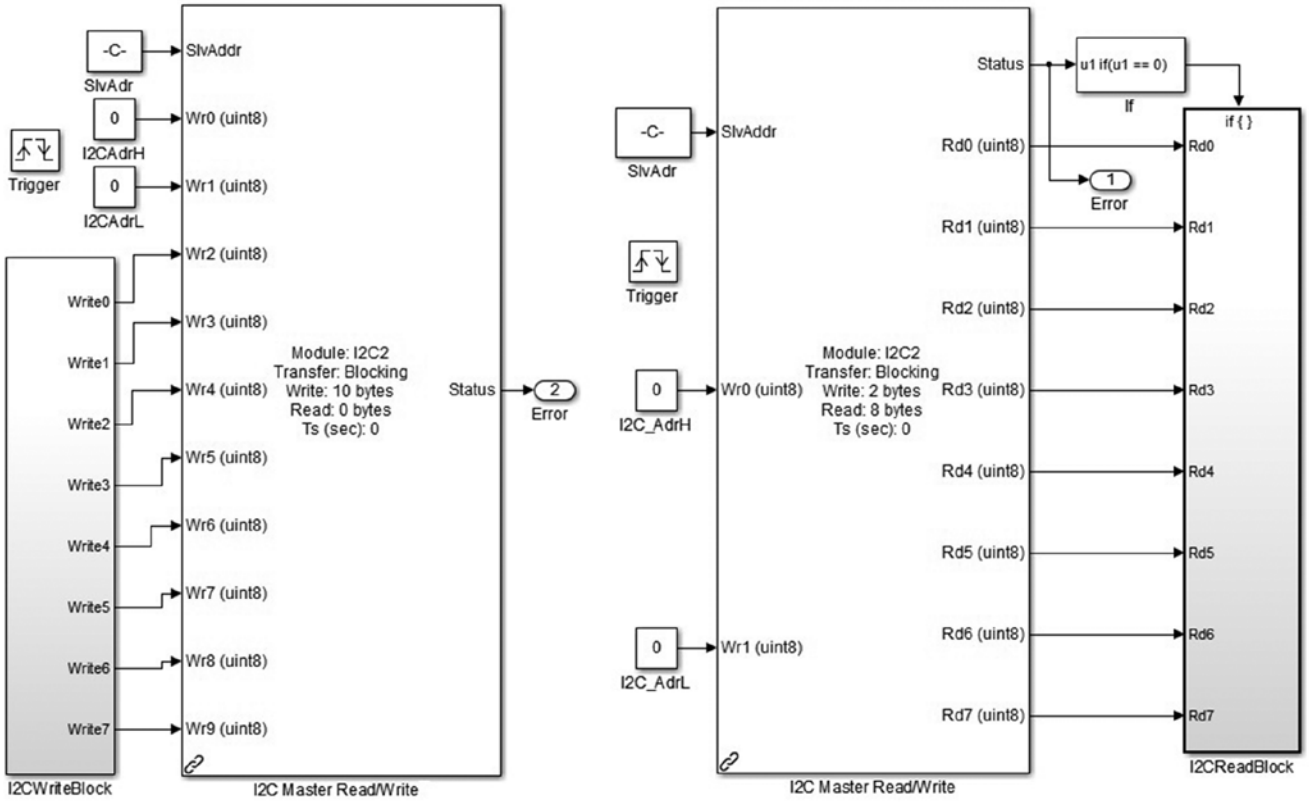
Протоколы обмена по интерфейсу асинхронного приемопередатчика и по сети *CAN* позволяют выбирать режимы работы электропривода (стабилизацию скорости движения, слежение), задавать управляющие воздействия, коэффициенты регуляторов и ограничения тока и скорости движения, отслеживать координаты (ток, скорость, положение), а также контрольную информацию о состоянии электропривода. Указанные возможности обеспечивают интерактивное управление электроприводом и его настройку, а также управление электроприводом в составе технологической системы.

Подсистемы исполняемой модели, выполняющие доступ к микросхеме ПЗУ по шине *I2C*, показаны на рис. 9. Алгоритм коммуникации по сети реализуется модельными блоками *I2C Master Read/Write*, а подключение микросхемы ПЗУ к шине *I2C* осуществляется по двум двунаправленным линиям связи — *SDA* (последовательная линия данных *Serial Data*) и *SCL* (последовательная линия тактирования *Serial Clock*). Скорость обмена данными составляет 100 Кбит/с.

Подсистемы исполняемой модели, показанные на рис. 9, вызываются при установке программных флагов с заданными параметрами — адресом ПЗУ на шине *I2C* (блок *SlvAdr*), а также двухбайтным адресом целевой области памяти *I2CAdrL*, *I2CAdrH* при записи и *I2CAdr_L*, *I2CAdr_H* при чтении данных. Для микросхемы *AT24C32*, использованной для хранения параметров и настроек электропривода, адрес на шине *I2C* равен 0x*A*0.

Информационные пакеты принимаются и передаются побайтно в числовом формате *uint8*. Для формирования информационных пакетов при передаче данных (записи в ПЗУ) подсистема *I2CWriteBlock*, показанная на рис. 9, *а*, разделяет исходные данные на блоки формата *uint*8, которые далее подаются на входы *Wr*2 — *Wr*9 блока *I2C Master Read/Write*. При успешном чтении данных, которое детектируется условным оператором, подсистема *I2CReadBlock*, изображенная на рис. 9, *б*, объединяет принятые блоки *Rd*0 — *Rd*7 в последовательности, заданной протоколом, и присваивает параметрам системы значения, которые прочитаны из памяти ПЗУ.

В качестве служебной информации, используемой системой управления и передаваемой по линиям связи, служат флаги состояния электропривода, а также коды неисправностей, например нарушения целостности принятых данных, превышения уровней ограничений тока и скорости, неисправности датчиков и линий для обмена



а *б*

Рис. 9. Подсистемы исполняемой модели для коммуникации по шине *I2C*:
а — для записи данных в ПЗУ; *б* — для чтения данных из ПЗУ

данными, снижения и превышения напряжения. Указанные события регистрируются в программном журнале.

Дополнительно следует пояснить реализацию специальной функции регистрации неисправностей, интегрированной в исполняемую модель электропривода. Данная функция вызывается при изменении программных флагов, что происходит при возникновении перечисленных выше событий и неисправностей в системе управления электропривода. Например, в схеме на рис. 2 подсистема *ErrReg*3 регистрирует нарушение целостности принятых данных, а код ошибки задается блоком *ErrCode*3. Аналогичные подсистемы с другими значениями кодов неисправностей подключены к выходам различных блоков и подсистем исполняемой модели. Так, в подсистемах на рис. 9 при успешном обращении к ПЗУ выходы *Status* блоков *I2C Master Read/Write* равны нулю, а в случае ошибки они устанавливаются равными единице, что регистрируется в исполняемой модели функцией журнала событий.

Представленная в статье разработка была реализована и испытана в рамках совместного проекта с ООО НПО «Рубикон Инновация». Результаты испытаний показали, что программное обеспечение системы управления электропривода, разработанное с использованием средств модельно-ориентированного программирования, по функциональности не уступает программному обеспечению, созданному с использованием традиционных средств программирования.

Выводы

Результаты разработки позволяют сделать следующие выводы.

Средства модельно-ориентированного программирования являются эффективным инструментом разработки программного обеспечения для информационной подсистемы электропривода, которая обеспечивает обмен данными между электроприводом и внешними устройствами по цифровым последовательным интерфейсам *UART, CAN* и *I2C*.

Разработанная коммуникационная подсистема способна обеспечить работу электропривода, интегрированного в состав технологической системы, содержащей схему управления верхнего уровня, а также персональный компьютер для настройки и сервисных функций.

Литература

1. **Полющенков И.С.** Разработка системы управления электропривода на основе метода модельно-ориентированного программирования // Вестник МЭИ. 2016. № 6. С. 87—95.

2. **Денисенко В.В.** Компьютерное управление технологическим процессом, экспериментом, оборудованием. М.: Горячая линия – Телеком, 2009.

3. **Model-Based** Design [Электрон. ресурс]. www.mathworks.com (дата обращения 01.06.2017)

4. **Waijung** Blockset [Электрон. ресурс]. http://waijung.aimagin.com. (дата обращения 23.06.2017)

5. **Дьяконов В.П.** Matlab 6/6.1/6.5 + Simulink 4/5 в математике и моделировании. М.:СОЛОН-Пресс, 2008.

6**. Хаммел Р.Л.** Последовательная передача данных: руководство программиста. М.: Мир, 1996.

References

1. **Poljushhenkov I.S.** Razrabotka Sistemy Upravlenija Jelektroprivoda na Osnove Metoda Model'no-orientirovannogo Programmirovanija. MPEI Vestnik. 2016;6:87—95. (in Russian).

2. **Denisenko V.V.** Komp'juternoe Upravlenie Tehnologicheskim Processom, Jeksperimentom, Oborudovaniem. M.: Gorjachaja Linija – Telekom, 2009. (in Russian).

3. **Model-Based** Design [Elektron. Resurs]. www.mathworks.com (Data Obrashhenija 01.06.2017)

4. **Waijung** Blockset [Elektron. Resurs]. http://waijung.aimagin.com. (Data Obrashhenija 23.06.2017)

5. **D'jakonov V.P.** Matlab 6/6.1/6.5 + Simulink 4/5 v Matematike i Modelirovanii. M.:SOLON-Press, 2008. (in Russian).

6. **Hammel R.L.** Posledovatel'naja Peredacha Dannyh: Rukovodstvo Programmista. M.: Mir, 1996. (in Russian).

Сведения об авторе

Полющенков Игорь Сергеевич – кандидат технических наук, доцент кафедры электромеханических систем филиала НИУ «МЭИ» в г. Смоленске, e-mail: polyushenckov.igor@yandex.ru

Information about author

Polyushchenkov Igor S. – Ph.D. (Techn.), Assistant Professor of Electromechanical systems Dept., Branch of NRU MPEI in Smolensk, e-mail: polyushenckov.igor@yandex.ru

Статья поступила в редакцию 07.09.2016