Проблемы достижения высокой производительности TCP/IP-коммуникаций во встраиваемых системах

Олег ИВАНОВ
oyi@efo.ru

В статье рассмотрены различные методы реализации сетевых коммуникаций для встраиваемых систем и даны рекомендации для получения высокой производительности канала передачи данных при использовании протоколов TCP/IP.

Введение

Несмотря на то, что в основу Ethernet-коммуникаций положена простая концепция стека протоколов, позволяющая разбить задачу на уровни, разработчику встраиваемых систем необходимо понимать работу всего стека целиком, и реализация всех уровней в микроконтроллере с ограниченными ресурсами может оказаться сложной задачей. Вопрос достижения высокой скорости передачи через Ethernet-соединение в последнее время становится все более актуальным именно в отношении встраиваемых устройств. Если раньше от этих устройств требовалась лишь эпизодическая передача небольших порций данных, например информации о состоянии устройства или нескольких отсчетов сенсора, то сейчас даже для конфигурирования устройства уже применяют WEB-сервер с полноценным графическим интерфейсом. Это прежде всего удобно и позволяет организовать дружеский для пользователя интерфейс взаимодействия с устройством (так называемый человеко-машинный интерфейс). В персональных компьютерах скорости подходят уже к гигабитным, а вот в сегменте встраиваемых устройств получение высокой производительности остается по-прежнему непростым вопросом.

В тех случаях, когда перед разработчиком встает задача обеспечить передачу данных через сетевой интерфейс, он сразу начинает рассматривать стек TCP/IP, портированный для применения в микроконтроллерах. Библиотеки, реализующие стек, доступны как в бесплатных вариантах (они распространяются под лицензией open source), так и предлагаются многими фирмами как коммерческий продукт.

Часто разработчики надеются, что, применяя микроконтроллер со встроенной поддержкой MAC-уровня и всего лишь добавив микросхему, реализующую физический уровень Ethernet, и используя одну из этих библиотек, они смогут добиться полной функциональности и высокой производительности Internet/Ethernet-соединения. К сожалению, это далеко от истины. Полнофункциональный стек TCP/IP требует организации сокетов и буферов, которые интенсивно используют оперативную память (ОЗУ). Как правило, размер этой памяти в микроконтроллере ограничен. А малые размеры выделенных буферов и обслуживание ресурсоемкого приложения одновременно с обслуживанием сетевого соединения могут привести к тому, что производительность реализованного TCP/IP-стека будет не лучше RS232-соединения. Это подтверждается многочисленными вопросами в форумах: «Почему я получаю такую низкую пропускную способность сетевого интерфейса, хотя использую 100-Мбит сеть?»

Это не является проблемой в том случае, когда к сетевому подключению предъявляются только требования передачи небольших порций данных и не ставится задача передачи больших потоков информации. То есть для случаев, когда высокая пропускная способность не нужна. Но, тем не менее, если устройство оснащено портом Ethernet, от него ожидают производительности, которая составляет хотя бы несколько мегабит в секунду. Это достижимо, но лишь при соблюдении правил, о которых мы расскажем далее.

Для примера рассмотрим особенности использования транспортного протокола (Transmission Control Protocol, TCP). Он аналогичен и для IP, и для UDP.

Рис. 1. Схематическое изображение размещения кадра Ethernet в буфере

Сетевые буферы

TCP/IP-стек размещает принимаемые и передаваемые через Ethernet пакеты в сетевые буферы, которые далее передаются следующим уровням стека. Сетевые буферы представляют собой выделенные области памяти в ОЗУ.

В буферах размещаются данные совместно со служебной информацией, обеспечивающей работу стека. На рис. 1 показано, как передаваемые данные инкапсулируются в Ethernet-кадр, который передается на втором уровне стека, то есть при помощи транспортного протокола.

Буфер включает данные одновременно со служебной информацией, которая используется стеком протоколов. Она информирует о данных, содержащихся в буфере, которые либо приняты сетевым интерфейсом (Network Interface Card, NIC) и будут переданы на обработку в стек, либо помещены в буфер из стека с целью дальнейшей передачи через NIC. Взаимодействие данных между уровнями стека TCP/IP показано на рис. 2.

Максимальный размер сетевого буфера зависит от того, какая сетевая технология будет использована. В наши дни для создания локальных сетей (Local Area Network, LAN) повсеместно используется Ethernet, поэтому все особенности программной реализации мы будем рассматривать применительно к Ethernet.

Первоначально в стандарте Ethernet был определен максимальный размер кадра — 1518 байт. Вычитая из размера кадра информацию для TCP- и IP-протоколов, мы полу-

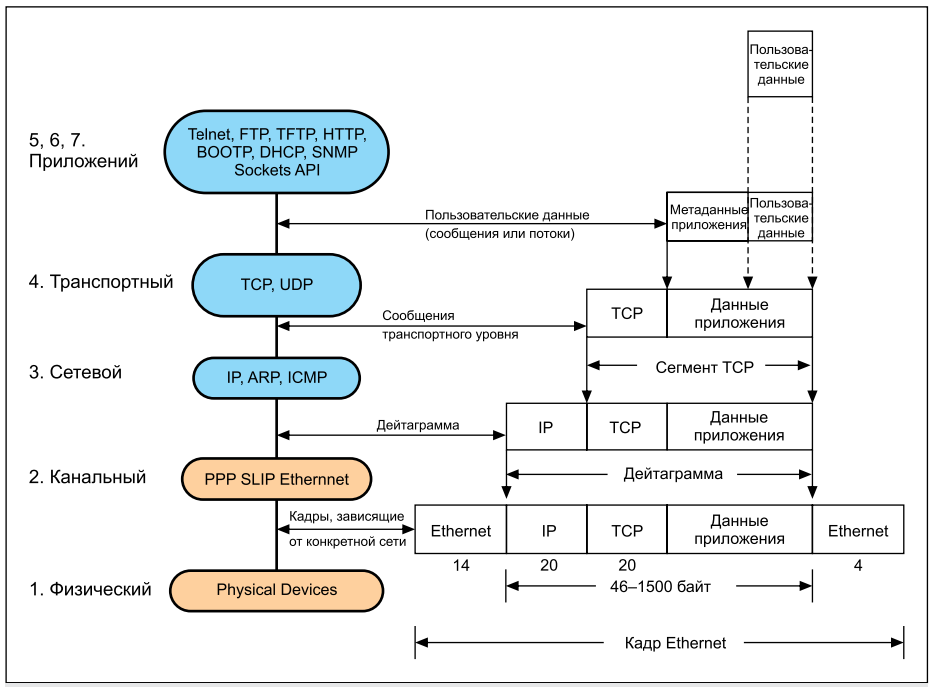

Рис. 2. Взаимодействие уровней протокола TCP/IP

чим 1460 байт для данных в TCP-сегменте. То есть для передачи одного кадра через стек TCP/IP требуется сетевой буфер в пределах 1600 байт, включая метаданные, необходимые для организации сетевого буфера.

Таким образом, для организации передачи данных, которые не требуют высокой пропускной способности, как, например, сенсор, периодически посылающий небольшие порции данных, достаточно выделить сетевой буфер, по размеру немного превышающий 1600 байт или даже меньше. Но необходимо учитывать, что количество служебной информации остается неизменным, и эффективность при выборе буфера меньшего размера снизится.

Устанавливая соединение на уровне TCP-протокола, сетевые устройства согласовывают между собой размер передаваемого кадра, известный под названием максимальный размер сегмента (Maximum Segment Size, MSS). Встраиваемые системы зачастую ограничены в оперативной памяти, которая используется не только под сетевые буферы, но и в других целях (ОЗУ, например, активно используется стеком и т. п.).

Сетевые операции

Множество одновременно выполняющихся сетевых операций также влияет на общую производительность устройства, например, когда сетевые буферы не успевают быстро освободиться после завершения задачи.

В случае если TCP-сегмент не освобождает буферы до момента подтверждения успешности приема или если подтверждение не получено своевременно, передача пакета повторяется.

Если размеры сетевых буферов и их количество ограничены, происходит приостановка передачи следующих данных (пакеты теряются), и это влияет на общую производительность системы. В случае, когда все сетевые буферы связаны с пакетами (которые посылаются однократно, повторно или принимается подтверждение об их приеме), TCP/IP-стек будет замедляться до тех пор, пока не произойдет освобождение буферов.

Преимуществом выделения нескольких небольших сетевых буферов, работающих одновременно, является возможность обслуживания большего количества различных протоколов использующих транспорт TCP.

Это решение хорошо подходит для случая, когда информационный обмен между устройствами происходит небольшими пакетами, например если удаленный сенсор периодически посылает информацию о своем состоянии.

Недостатком в этом случае является то обстоятельство, что каждый пакет переносит меньше данных. Соответственно, протоколы HTTP, FTP и другие, требующие высокой производительности, не очень подходят для этой модели выделения буферов.

В конечном счете, если размер ОЗУ недостаточен для выделения нескольких сетевых буферов, стек TCP/IP будет просто «ползать».

Управление потоком через «оконный доступ»

TCP осуществляет управление потоком пересылаемых данных с помощью механизма, называемого «скользящее окно», который используется как при передаче, так и при приеме пакетов. Поле в TCP-заголовке используется для работы «оконного механизма» следующим образом:

- Ширина окна, перемещаемого по сетевому буферу, соответствует количеству информации (в байтах), которое способна принять принимающая сторона. Это позволяет TCP-протоколу контролировать поток посылаемых данных.
- Количество принимаемых данных зависит от размера и количества приемных сетевых буферов.
- Максимальный размер окна составляет 65 535 байт (так как поле имеет 16 бит).
- Опустошение буфера приводит к остановке передачи. Передающая сторона пересылает байты последовательно.

На рис. 3 показан принцип работы «оконного доступа»:

- Окно имеет размер 2048 байт.
- Байты с первого по 512-й уже посланы, и установлен флаг TCP PSH, так как уже получено подтверждение от принимающей стороны (PSH и ACK).
- Байты с 513-го по 1536-й уже посланы, но еще нет подтверждения успешности приема.
- Байты с 1537-го до 2560-го могут быть посланы немедленно.

Сразу же как будет принято подтверждение о приеме байтов с 513-го до 1536-го, окно

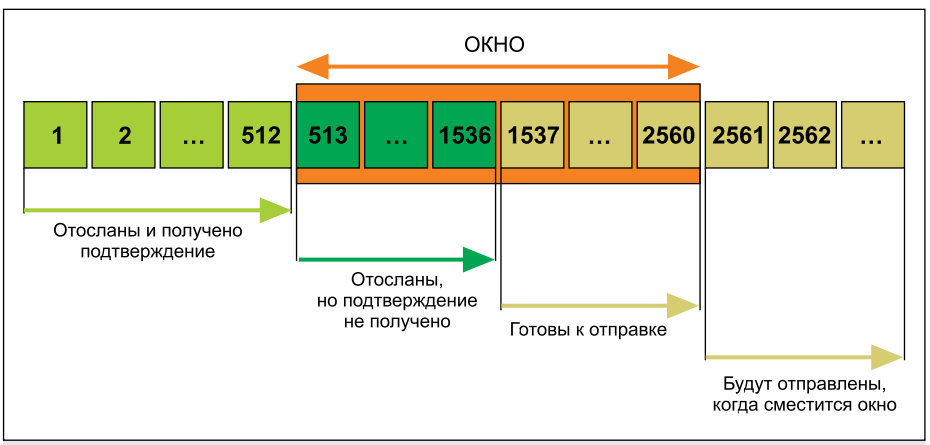

Рис. 3. Метод «оконного доступа»

будет перемещено на 1024 байта вправо, и байты с 2561-го до 3584-го могут быть посланы.

Таким образом, в разрабатываемом устройстве размер окна следует конфигурировать в соответствии с величиной располагаемого сетевого буфера.

Для примера возьмем устройство, которое будет иметь восемь буферов с размером MSS 1460 байт, это позволит их распределить как четыре сетевых буфера на прием и четыре буфера на передачу. В соответствии с этим окна для приемного и передающего буферов будут составлять 4 раза по 1460 байт (в сумме 5840 байт). Каждый принимаемый пакет будет уменьшать размер окна на 1460 байт и сообщать новый размер окна передающему устройству. После того как принятый пакет будет забран из буфера, размер окна опять восстановит свое значение и сообщит принимающей стороне, когда следующий пакет будет послан.

Обычно сеть может пересылать пакеты значительно быстрее, чем встраиваемый микроконтроллер способен их обрабатывать. В случае если приемный буфер уже содержит четыре пакета, а микроконтроллер еще не обработал их, окно будет уменьшено до нулевой величины, и передающее устройство, получив нулевой размер окна, приостановит передачу до тех пор, пока принимающее устройство не сообщит о способности снова принимать пакеты.

С передающей стороны стек, выполняющий заполнение буфера передачи, будет остановлен и в зависимости от реализации (конфигурации) будет пробовать снова передать пакеты через некоторое время или осуществит выход (Blocking/Non-blocking sockets).

UDP-протокол не имеет такого механизма. И при недостаточном размере входного буфера на принимающей стороне пакеты будут потеряны. Контроль ситуации в этом случае возлагается на приложение, для которого адресованы пакеты, и это потребует усложнения кода, так как необходимо будет предусмотреть, что делать с потерянными пакетами: перезапрашивать их снова или не обращать внимания на потери.

Расчет производительности TCP

Количество пакетов, отосланных и принятых устройством, в наилучшем случае можно примерно определить по формуле:

$$\text{Суммарное количество TCP-пакетов} \approx$$
$$\approx \text{суммарное количество соединений} \times \text{размер окна}.$$

Это будет соответствовать максимальной пропускной способности TCP-соединения. Общая пропускная способность канала сильно зависит от количества буферов, выделенных на канал.

Так как большинство микроконтроллеров для встраиваемых приложений имеет ограниченную производительность ядра, ресурсы которого используются и для выполнения основных задач (а сетевой обмен, как правило, это вспомогательная задача, если приложение принимает пакеты от таких устройств, как маршрутизатор или компьютер), обработка сетевых буферов будет недостаточно быстрой. Это приведет к потере пакетов и будет вызывать их повторную передачу, что еще более снизит общую производительность.

Работа окна с множеством сокетов

Предыдущие положения касались случая, когда встраиваемое устройство использовало только один сокет (то есть одно логическое соединение). Рассмотрим, что происходит, когда необходимо обслуживать несколько логических соединений одновременно. Разумеется, суммарная пропускная способность будет поделена на все логические соединения. А значит, размер окна следует выбирать с учетом максимального количества таких соединений (или максимального количества сокетов, обслуживаемых одновременно).

Используем тот же пример с пятью сокетами, где принимающее окно размером 5840 байт приходится на каждый сокет. Таким образом, должно быть сконфигурировано 20 сетевых буферов (четыре буфера на окно умножаем на пять сокетов). Полагая, что наибольший сетевой буфер имеет размер 1600 байт, нам необходимо выделить 32 кбайт памяти (20×1600 байт). В противном случае производительность системы будет значительно снижена за счет чрезмерного увеличения повторных посылок пакетов, которые устройство не успевает принять и разместить в свободных буферах.

Как же определить оптимальные размеры окон для приема и передачи? Это можно сделать методом обратных вычислений.

Когда для приема зарезервировано 20 сетевых буферов, система может обслужить максимум пять логических соединений одновременно. (И, соответственно, размер окна для приема будет равен произведению количества буферов на MSS, поделенному на количество одновременно открытых сокетов.)

Из этого следует, что чем меньше размер MSS, тем больше оперативной памяти потребуется для организации дополнительных буферов.

Задержка в подтверждении

Другим важным фактором, влияющим на производительность TCP, является ситуация, когда микроконтроллер не успевает своевременно передать пакет с подтверждением того, что прием пакета был успешным. Это вызывает необходимость для передающей стороны повторить передачу пакета. В этом случае необходимо организовать так называемый режим с отложенным подтверждением. Отложенное подтверждение способно повысить производительность до 30%.

Для того чтобы определить размер окна для передающих буферов в зависимости от максимального количества одновременно открытых сокетов, мы должны использовать следующие формулы:

- Для случая, если мы не применяем отложенное подтверждение:

$$\text{Размер окна для передачи} =$$
$$= (\text{количество буферов} \times \text{MSS})/\text{удвоенное количество сокетов}.$$

- Для случая с применением отложенного подтверждения:

$$\text{Размер окна для передачи} =$$
$$= (\text{количество буферов} \times \text{MSS})/\text{количество сокетов}.$$

Похожие соотношения должны быть использованы и для UDP-пакетов, просто в этом случае контроль потока передачи выполняется на уровне приложений. Поэтому разработчик должен для себя решить, какой протокол — IP или UDP — он будет использовать, и при использовании только UDP часть программного кода для организации IP может быть удалена или не реализовываться. Но тогда контроль принимаемых данных он должен организовать сам.

Из этого следует, что для повышения производительности сетевого соединения необходимо как можно быстрее реагировать на прием и передачу пакета, и для этого требуется режим DMA (прямой доступ к памяти) и повышение частоты тактирования ядра микропроцессора.

Для получения приемлемой производительности нам необходимо иметь достаточный размер ОЗУ в микроконтроллере. Разумеется, старшие представители микроконтроллеров у разных производителей уже имеют достаточное количество ОЗУ для выделения сетевых буферов или интерфейс, позволяющий расширить ОЗУ за счет применения дополнительной памяти. Но такое решение приемлемо только в том случае, когда на микроконтроллер возложена серьезная задача, эффективно использующая его ресурсы, и к тому же возможности микроконтроллера позволяют одновременно осуществить эффективную передачу данных. Если задача не ресурсоемкая, то выбор этого микроконтроллера будет избыточен с точки зрения располагаемых ресурсов, а это в свою очередь приведет к нерациональному повышению стоимости конечного изделия.

При выборе решения нужно хорошо взвешивать все «за» и «против»: реализовывать стек TCP/IP программно или использовать спе-

циализированный чип, имеющий необходимые буферы, а зачастую уже реализующий стек аппаратно. В первом случае в дополнение к микроконтроллеру необходимо будет использовать микросхему, реализующую физический уровень Ethernet. (Ассортимент микроконтроллеров с уже реализованным физическим уровнем постоянно уменьшается, что зачастую не позволяет выбрать оптимальное аппаратное решение.)

Рассмотрим рекомендации, которых необходимо придерживаться для получения высокой пропускной способности Ethernet-соединения применительно к встраиваемым устройствам.

Начнем с рассмотрения случая, когда разработчик решил использовать программный стек протоколов, например lwIP:

- Использовать микроконтроллер с big-endian организацией памяти, если это возможно, так как это избавит программу от дополнительных преобразований (в сетевых технологиях, как правило, используется big-endian).
- Использовать прерывания и прямой доступ к памяти (DMA), когда это возможно.
- Учитывая, что стандартные драйверы могут быть написаны для условия преобладания трафика в одном направлении, необходимо проверять, что данное направление наиболее подходит для вашей задачи.
- Если аппаратные средства микроконтроллера это позволяют, использовать драйверы с поддержкой scatter-gather через DMA. (Иначе ядро микроконтроллера основную часть времени будет заниматься пересылкой данных в сетевые буферы, и на другие задачи не останется свободных ресурсов.)

Другим важным фактором, влияющим на производительность, является вычисление контрольных сумм отсылаемых пакетов и проверка корректности для принятых.

Если микроконтроллер имеет в своей периферии аппаратную реализацию вычисления и проверки контрольных сумм, ее нужно обязательно использовать. Если же аппаратная поддержка не предусмотрена, необходимо постараться максимально оптимизировать эти процедуры при компиляции и, возможно, даже реализовать эти процедуры на ассемблере.

Далее следует выбрать, какой протокол будет использован: UDP или IP.

На первый взгляд, наиболее удачное решение — это выбрать передачу информации посредством дейтаграмм (UDP). Этот протокол требует передачи меньшего количества служебной информации по сравнению с IP, но функции контроля переданной информации, то есть целостность и порядок принимаемых и посылаемых данных, необходимо будет реализовать в приложении. Кроме того, когда тестируется приложение в локальной сети с малыми задержками, кажется, что все хорошо, но когда требуется пересылка за пределы локальной сети, все становится не так очевидно.

Важно также выбирать размер формируемого пакета таким образом, чтобы он не превысил максимальный размер для вашей сети (как правило, 1472 байт для стандартного Ethernet). Поэтому наиболее надежным способом является применение IP-протокола, несмотря на повышение суммарного трафика и других особенностей, таких как невозможность выбрать удобный для себя размер пакета.

Второе и, может быть, более рациональное решение — применить специализированный микроконтроллер, обеспечивающий сетевые соединения и зачастую реализующий полный стек протокола аппаратно, а также снабженный ОЗУ для сетевых буферов и позволяющий обеспечить обмен информацией с целевым микроконтроллером посредством стандартных параллельного или последовательного интерфейсов.

Компании, которые выпускают такие микроконтроллеры: Microchip, Zilog, Lantronix, Micrel, Connect One, Digi, Wiznet, ASIX и т. д.

Фирмы-производители применяют различный подход — начиная от полнофункциональных систем на кристалле до минималистских решений, обеспечивающих ограниченную функциональность сетевого соединения. Первые системы дороги и функционально избыточны, снабжены полноценной операционной системой и в силу этого подвержены различным вариантам интернет-атак. Вторые обеспечивают недостаточную производительность. Да и подобрать наиболее приемлемый вариант бывает зачастую проблематично, так как ассортимент, предлагаемый фирмами, не очень широк.

В качестве решения этой проблемы можно рассмотреть микросхемы компании Wiznet. Это целая серия, что позволяет выбрать наиболее подходящее решение в зависимости от интерфейса, необходимой производительности и уровня абстрагирования для решения задачи сетевой коммуникации. Микросхемы Wiznet максимально облегчают задачу реализации сетевого взаимодействия, так как протокол TCP/IP реализован в них аппаратно.

Разумеется, в конечном счете эти микросхемы представляют собой микроконтроллер, в котором прошита соответствующая программа. Но это и является преимуществом, так как, во-первых, эти микроконтроллеры выпускаются уже в течение долгого времени, они работают во множестве устройств в сетях с различной топологией и в связке с оборудованием различных производителей. Во-вторых, эти микросхемы обеспечивают работу от четырех до восьми сокетов одновременно, причем пользователь имеет возможность перераспределить располагаемое ОЗУ между сетевыми буферами в соответствии со своей необходимостью. В-третьих, применение двухпортовой памяти значительного объема (от 16 до 128 кбайт) дополнительно способствует повышению производительности за счет возможности выполнять одновременно операции чтения и записи. В-четвертых, уже реализованы все наиболее востребованные протоколы, и, кроме того, пользователь может сам дополнить функциональность необходимым протоколом, реализуя его в программе своего устройства. В-пятых, компания Wiznet поставляет разъемы RJ-45 со встроенными трансформаторами. (Есть вариант с дополнительно встроенными диодными мостами для обеспечения технологии PoE, то есть питания устройства через линию Ethernet. В последнее время такие решения становятся все более популярны.) Это обеспечивает полную совместимость микросхемы с примененным MagJack (в том числе и в небольших количествах).

В дополнение к отдельным микросхемам компания Wiznet выпускает мезонинные модули, подключение которых позволяет максимально упростить задачу реализации сетевого соединения, потому что такой модуль уже содержит все необходимые компоненты — сам микроконтроллер, цепи питания, систему тактирования, разъем с разделительным трансформатором и т. д.

В плане обеспечения максимальной производительности сетевого интерфейса наиболее подходят микросхемы W5300 и W5500.

Микросхема W5300 выпускается уже длительное время и хорошо себя зарекомендовала. Она может работать через интерфейс 16 или 8 бит и включает в себя 128 кбайт двухпортового ОЗУ, что обеспечивает выделение под каждый сокет (до восьми одновременно) по 8 кбайт на прием и 8 кбайт на передачу. Сравните с объемами памяти конкурентов — максимально 32 кбайт.

Для наиболее популярного в последнее время последовательного интерфейса у компании Wiznet до недавнего времени не было предложения в плане производительности (в основном из-за недостатков примененного SPI-интерфейса). Но недавно Wiznet выпустила новый чип W5500, за основу взяв кристалл от Nuvoton с архитектурой Cortex M0. Это позволило не только обеспечить высокоскоростной SPI с поддержкой расширенных режимов, но и значительно уменьшить стоимость микросхемы, а также дало возможность уменьшить ее энергопотребление. Микросхема имеет на борту 32 кбайт ОЗУ под сетевые буферы, поддерживает до восьми одновременно работающих сокетов и выпускается в 48-выводном LQFP-корпусе.

И традиционно на основе этой микросхемы компания выпустила мезонинный модуль Wiz550, содержащий все необходимое для обеспечения сетевого соединения. Внешний вид модуля представлен на рис. 4.

При его разработке компания оптимизировала его в ценовом отношении. (На рис. 4 видно, что на плате установлен отдельный раз-

Рис. 4. Внешний вид модуля Wiz550

вязывающий трансформатор, хотя в последнее время Wiznet применяла разъемы со встроенными трансформаторами, что, конечно, позволяло сделать модули более компактными, но одновременно увеличивало их стоимость.) Модули уже содержат запрограммированный MAC- и IP-адрес. В некоторых предыдущих продуктах модули не имели MAC-адреса, а покупка их диапазона под небольшие серии была проблематичной. Несмотря на применение отдельного трансформатора, размеры модуля составляют всего 30×60 мм.

Заключение

Достижение высокой производительности для встраиваемых систем — сложная задача, поэтому в статье даны рекомендации по выбору программного и аппаратного решений и основные правила, использование которых позволит успешно реализовать проект.

Литература

1. Олифер В., Олифер Н. Компьютерные сети: принципы, технологии, протоколы: Учебник для вузов. СПб.: Питер, 2010.
2. Иди Ф. Сетевой и межсетевой обмен данными с микроконтроллерами. М.: ИД «Додека-XXI», 2007.
3. Крамер Д. Э. Сети TCP/IP, принципы, протоколы и структура. М.: Вильямс, 2003.
4. **http://www.wiznet.co.kr/Sub_Modules/en/**
5. **http://www.iar.com/Global/Resources/Achieving TCP/IP performance in embedded systems.pdf**