Анализ возможностей внедрения корректирующих линейных блочных кодов в протокол Modbus

С. А. Чаплиев [1], А. Б. Левина [1], А. И. Плотников[1]
[1]Санкт-Петербругский государственный электротехнический университет «ЛЭТИ»,
Россия, 197022, г. Санкт-Петербург

Аннотация. Корректирующие линейные блочные коды представляют собой перспективный подход к обеспечению целостности данных, сочетающий возможность не только обнаружения, но и исправления ошибок передачи. В статье анализируется целесообразность интеграции таких кодов в протокол Modbus, широко используемый в системах промышленной автоматизации, где в настоящее время применяются базовые механизмы обнаружения ошибок, такие как циклический избыточный контроль (CRC). Данное исследование демонстрирует, что внедрение корректирующих линейных блочных кодов способно значительно повысить надёжность передачи данных при сохранении приемлемого уровня избыточности.

Ключевые слова: корректирующие линейные блочные коды, код Боуза-Чоудхури-Хоквингема, БЧХ, коды с низкой плотностью проверок на чётность, LDPC, целостность информации, Modbus.

Abstract. Correcting linear block codes represent a promising approach to ensuring data integrity, combining not only error detection but also error correction capabilities. The article analyzes the feasibility of integrating such codes into the Modbus protocol, widely used in industrial automation systems where basic error detection mechanisms like Cyclic Redundancy Check (CRC) are currently employed. This study demonstrates that implementing correcting linear block codes can significantly enhance data transmission reliability while maintaining an acceptable level of redundancy.

Keywords: correcting linear block codes, Bose-Chaudhuri-Hocquenghem codes, BCH, Low-Density Parity-Check codes, LDPC, data integrity, Modbus.

Введение

В современных условиях информационные технологии играют решающую роль в обеспечении безопасности, надёжности и эффективности коммуникационных систем. Одним из ключевых требований к передаче данных является соблюдение таких принципов информационной безопасности как конфиденциальность, доступность и целостность. Последний из перечисленных принципов является наиболее критичным, так как именно целостность гарантирует, что передаваемая информация сохраняет своё исходное содержание и структуру на всех этапах обработки и передачи, оставаясь точной и неизменной, несмотря на возможные помехи и ошибки в каналах связи.

Для обеспечения целостности информации широко применяются корректирующие линейные блочные коды (далее КЛБК) – математические структуры, позволяющие обнаруживать и исправлять ошибки, возникающие при передаче или хранении данных. Наибольшее распространение получили коды Боуза-Чоудхури-Хоквингема (БЧХ) и коды с низкой плотностью проверок на чётность (LDPC), благодаря высокой надёжности и эффективности в системах, где требуется устойчивость к ошибкам и высокая точность передачи данных.

Полностью исключить искажения в каналах связи невозможно, поскольку передача данных подвержена влиянию случайных шумов и внешних помех. Особенно данная проблема актуальна для беспроводных сетей, где уровень ошибок может значительно варьироваться в зависимости от условий среды и качества сигнала. Согласно данным [1], в сетях стандарта IEEE 802.15.4 худшее значение коэффициента битовых ошибок (далее BER) может достигать 0,01, что означает один ошибочный бит на каждые 100 переданных. Даже при благоприятных условиях BER редко опускается ниже 0,00036, что эквивалентно одной ошибке на 2778 бит.

Использование корректирующих кодов позволяет существенно снизить вероятность искажений, обеспечивая целостность информации на выходе. В данной работе рассматривается возможность интеграции таких кодов в широко используемый протокол Modbus [2]. В статье представлены подходы к внедрению новых алгоритмов, выполнено их сравнительное тестирование, а также проанализировано влияние предложенных решений на надёжность и производительность системы передачи данных с точки зрения существующих исследований.

1. Рост рынка IoT и сопутствующие угрозы безопасности

Modbus [2] – это прикладной коммуникационный протокол модели OSI, построенный на архитектуре «ведущий-ведомый» (master-slave), позволяющий объединять до 247 устройств по одной шине. Протокол определяет структуру сообщений, правила обращения к данным и форматы ответа, при этом оставаясь независимым от физической среды – благодаря этому Modbus может быть реализован как на последовательных интерфейсах, так и в сетях TCP/IP.

Архитектура протокола предполагает, что ведущий инициирует запросы и управляет передачей данных, когда как ведомые устройства отвечают на полученные команды, не имея возможности инициировать коммуникацию. Каждое сообщение Modbus включает адрес устройства, код функции, поле данных и контрольную сумму, однако конкретная структура пакета зависит от используемой версии протокола.

Существует три основных реализации Modbus: RTU, ASCII и TCP. В Modbus RTU данные передаются в двоичном виде, а сообщения разделяются временными задержками. Данная версия отличается высокой компактностью, однако чувствительна к задержкам и требует точной синхронизации на физическом уровне. Для проверки целостности используется контрольная сумма CRC16. Структура пакета Modbus RTU в общем виде представлена в таблице 1.

Таблица 1 – Пакет Modbus RTU

Адрес ведомого устройства	Protocol Data Unit	Контрольная сумма CRC16
1 байт	≤ 253 байт	2 байта

Modbus ASCII использует шестнадцатеричное представление данных, где сообщения отделяются специальными символами: двоеточием в начале и символами возврата каретки и переноса строки в конце. Данный протокол более требователен к объёму передаваемых данных, однако менее чувствителен к временным задержкам в канале связи. Контроль осуществляется с помощью контрольной суммы LRC, а пакет включает символьные маркеры начала и конца, адрес, данные и итоговую сумму (таблица 2).

Таблица 2 – Пакет Modbus ASCII

“:”	Адрес ведомого устройства	Protocol Data Unit	LRC	“\r\n”
1 байт	2 байта	≤ 506 байт	2 байта	2 байта

Modbus TCP работает поверх стека TCP/IP и, по сути, представляет собой расширение Modbus RTU для сетевой среды. В данной версии отсутствует адрес ведомого устройства и контрольная сумма, так как функции маршрутизации и проверки целостности выполняются средствами транспортного протокола TCP. В общем виде структура пакета Modbus TCP представлена в таблице 3.

Таблица 3 – Пакет Modbus TCP

ID Транзакции	ID Протокола	Длина пакета	ID Блока	Protocol Data Unit
2 байта	2 байта	2 байта	1 байт	≤ 253 байт

Общим элементом всех рассмотренных версий является Protocol Data Unit (PDU), содержащий в себе два блока – код функции (принимает значение от 1 до 127, диапазон от 128 до 255 зарезервирован для кодов ошибок) и поле данных (до 252 байт).

2. Способы внедрения КЛБК в протокол Modbus

Существуют различные подходы к внедрению корректирующих линейных блочных кодов, каждый из которых имеет свои преимущества и ограничения, связанные с

вычислительной нагрузкой, совместимостью с существующими реализациями и архитектурными особенностями самих устройств.

Одним из возможных решений является полная замена последних бит контрольной суммы, таких как CRC или LRC, на контрольные биты, сформированные на основе корректирующего кода – такой подход позволяет не только обнаруживать, но и исправлять ошибки без необходимости повторной передачи пакета. Однако данный способ требует модификации всех компонентов системы и не обеспечивает обратной совместимости, а дополнительная нагрузка при обработке и увеличенная длина сообщений накладывают ограничения на его применение в ресурсоограниченных средах.

Другой вариант основан на надстройке КЛБК поверх существующей структуры пакета. В данном случае контрольные биты, сформированные с использованием КЛБК, добавляются в конец сообщения после контрольной суммы, рассчитанной по стандартной схеме – данное решение обеспечивает обратную совместимость: устройства, не поддерживающие КЛБК, могут просто игнорировать добавленную часть пакета, обрабатывая его по прежним правилам. Несмотря на возросшие накладные расходы на вычисления и передачу, данный подход позволяет постепенно внедрять новые методы коррекции ошибок без необходимости полной модернизации инфраструктуры.

Альтернативным решением является адаптивное кодирование, при котором выбор метода контроля целостности зависит от длины сообщения. Короткие пакеты передаются с использованием стандартных механизмов контроля, тогда как длинные сообщения дополнительно кодируются с помощью КЛБК. Рассмотренная стратегия позволяет сохранить эффективность и снизить накладные расходы в случае малых объёмов данных, одновременно обеспечивая надёжность передачи более критичных и объёмных сообщений. Несмотря на имеющиеся плюсы, реализация подобного подхода требует более сложной логики обработки пакетов, не обладает обратной совместимостью и увеличивает накладные расходы на вычисления и передачу длинных сообщений.

Важно отметить, что вопрос обратной совместимости во всех рассмотренных примерах во многом зависит от конкретной программной реализации Modbus. Поскольку протокол реализуется на прикладном уровне, его архитектура допускает гибкую настройку, включая возможность согласования методов контроля целостности на этапе инициализации соединения – это открывает путь к созданию гибких и адаптивных систем передачи данных, в которых выбор алгоритма может производиться в зависимости от возможностей участвующих устройств и условий внешней среды.

3. Сравнение методов кодирования

Для модификации существующих версий протокола Modbus с использованием КЛБК необходимо оценить эффективность уже существующих механизмов обеспечения целостности данных и сопоставить их с возможностями, которые предоставляют КЛБК. В частности, Modbus RTU использует контрольную сумму CRC16, представляющую собой циклический код, основанный на вычислении остатка от деления по заданному полиному. Согласно [3], программная реализация CRC16 требует порядка 64 тактов процессора на один байт данных, а поскольку проверка целостности включает повторное вычисление контрольной суммы для принятого пакета, то общее число тактов достигает 128 на байт. Аппаратная реализация значительно эффективнее: при использовании

специализированных модулей вычисление суммы осуществляется за 8 тактов на байт, то есть 16 тактов в расчёте на полный цикл кодирования и проверки.

Оценка производительности кодека БЧХ требует отдельного анализа процессов кодирования и декодирования. Аппаратная реализация кодирования, использующая линейные сдвиговые регистры с обратной связью, требует n тактов для кода (n, k). В частности, аппаратное декодирование БЧХ-кодов (15, 11, 1), (15, 7, 2) и (15, 5, 3) занимает 33 такта [4]. Таким образом, полный цикл работы кодека составляет 48 тактов на 15 бит, или приблизительно 26 тактов на байт. Согласно [5], в наихудшем случае аппаратный декодер БЧХ-кода с параметрами (n, k, t) требует порядка $2n + 2t$ тактов, что при указанных кодах даёт от 47 до 51 такта на блок, что эквивалентно около 25–27 тактов на байт. Программные реализации БЧХ-кодов в данном контексте не рассматриваются, так как их производительность значительно ниже. Так, кодек (255, 131, 18), реализованный программно и выполняемый на процессоре с тактовой частотой 3,1 ГГц, обрабатывает один кодовый блок за 73 микросекунды, что эквивалентно примерно 226 300 тактам или около 888 тактов на байт [6].

Для LDPC-кодов картина отличается. Аппаратное кодирование с использованием LDPC-кода (1536, 1024), по данным [7], требует 5632 такта, что соответствует приблизительно 44 тактам на байт, а декодирование 1536 бит также занимает 5632 такта, что эквивалентно около 30 тактам на байт, что даёт в сумме 74 такта на байт. В то же время программное декодирование LDPC-кода (16384, 4096), выполняемое на серверном процессоре Intel Xeon Gold 6148, занимает 34 микросекунды на блок, что соответствует примерно 57 тактам на байт [8]. Хотя в рассматриваемом случае кодирование не анализируется, предполагается, что его сложность не превышает сложности декодирования и является линейной, что позволяет оценить суммарные затраты на уровне не выше 114 тактов на байт [9].

Обобщённые характеристики приведённых алгоритмов с учётом как программной, так и аппаратной реализации представлены в таблице 4.

Таблица 4 – Характеристики алгоритмов

Алгоритм	Программная реализация, тактов на 1 байт	Аппаратная реализация, тактов на 1 байт
CRC16	128	16
БЧХ	888	26
LDPC	114	74

4. Целесообразность реализации КЛБК в Modbus

На основе анализа, представленного в предыдущем разделе, можно сделать выводы о целесообразности внедрения линейных блочных кодов в протокол Modbus.

Эффективность конкретного кодека зависит от множества факторов, включая аппаратную платформу, особенности программной реализации, используемые алгоритмы и

оптимизацию кода, поэтому однозначно определить универсальный лучший вариант невозможно. Тем не менее, сопоставление характеристик показывает, что по скорости аппаратной реализации БЧХ-коды уступают только CRC16, но при этом обладают существенно лучшими возможностями по исправлению ошибок. Если контрольная сумма CRC16 вычисляется программно, например, на микроконтроллере без встроенного аппаратного модуля, то её замена на программную реализацию LDPC может не только повысить производительность, но и значительно увеличить помехоустойчивость канала передачи данных. Аппаратные реализации кодеков обеспечивают ещё больший прирост: в случае БЧХ – почти в 5 раз, а для LDPC – примерно в 1,7 раза по сравнению с программным CRC.

При реализации LDPC-кодека в программной среде важно учитывать аппаратные возможности целевой платформы (например, поддержку процессором AVX инструкций), так как без их поддержки достигнуть заявленной эффективности и скорости работы не удастся, а упрощённые варианты алгоритма с ограниченным числом итераций могут привести к снижению качества процесса декодирования.

В случае, если CRC16 вычисляется аппаратными средствами, то любые изменения неизбежно приведут к снижению производительности системы. В данном случае наименее вычислительно затратным вариантом будет использование аппаратного кодека БЧХ, при котором увеличение задержки по сравнению с CRC16 составит 10 тактов на байт в случае полной замены и 26 тактов при добавлении к существующей структуре.

Внедрение LDPC в аппаратной форме приведёт к значительно большим издержкам – от 74 до 90 тактов на байт в зависимости от выбранного подхода, однако и этот вариант остаётся более эффективным и надёжным по сравнению с программной реализацией CRC.

Наряду с оценкой вычислительных затрат необходимо учитывать и экономическую составляющую. Переход к программным реализациям КЛБК может потребовать перехода на более современные или высокопроизводительные системы, увеличения объёма памяти и, в отдельных случаях, полной замены микроконтроллеров, что влечёт за собой как капитальные вложения, так и необходимость переработки программного обеспечения. Аппаратная реализация, напротив, предполагает проектирование специализированных модулей (кодеров и декодеров) на основе FPGA или ASIC, что существенно увеличивает производственные и временные издержки, особенно на этапах разработки, тестирования и интеграции. Таким образом, решение о внедрении корректирующих кодов должно приниматься с учётом баланса между необходимой степенью устойчивости к ошибкам, допустимыми затратами и возможностями аппаратной платформы.

Заключение

Данная статья представляет собой обзор перспектив модернизации протокола Modbus с целью повышения уровня обеспечения целостности данных при их передаче в промышленных и телекоммуникационных системах. В рамках исследования проведён сравнительный анализ алгоритмов, используемых для контроля целостности и применяемых в протоколе, с алгоритмами LDPC и БЧХ. Полученные результаты демонстрируют, что внедрение рассмотренных кодов возможно без существенного снижения общей производительности системы, предлагая дополнительное преимущество в виде возможности исправления ошибок, возникающих в процессе передачи. В дальнейшем планируется проведение прикладных исследований, направленных на экспериментальную

оценку эффективности предложенных подходов в реальных условиях эксплуатации, что позволит более точно определить их целесообразность и область применения.

Благодарности

Работа выполнена в рамках государственного задания Министерства науки и высшего образования Российской Федерации № 075-00003-24-02 (проект FSEE-2024-0003).

Список литературы

1. Bit error rate analysis of wireless sensor nodes with different packet size and distance / H. Arora, D. Sharma, H. P. Singh and J. P. Singh // 2016 International Conference on Advances in Computing, Communication, & Automation (ICACCA) (Spring), Dehradun, India, 2016;
2. Modbus Protocol Reference Guide / Minamitsumori, Nishinari-ku, MG CO., LTD // EM-5650 Rev.11;
3. Cyclic Redundancy Check (CRC) / Sudhir Bommena, standard ISO/TS 16949:2002 AN1148 // Microchip Technology Inc;
4. Implementation of BCH Code (n, k) Encoder and Decoder for Multiple Error Correction Control / Yathiraj H U, Mahasiddayya R Hiremath // International Journal of Computer Science and Mobile Applications, Vol.2 Issue. 5, May- 2014, pg. 45-54;
5. Improved High Code-Rate Soft BCH Decoder Architectures With One Extra Error Compensation / Y. M. Lin, H. C. Chang and C. Y. Lee // in IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 21, no. 11, pp. 2160-2164, Nov. 2013;
6. Performance investigation on BCH codec implementations / Dejan Azinović, Klaus Tittelbach-Helmrich, Zoran Stamenković // in IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 27 March 2017;
7. Research and Implementation of High-Efficiency and Low-Complexity LDPC Coding Algorithm / Liao, Xiong & Guo, Junxiong & Luo, Zhenghua & Xu, Yanghui & Chu, Yingjun // Electronics. 12. 3696. 10.3390/electronics12173696;
8. Fair comparison of hardware and software LDPC decoder implementations for SDR space links / V. Pignoly, B. Le Gal, C. Jego, B. Gadat and L. Barthe // 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS), Glasgow, UK, 2020;
9. Efficient encoding of low-density parity-check codes / T. J. Richardson and R. L. Urbanke // in IEEE Transactions on Information Theory, vol. 47, no. 2, pp. 638-656, Feb 2001.