Инженерный вестник Дона, №9 (2024)
ivdon.ru/ru/magazine/archive/n9y2024/9487

Разработка автоматизированной системы управления на Modbus-протокол

Н.Б. Лазарева, М.В. Еремеев
Тихоокеанский государственный университет, Хабаровск

Аннотация: В статье представлен анализ протокола Modbus, включая его архитектуру, типы сообщений, особенности реализации и собственный язык программирования. Рассмотрена система управления производственным процессом на основе Modbus.
Ключевые слова: Modbus, программируемый логический контроллер, протокол обмена данными, программирование, промышленная автоматизация, управление процессами, мониторинг, регистры, интеграция систем.

В современном мире автоматизация промышленных процессов играет ключевую роль. Для обеспечения бесперебойной работы систем автоматизации необходимо эффективное средство обмена данными между различными устройствами. У каждого производителя были свои собственные протоколы обмена данными, и их интеграция между различными устройствами была сложной и порой неэффективной. Это приводило к трудностям в обмене информацией, управлении и контроле на производственных объектах, что затрудняло создание автоматизированных систем и их связь между собой. Изобретение основы многоцелевого двунаправленного системного стандарта (multi-purpose bidirectional system standard – Modbus) стандартизировало протокол обмена данными, обеспечивая простой и эффективный способ взаимодействия между различными устройствами в промышленной среде. Modbus стал основой для создания единой системы связи, позволяющей разным устройствам обмениваться информацией независимо от их производителя. Это решение проблемы разнообразия протоколов обмена данными способствовало более эффективной и гибкой автоматизации промышленных систем, а также уменьшило сложность интеграции различных устройств, улучшая общий уровень автоматизации и эффективности производства [1].

Modbus является одним из наиболее распространенных протоколов в области промышленной автоматизации. Он играет важную роль в передаче данных между различными электронными устройствами, такими, как контроллеры, датчики, приводы и другие устройства, используемые в промышленных системах. Modbus обладает богатым набором функций, делающих его универсальным инструментом для промышленной автоматизации:

1. Чтение/запись регистров: позволяет считывать и записывать значения из памяти ведомых устройств.

2. Диагностика: обеспечивает доступ к информации о состоянии устройств, включая коды ошибок.

3. Управление: поддерживает команды управления устройствами, например, запуск/останов двигателя.

Modbus – это протокол прикладного уровня, основанный на архитектуре «ведущий-ведомый» (master-slave), где одно ведущее (master) устройство инициирует запросы к нескольким подчиненным (slave) устройствам. Ведущее устройство может быть программируемым логическим контроллером (ПЛК), системой диспетчерского контроля и управления (supervisory control and data acquisition – SCADA) или другим устройством, способным управлять сетью [2]. Ведомые устройства – это датчики, исполнительные механизмы, контроллеры и другие устройства, которые могут предоставлять или получать данные. Запросы инициирует только ведущее устройство, ведомые устройства могут только на эти запросы отвечать, и не могут самостоятельно начинать передачу данных. Каждый пакет данных, будь то запрос или ответ, начинается с адреса устройства или адреса ведомого устройства, за которым следует код функции, а за ним следуют параметры, определяющие, что запрашивается или предоставляется.

Одним из ключевых моментов в автоматизации производства является гибкая архитектура, способность различных устройств и компонентов обмениваться данными и командами. Modbus предоставляет эффективную, надежную и стандартизированную коммуникацию между различными устройствами в промышленной среде. Это особенно важно в сценариях, где требуется управление, мониторинг и согласованная работа множества разнообразных устройств.

Рассмотрим пример использования модели ведущий-ведомый.

На рисунке 1 показан объект и диспетчерская, в которой установлены автоматизированные рабочие места (АРМ). АРМ соединяются Ethernet-интерфейсом, используя Modbus TCP протокол, с ПЛК. Он и будет выполнять роль ведущего. ПЛК с помощью RS-485 интерфейса и протокола Modbus RTU опрашивает датчики влажности технологического процесса. Датчики являются ведомыми устройствами, отвечают на запрос ПЛК, предоставляя информацию о текущем параметре [3].

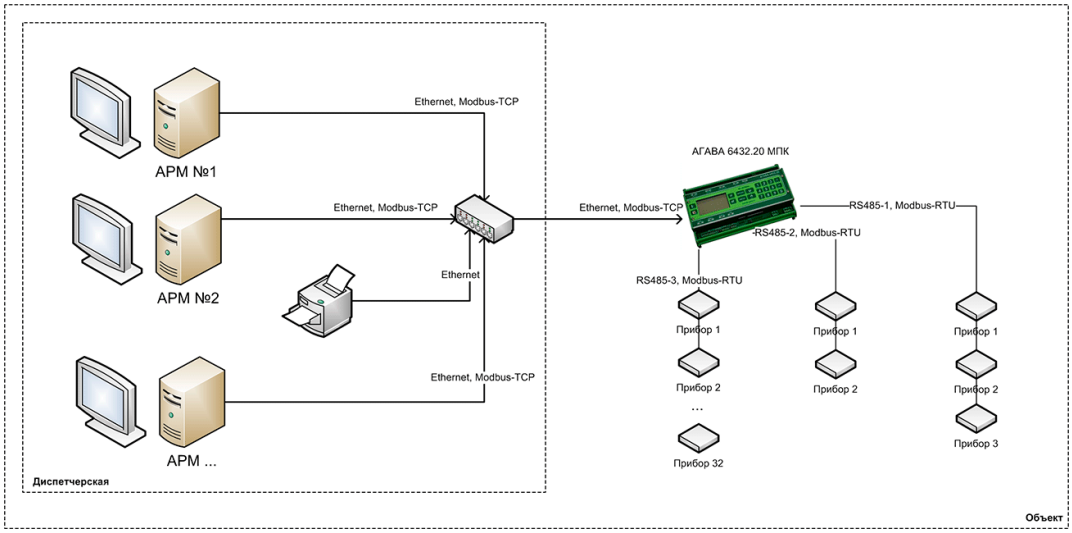

Рис. 1. Условная схема работы протокола

Данные Modbus чаще всего считываются и записываются в виде «регистров», которые представляют собой 16-разрядные фрагменты данных. Регистр может быть 16-разрядным целым числом со знаком или без знака. Если требуется 32-разрядное целое число или значение с плавающей запятой,

то эти значения фактически считываются как пара регистров. Наиболее часто используемый регистр называется «регистром хранения». Регистры хранения могут быть считаны или записаны. Другой возможный тип – «входной регистр», который доступен только для чтения. Исключения из 16-битных регистров составляют катушка и дискретный вход, каждый из которых имеет только 1 бит. Катушки могут считываться или записываться, в то время как дискретные входы доступны только для чтения. Катушки обычно связаны с релейными выходами.

Различные протоколы Modbus:

1. Modbus RTU кодирует данные в двоичном формате, используя временные интервалы в качестве разделителей пакетов. Он имеет ограниченную способность к приспособлению к задержкам и несовместим с модемными линиями. Однако, он имеет меньшие накладные расходы на передачу данных, поскольку сообщения имеют меньший размер [4].

2. Modbus TCP использует структуру пакетов, аналогичную Modbus RTU, для передачи двоично закодированных данных в обычные TCP-пакеты через IP-сети. Поскольку TCP уже обеспечивает собственный контроль целостности, контроль целостности данных, применяемый в Modbus RTU, в нем не используется [5].

Количество возможных подключаемых устройств в сети Modbus RTU (RS-485) не может быть более одного ведущего устройства. Следовательно, если шлюз должен быть настроен как ведущий, то он может быть только один. Нельзя использовать несколько шлюзов для считывания большего количества точек с одного подчиненного устройства Modbus. Несколько шлюзов, настроенных как ведомые устройства, могут находиться в одной сети Modbus RS-485. Если использовать устройства RS-232, всего может быть только два устройства, независимо от того, как они настроены. RS-232 не является многопользовательским. Логически можно обращаться к более

чем 250 устройствам. Однако приемопередатчики RS-485 физически не способны управлять таким количеством устройств. Протокол Modbus утверждает, что ограничение составляет 32 устройства, и большинство приемопередатчиков RS-485 согласятся с этим. Если все устройства в сети оснащены приемопередатчиками с низкой нагрузкой, количество устройств может превышать 32.

Ведомые устройства Modbus можно представить, как таблицу с числами. Ведущее устройство запрашивает у ведомого данные по указанным адресам, и ведомое устройство отправляет эти данные обратно. Также ведущее устройство может сообщить ведомому, какое число записать в таблицу по указанным адресам.

Рассмотрим особенности работы Modbus на примере системы автоматического управления, а именно аппарата воздушного охлаждения антифриза (АВОА).

Аппарат воздушного охлаждения антифриза представляет собой комплексную систему, включающую в себя несколько ключевых компонентов. В основе АВОА лежат блок вентиляторов и блок теплообменных секций, которые состоят из оребрённых труб, наполненных антифризом. Все эти элементы контролируются и управляются с помощью щита автоматизации, где установлен программируемый логический контроллер. Этот ПЛК отвечает за координацию работы всех составляющих системы, обеспечивая их согласованное взаимодействие. Для измерения параметров системы, таких, как температура и давление, используются датчики, которые монтируются на ёмкость с охлаждающей жидкостью. Исполнительные механизмы, включая вентиляторы и клапаны, регулируют потоки воздуха для поддержания оптимальной температуры. ПЛК взаимодействует с этими датчиками и механизмами с помощью протокола Modbus, который обеспечивает надежную передачу данных и команд.

В АВОА применяются два основных интерфейса связи: RS-485 для Modbus RTU, обеспечивающий высокую помехозащищенность и надёжность на больших расстояниях, и Ethernet для Modbus TCP/IP, который обеспечивает высокую скорость передачи данных и легкую интеграцию в существующую сетевую инфраструктуру [6]. Инициализация связи начинается с того, что мастер (ПЛК) отправляет запрос к ведомому устройству (слейву) по заданному адресу. Слейв, например, датчик температуры, отвечает на запрос, отправляя данные обратно мастеру.

Modbus позволяет организовать четкую и надежную передачу данных, используя функции чтения и записи регистров. Это обеспечивает непрерывный мониторинг и управление процессами. Температура измеряется с помощью термопреобразователя сопротивления, а выходной унифицированный сигнал поступает на ПЛК [7]. Если параметр отклоняется от заданного значения, ПЛК активирует исполнительный механизм для регулировки температуры антифриза. Положение регулирующего органа отображается на дистанционном индикаторе, что позволяет операторам контролировать процесс [8]. При помощи Modbus RTU данные о температуре и давлении передаются с датчиков на ПЛК, который их обрабатывает и принимает решение о включении или выключении исполнительных механизмов. Использование Modbus TCP/IP позволяет интегрировать систему в более крупные сети и передавать данные на центральные системы мониторинга и управления. Типы сообщений между ПЛК и датчиками включают функции чтения (Read), когда мастер запрашивает данные у слейва, и записи (Write), когда мастер отправляет команду слейву для выполнения определённого действия, например, включения или отключения вентилятора. Инициализация и настройка системы производится путем подключения датчиков температуры и давления к ПЛК через интерфейс RS-485 для Modbus RTU. Вентиляторы и клапаны также подключаются к ПЛК

через Modbus RTU, и каждому устройству в сети присваивается уникальный адрес Modbus.

В Системе разработки контролера (Controller Development System – CoDeSys) настраивается библиотека Modbus для обработки запросов и ответов [9]. ПЛК запрашивает данные с датчика температуры по заданному адресу, датчик измеряет температуру и отправляет значение обратно ПЛК. Код для получения данных с датчика температуры на языке структурированного текста (Structured Text – ST) выглядит следующим образом:

```
// Чтение температуры
tempSensorValue := MODBUS_READ(ADDRESS_TEMP_SENSOR);
```

Управление исполнительными механизмами осуществляется следующим образом: ПЛК анализирует полученные данные о температуре, если температура превышает 25°C, ПЛК отправляет команду на включение вентилятора по адресу 0x02. Код для включения/отключения вентилятора на языке ST [10]:

```
IF tempSensorValue > 25.0 THEN
   MODBUS_WRITE(ADDRESS_FAN, ON);
ELSE
   MODBUS_WRITE(ADDRESS_FAN, OFF);
END_IF
```

Мониторинг и диагностика работы процесса охлаждения в CoDeSys включают отслеживание текущих значений температуры, давления и состояния вентиляторов в реальном времени. ПЛК периодически опрашивает все устройства, обновляя данные на экране оператора. При отсутствии ответа от устройства или получении некорректных данных, ПЛК регистрирует ошибку и может отправить уведомление оператору или включить аварийный режим. CoDeSys поддерживает функции ведения журналов событий и построения графиков, что помогает в анализе работы системы и выявлении возможных сбоев.

Код диагностики ошибок протокола на языке ST:

```
PROGRAM ErrorHandling
VAR
   errorCode: INT;
   errorMessage: STRING;
END_VAR
// Обработка ошибок
errorCode := MODBUS_CHECK_ERROR();
IF errorCode <> 0 THEN
   CASE errorCode OF
      1: errorMessage: = 'Illegal Function';
      2: errorMessage: = 'Illegal Data Address';
      3: errorMessage: = 'Illegal Data Value';
      4: errorMessage: = 'Slave Device Failure';
      ELSE
         errorMessage := 'Unknown Error';
   END_CASE
   // Логирование ошибки
   LOG_ERROR (errorMessage);
   // Оповещение оператора
   SEND_NOTIFICATION (errorMessage);
END_IF
```

Разбор возможных ошибок:

errorMessage := 'Illegal Function' – эта ошибка возникает, если мастер отправляет запрос с функцией, которая не поддерживается slave-устройством. В Modbus существует множество функций (например, чтение, запись регистров), и, если устройство не поддерживает запрошенную функцию, оно вернет эту ошибку.

errorMessage: = 'Illegal Data Address'; – ошибка указывает на то, что адрес данных, указанный в запросе, неверен или выходит за пределы доступного адресного пространства slave-устройства.

errorMessage: = 'Illegal Data Value'; – эта ошибка указывает на то, что значение данных в запросе неверно или недопустимо. Например, мастер пытается записать значение, которое выходит за допустимые пределы для данного регистра.

errorMessage: = 'Slave Device Failure'; – эта ошибка означает, что на slave-устройстве произошел сбой или оно не может обработать запрос. Это может быть вызвано аппаратными проблемами или внутренними сбоями устройства.

errorMessage: = 'Unknown Error'; – если возвращенный код ошибки не соответствует ни одному из перечисленных выше случаев, то программа присваивает сообщение «Unknown Error». Это помогает обработать непредвиденные или редкие ошибки, которые не были явно определены.

В случае необходимости, можно использовать программный код для работы Modbus на языке C++ с использованием libmodbus:

```
#include <iostream>
#include <modbus/modbus.h>
using namespace std;
int main() {
 // Подключение к серверу MODBUS
 modbus_t *ctx = modbus_new_tcp("192.168.1.100", 502);
 if (ctx == NULL) {
  cerr << "Ошибка подключения к серверу MODBUS" << endl;
  return 1;
 }
 // Чтение дискретного входа
 uint8_t value;
 modbus_read_input_bits(ctx, 1, 1, &value);
 cout << "Значение дискретного входа: " << (int)value << endl;
 // Чтение регистра хранения
 uint16_t registerValue;
 modbus_read_holding_registers(ctx, 1, 1, &registerValue);
 cout << "Значение регистра хранения: " << registerValue << endl;
 // Запись в регистр хранения
 registerValue = 100;
 modbus_write_holding_registers(ctx, 1, 1, &registerValue);
 cout << "В регистр хранения записано значение: " << registerValue << endl;
 // Закрытие соединения
 modbus_free(ctx);
 return 0;
}
```

Перед запуском кода необходимо изменить адрес сервера и номера регистров в соответствии с конфигурацией. Код демонстрирует базовые операции. Для более сложных задач, таких как чтение/запись массивов

данных, необходимо использовать другие функции библиотеки Modbus. Дополнительно можно использовать многопоточность для работы с несколькими серверами одновременно, добавить функции для чтения и записи других типов данных, например, плавающих чисел и строк.

Программный код диагностики ошибок в работе Modbus на языке C++:

```
#include <iostream>
#include <modbus/modbus.h>
using namespace std;
int main() {
 // Подключение к серверу MODBUS
 modbus_t *ctx = modbus_new_tcp("192.168.1.100", 502);
 if (ctx == NULL) {
  cerr << "Ошибка подключения к серверу MODBUS" << endl;
  return 1;
 }
 // Проверка ошибок при чтении дискретного входа
 uint8_t value;
 int status = modbus_read_input_bits(ctx, 1, 1, &value);
 if (status < 0) {
  switch (status) {
   case MODBUS_SOCKET_ERROR:
    cerr << "Ошибка сокета" << endl;
    break;
   case MODBUS_TIMEOUT:
    cerr << "Превышение времени ожидания" << endl;
    break;
   case MODBUS_SLAVE_REPLY_ERROR:
    cerr << "Ошибка ответа slave-устройства" << endl;
    break;
   default:
    cerr << "Неизвестная ошибка: " << status << endl;
  }
  modbus_free(ctx);
  return 1;
```

Таким образом происходит поиск самой распространённой ошибки – подключения к серверу Modbus. Дополнительно, в программу будет включена диагностика других возможных ошибок:

1. Ошибка сокета – указывает на проблему с сокетом, используемым для сетевого соединения. Возможные причины могут включать ошибки при

открытии сокета, потерю соединения, неправильные параметры сокета или проблемы с сетью.

2. Превышение времени ожидания – когда истекает время ожидания ответа от устройства. Это может случиться, если устройство не отвечает вовремя из-за перегрузки, отказа или проблем с сетью.

3. Ошибка ответа slave-устройства – указывает на проблему с ответом от slave-устройства. Это может быть связано с неверным или поврежденным ответом, или если устройство не может обработать запрос должным образом.

4. Неизвестная ошибка – обрабатывает все остальные неопознанные ошибки, которые не соответствуют ни одному из вышеперечисленных случаев. Код ошибки сохраняется в переменной «status» и выводится для дальнейшего анализа.

Исходя из совокупности всех факторов, можно заключить, что протокол Modbus продолжает оставаться ключевым стандартом в области промышленной автоматизации благодаря своей простоте, надежности и широкому распространению. Его универсальность позволяет интегрировать разнообразные устройства и системы, обеспечивая эффективное и стандартизированное взаимодействие. Несмотря на появление новых технологий, Modbus сохраняет свою актуальность, предлагая проверенные решения для управления и мониторинга промышленных процессов. Протокол играет важную роль в автоматизации, оставаясь ценным инструментом для обмена данными между устройствами, стимулирует развитие промышленных коммуникаций, способствуя созданию новых устройств и стандартов.

Литература

1. Фрасын П. Г., Никитин Н. В., Масанов Д.В., Рыжкова Е.А. Методологические основы работы с протоколом Modbus TCP с примером на

высокоуровневом языке программирования Python // Инженерный вестник Дона, 2023, № 11. URL: ivdon.ru/ru/magazine/archive/n11y2023/8785

2. Долидзе А. Н. Автоматическое согласование электроприводов без обратной связи с применением протокола Modbus // Инженерный вестник Дона, 2024, № 1. URL: ivdon.ru/ru/magazine/archive/n1y2024/8959

3. Гофман П. М., Кузнецов П. А. Инструменты программирования промышленных контроллеров. – Красноярск: СибГУ им. академика М. Ф. Решетнёва, 2019. — 94 с.

4. Petrulis B. Modbus RTU Tutorial: Tutorial On Modbus TCP/RTU Design: Modbus Tcp/Rtu (C#) Modbus Programming In C#. – Independently Published, 2021. – P. 40.

5. Matt Coutu. The Technicians Guide to Modbus TCP. – Amazon Digital Services LLC – KDP Print US, 2020. – P. 41.

6. Bryan E. A., Bryan L. A. Programmable Controllers Theory and Implementation. – Amer Technical Pub, 2003. – P. 1035.

7. Смит С. Цифровая обработка сигналов. Практическое руководство для инженеров и научных работников. – Москва: ДМК Пресс, 2011. – с. 720.

8. Bolton W. Programmable Logic Controllers. – Newnes, 2011. – P. 304.

9. Pratt G. The Book of CODESYS: The Ultimate Guide to PLC and Industrial Controls Programming with the CODESYS IDE and IEC 61131-3. – ControlSphere LLC, 2021. – P. 492.

10. Деменков Н. П. Языки программирования промышленных контроллеров. – Москва: МГТУ имени Н. Э. Баумана, 2004. – с. 172.

References

1. Frasyn P. G., Nikitin N. V., Masanov D.V., Ryzhkova E.A. Inzhenernyj vestnik Dona, 2023, №11. URL: ivdon.ru/ru/magazine/archive/n11y2023/8785

2. Dolidze A. N. Inzhenernyj vestnik Dona, 2024, №1. URL: ivdon.ru/ru/magazine/archive/ n1y2024/8959

3. Gofman P. M., Kuznecov P. A. Instrumenty programmirovanija promyshlennyh kontrollerov [Industrial controller programming tools]. Krasnojarsk: SibGU im. akademika M. F. Reshetnjova, 2019. 94 p.

4. Petrulis B. Modbus RTU Tutorial: Tutorial On Modbus TCP/RTU Design: Modbus Tcp/Rtu (C#) Modbus Programming In C#. Independently Published, 2021. P. 40.

5. Matt Coutu. The Technicians Guide to Modbus TCP. Amazon Digital Services LLC – KDP Print US, 2020. P. 41.

6. Bryan E. A., Bryan L. A. Programmable Controllers Theory and Implementation. Amer Technical Pub, 2003. P. 1035.

7. Smith S. Digital signal processing. A practical guide for engineers and scientists [Cifrovaja obrabotka signalov. Prakticheskoe rukovodstvo dlja inzhenerov i nauchnyh rabotnikov]. Moskva: DMK Press, 2011. 720 p.

8. Bolton W. Programmable Logic Controllers. Newnes, 2011. P. 304.

9. Pratt G. The Book of CODESYS: The Ultimate Guide to PLC and Industrial Controls Programming with the CODESYS IDE and IEC 61131-3. ControlSphere LLC, 2021. P. 492.

10. Demenkov N. P. Programming languages for industrial controllers [Jazyki programmirovanija promyshlennyh kontrollerov]. Moskva: MGTU imeni N. E`. Baumana, 2004. 172 p.

Дата поступления: 27.06.2024

Дата публикации: 29.08.2024