Информационный подход к структурному синтезу контроллеров последовательных интерфейсов

Игорь Майкив[1], Анатолий Саченко[1], Владимир Кочан[1], Алексей Рощупкин[1],
Виктор Абабий[2], Виорика Судачевски[2]
[1]*Научно-исследовательский институт интеллектуальных компьютерных систем,*
Тернопольский национальный экономический университет, Минвуз Украины,
Институт кибернетики им. В.М.Глушкова, НАН Украины
imaykiv@yahoo.com, as@tneu.edu.ua, vk@tneu.edu.ua, o.roshchupkin@chnu.edu.ua
[2]*Технический Университет Молдовы*
ababii@mail.utm.md, svm700@mail.ru

***Abstract* —Информационный подход к проектирования контроллеров последовательных интерфейсов (КПИ) предполагает разделение информационной совместимости интерфейса на две составляющие: информационная совместимость на уровне целого сообщения и информационная совместимость на уровне бита. На основании такого подхода предложена 4-х уровневая модель, отображающая пространственно-временную организацию процессов, которые реализует КПИ при обработке сообщений. Это позволило упростить задачу проектирования КПИ и разработать новый программно-аппаратный метод их проектирования.**

***Index Terms* — контроллер последовательного интерфейса, информационная совместимость, цифровой автомат, операционное устройство.**

I. ВВЕДЕНИЕ

Компьютерные средства являются основой современных систем автоматизации и управления производственными процессами. Для объединения и взаимодействие отдельных компонентов указанных систем используются сетевые технологии на основе открытых промышленных шин. Для их эффективного функционирования достаточно обеспечить поддержку только трех уровней (прикладного, канального и физического) согласно семи уровней модели взаимодействия открытых систем ISO/OSI [1, 2]. При этом образуется единая программно-аппаратная среда для обмена информацией, а функции интерфейса (Рис.1) обеспечивают информационную (informational), электрическую (electrical) и конструктивную (design) совместимость (compatibility) между элементами системы или сети [3].

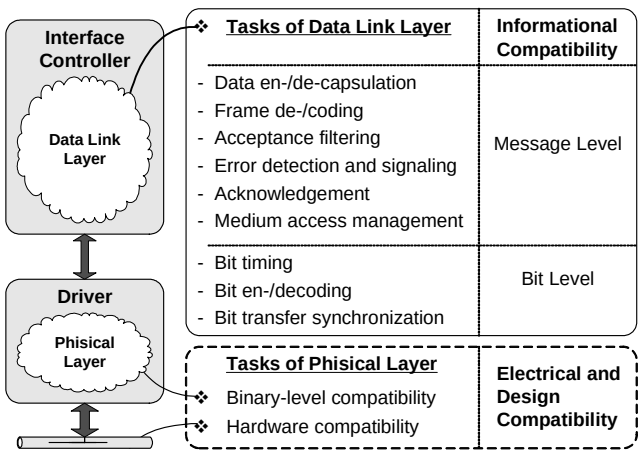

Рис.1 Функции интерфейса согласно модели ISO/OSI

Традиционно контроллер интерфейса реализуют аппаратно или же программно. При аппаратной реализации используется микроконтроллер с набором встроенных интерфейсов или же специализированная микросхема, которая обеспечивает поддержку одного из интерфейсов [4, 5]. При таком подходе можно достичь высокой скорости передачи данных и достоверности приема данных, а также низкого уровня загрузки микроконтроллера процессами, связанными с обработкой сообщений. Однако возрастает аппаратная сложность устройства, что приводит к уменьшению надежности, а также к увеличению энергопотребления, габаритов и стоимости оборудования.

Программный метод реализации контроллера интерфейса предполагает реализацию протокола обмена данными посредством последовательного выполнения набора подпрограмм, которые обеспечивают прием и передачу отдельных битов и сообщений в целом [6-9], а также способствуют возрастанию гибкости системы. Этот метод является наилучшим с позиции минимизации аппаратных затрат, однако он не может обеспечить высокую достоверность приема данных, а также эффективное использование вычислительной мощности микроконтроллера.

Поэтому авторами предложен новый подход к созданию контроллеров последовательных интерфейсов (КПИ), который позволит объединить преимущества аппаратного и программного методов реализации.

II. ОРГАНИЗАЦИЯ ПРОЦЕССОВ ОБРАБОТКИ ДАННЫХ В КПИ

Контроллер последовательных интерфейсов, обеспечивая информационную совместимость, реализует функции канального уровня (см. рис.1). В большинстве случаев информационная совместимость интерфейса рассматривается на уровне сообщения. Так как элементарной единицей информации является бит, представляется логичным разделение информационной совместимости интерфейса на две составляющие. Первая, это информационная совместимость на уровне целого сообщения. Вторая – информационная совместимость на уровне бита(см.рис.1). Такой подход позволяет рассматривать процесс передачи и приема сообщений как набор отдельных и независимых процессов. Первый из них реализует обработку сообщения в целом, а второй реализует обработку отдельного бита, как составляющей сообщения. В зависимости от протокола обмена, эти процессы выполняются одновременно или последовательно при приеме/передаче сообщений.

Последовательность процессов, реализуемых КПИ при приеме сообщений, представлена на рис. 2.

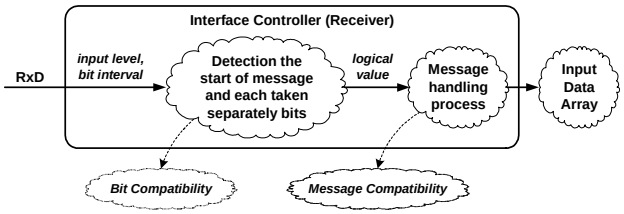

Рис.2 Множество процессов при приеме сообщений

При реализации процесса обработки бита приемник обеспечивает: (i) обнаружение признака начала сообщения; (ii) тактирование временного интервала бита; (iii) синхронизацию к входному потоку данных; (iv) фиксацию логического уровня. Данные на входе приемника характеризуются входным уровнем и временем присутствия. Результатом выполнения процесса обработки бита является набор логических значений зафиксированных на протяжении интервала времени приема одного бита. Одновременно реализуется процесс обработки сообщения, обеспечивая: (i) идентификацию логического значения принятого бита; (ii) принятие решения о начале сообщения; (iii) проверку адреса; (iv) обнаружение ошибок в сообщении; (v) выделение данных из принятого сообщения и др. Результатом выполнения этого процесса является набор данных поступающих далее на обработку соответствующим программным обеспечением (прикладной уровень).

Последовательность процессов, реализуемых КПИ при передачи сообщений, представлена на рис. 3.

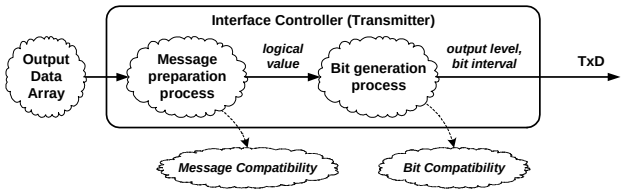

Рис.3 Множество процессов при передаче сообщений

В процессе формирования сообщения передатчик принимает набор данных от прикладного программного обеспечения; формирует сообщение и реализует побитовую передачу. Результатом выполнения процесса формирования сообщения является двоичная последовательность. Одновременно передатчик реализует процесс формирования и передачи бита. В результате выполнения этого процесса каждый отдельный бит на выходе передатчика характеризуется логическим значением и временем передачи.

Из проведенного анализа следует, что при обмене данными выполняется четыре процесса. При этом два процесса реализуют прием и передачу отдельного бита, а два других - передачу сообщений в целом. Кроме того, эти две группы процессов принципиально различаются между собой. Процессы обработки и формирования бита связаны с обработкой информации во времени. В тоже время процессы обработки и формирования сообщений реализуют набор логических функций, необходимых для формирования дополнительных данных (полей).

Каждый из процессов может быть реализован как программно, так и аппаратно, что позволяет иметь четыре комбинации, как для приемника, так и передатчика. В этом случае формулируется задача выбора оптимального способа реализации каждого из процессов.

Для отображения взаимодействия между выше обозначенными процессами предложена 4-х уровневая модель (Рис. 4).

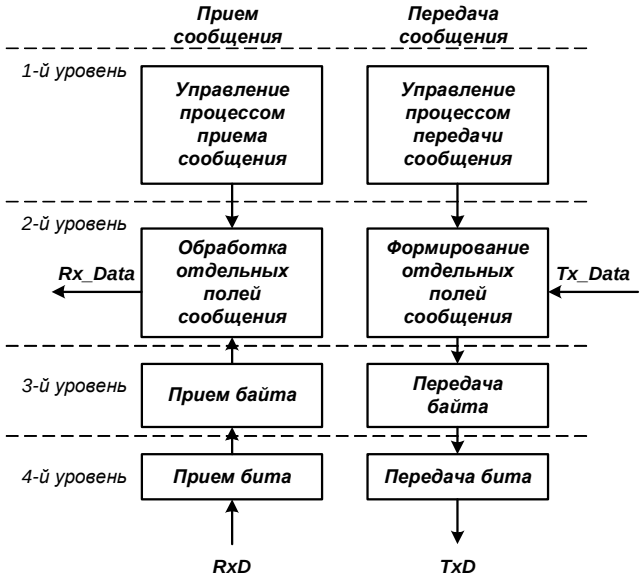

Рис.4 Модель взаимодействия процессов при приеме/передачи сообщений

На первом уровне расположены процесс непосредственно управляющие приемом и передачей сообщений. На втором уровне находятся множество процессов выполняющих обработку и формирование отдельных полей сообщения.

На третьем уровне расположены процессы обеспечивающие прием и передачу байта (основной информационной единицы для большинства полей

сообщения). На четвертом уровне расположены процессы, обеспечивающие прием и передачу отдельных битов. Взаимодействие между процессами осуществляется с помощью переменных, которые разделены на две группы: информационные и управляющие.

С представленной на рис. 4 модели следует, что процессы первого уровня управляют процессами приема/передачи сообщений, в то время как процессы 2-го и 3-го уровней обеспечивают обработку отдельных полей сообщения. Аналогичный подход применен к процессам 4-го уровня, которые в свою очередь разделены на два отдельных подпроцессы - управляющий и операционный.

III. СТРУКТУРНАЯ ОРГАНИЗАЦИЯ КОНТРОЛЛЕРОВ ПОСЛЕДОВАТЕЛЬНЫХ ИНТЕРФЕЙСОВ

Подход, предусматривающий разделение функционального узла на управляющую и операционную части, применяется в прикладной теории цифровых автоматов (ПТЦА) [10, 11] при проектировании микропрограммных автоматов (МПА), которые, включают два независимых узла – управляющий цифровой автомат (ЦА) и операционное устройство (ОУ).

Это позволяет рассматривать КПИ как сеть синхронно взаимодействующих МПА(Рис. 5).

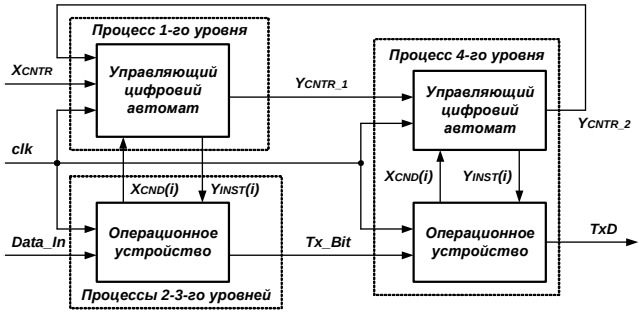

Рис.5 Структурная схема блока передатчика как сети МПА

Процессы 1-3 уровней формируют независимый МПА, обеспечивающий формирование сообщения в целом. Процесс 4-го уровня также реализован как независимый МПА. Аналогично можно представить блок приемника КПИ, но в этом случае первым будет стоять МПА, обеспечивающий прием бита (процесс 4-го уровня), а вторым – МПА, выполняющий обработку принятого сообщения (процессы 1-3 уровней).

На практике управляющий ЦА реализуют как автомат Мура/Мили или совмещенный С-автомат. Математическая модель последнего задается шести компонентным вектором $W = \langle X, Y, S, \delta, \lambda, s_1 \rangle$, где $X = X_{CNTR} \cup X_{CND}$ - множество входных сигналов, объединяющее входные управляющие сигналы (X_{CNTR}) и микроусловия (X_{CND}); $Y = Y_{CNTR} \cup Y_{INST}$ - множество выходных сигналов, объединяющее выходные управляющие сигналы (Y_{CNTR}) и микрокоманды (Y_{INST}); S - множество внутренних состояний ЦА; δ - функция переходов ЦА; λ - функция выходов ЦА; $s_1 \in S$ - начальное состояние ЦА.

Предложенный подход позволяет разделить задачу синтеза КПИ на две отдельные задачи: 1) синтез управляющего ЦА, что решается традиционными методами с использованием ПТЦА [5, 6]; 2) синтез ОУ путем аппаратного отображения потокового графа алгоритма [7].

Описанные выше процессы можно реализовать как программным так и аппаратным способом. В случае программной реализации КПИ - функции канального уровня, реализуемые процессами 1-3 уровней, достаточно легко выполняются процессором. В то же время, реализация набора временных интервалов, формирующих t_{bit} как набор «пустых циклов», ведет к неэффективному использованию вычислительной мощности процессора.

В случае аппаратной реализации КПИ – процессы 4-го уровня сравнительно легко реализуются с помощью счетчиков-делителей и несложного ЦА. В тоже время реализация процессов 1-3 уровней требует значительных аппаратных затрат.

Выше обозначенные противоречия являются принципиальными, и не могут бить устранены в случае применения традиционных методов реализации КПИ. Поэтому был предложен новый программно-аппаратный метод создания КПИ [12, 13], в котором аппаратная реализация процессов 4-го уровня, и программная реализация процессов 1-3 уровня выполнены с помощью процессорного ядра в составе программируемой логической интегральной схемы (ПЛИС)(Рис. 6).

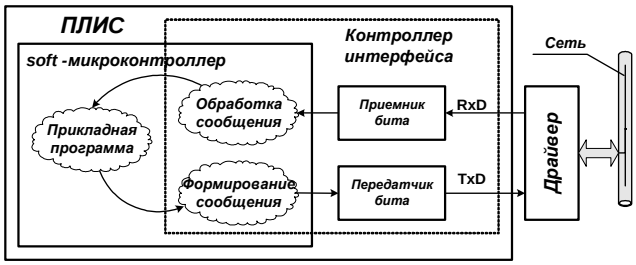

Рис.6 Структурная схем системы на база ПЛИС с программно-аппаратным КПИ

В состав ПЛИС входит ядро микроконтроллера, который программным путем выполняет обработку и формирование сообщения, а также два независимых аппаратных узла для приема и передачи бита. Преимуществом предложенной структуры, кроме уменьшения аппаратных затрат на реализацию КПИ, является возможность независимого перепрограммирования как МК так и ПЛИС, что позволяет повысить эффективность системы в целом.

Экспериментальная оценка эффективности предложенного метода выполнена на примере интерфейса 1-Wire с использованием базы «Spartan-3E Development Kit» фирмы XILINX и ядра 8-разрядного

микроконтроллера PicoBlaze. В результате подтверждено, что объем необходимых аппаратных ресурсов ПЛИС для реализации системы с программно-аппаратным КПИ 1-Wire уменьшился на 35% по сравнению с аппаратно реализованным КПИ. Кроме того, требования к объему памяти программ МК сократились на 53% по сравнению с программно реализованным КПИ. В целом можно считать, что КПИ на основе предложенного программно-аппаратного метода, по указанным выше характеристикам, превосходят известные технические решения, представленные на рынке [14].

IV. ВЫВОДЫ

На основе информационного подхода к процессу проектирования контроллеров последовательных интерфейсов предложена 4-уровневая модель, которая позволяет наглядно отображать информационные связи между отдельными процессами при приеме и передаче сообщений и служит эффективным инструментом при реализации контроллеров последовательных интерфейсов этапах их функционального анализа и структурного синтеза. Кроме того, данная модель демонстрирует в наглядной форме сущность противоречий между аппаратным и программным методами реализации контроллеров последовательных интерфейсов.

На базе предложенной модели разработан новый программно-аппаратный метод создания контроллеров последовательных интерфейсов, экспериментальная реализация которых на интерфейсе 1-Wire подтвердила их преимущество по сравнению с известными техническими решениями.

БЛАГОДАРНОСТИ

Авторы хотели бы поблагодарить Министерство Образования и Науки Украины за поддержку данной работы, которая выполнена в рамках двухстороннего международного проекта «Дистрибутивные сенсорные сети с реконфигурацией вычислительных узлов» №3367-А от 04.06.2013.

ЛИТЕРАТУРА

[1] Warren Miller Select Serial Interfaces Wisely for Optimal Connectivity. Available: http://www.digikey.com/en/articles/techzone/2014/sep/select-serial-interfaces-wisely-for-optimal-connectivity

[2] M. Schleicher, F. Blasinger Digital Interfaces and Bus Systems for Communication. Practical fundamentals. M.K. Juchheim GmbH, 2005. Available: https://www.scribd.com/doc/17366196/Digital-Interfaces-and-Bus-Systems-for-Communication-Practical-Fundamentals-FAS603gb

[3] Мячев А.А Интерфейсы систем обработки данных / А.А. Мячев, В.Н. Степанов, В.К. Щербо; под ред. А.А. Мячева. – М.: Радио и связь. - 1989. – 416 с.

[4] D. Wobshall, “An Implementation of IEEE 1451 NCAP for Internet Access of Serial Port-Based Sensors,” Proceedings of second Sensor for Industry Conference SIcon/02, 19-21 November 2002, Houston, Texas, ISBN 1-55617-834-4, pp.157-160.

[5] D. Wobschall, A. Stepanenko, I. Maykiv, R. Kochan, A. Sachenko, V. Kochan A Multi-port Serial NCAP using IEEE1451 Smart Transducer Standard. IEEE Sensor Application Symposium (SAS-2009): IEEE international conference, 17-19 February 2009: proceedings. - New Orleans, USA, 2009. - P.293-297.

[6] Software SPI examples for the C8051F30x family. Application note AN128. Silicon Laboratories. Available: http://www.silabs.com/Support%20Documents/TechnicalDocs/an128.pdf

[7] Implementing a LIN Slave Node on a PIC18F1320. Application note AN864. Microchip. Available: http://ww1.microchip.com/downloads/en/AppNotes/00864a.pdf

[8] LIN v1.3 Protocol Implementation on Atmel AVR Microcontrollers. Application note AVR 322. Atmel. Available: http://www.atmel.com/images/doc7548.pdf

[9] Wireless M-bus software implementation. Application note AN451. Silicon Laboratories. Available: http://www.silabs.com/Support%20Documents/TechnicalDocs/AN451.pdf

[10] Баранов С.И. Синтез микропрограммных автоматов. – Л.: Машиностроение, 1978. – 234 с.

[11] Прикладная теория цифровых автоматов / К.Г. Самофалов, А.М. Романкевич и др. – К.: Вища шк. Головное изд-во, 1987. – 357 с.

[12] I. Maykiv, A. Stepanenko, D. Wobschall, R. Kochan, V. Kochan, A. Sachenko. Software-Hardware Method of Serial Interface Controller Implementation // Computer Standards & Interfaces. Volume 34, Issue 6, pp.509-516, November 2012. Available: http://www.sciencedirect.com/science/article/pii/S0920548911001103.

[13] Майкив И.М.. Кочан Р.В., Кочан В.В. Программно-аппаратный контролер интерфейса. Патент на изобретение № 90766 от 25.05.2010. (на украинском языке).

[14] Understanding the DS1WM Synthesizable 1-Wire Bus Master. Application note 5507. Maxim Integrated Products, Inc. Available: http://www.maximintegrated.com/en/app-notes/index.mvp/id/5507.