Реализация стека TCP/IP на микроконтроллере Microchip PIC18

Стеки протоколов TCP/IP для 8-разрядных микроконтроллеров пополнились реализацией для Microchip PIC18. Отличие этой реализации от существующих состоит в том, что вся она выполнена на С. Оставив за рамками рассмотрения данной статьи вопросы применения TCP/IP на 8-разрядных микроконтроллерах, обратим внимание на возможность применения языка С и средств разработки для микроконтроллеров Microchip для реализации сложных проектов.

Сергей Емец

syemets@yandex.ru

Исходные коды и документация по реализации стека TCP/IP на микроконтроллере Microchip PIC18 доступны на сайте компании w ww.microchip.c om — application note 833 (ht tp://ww w.microchip.c om/1010/suppdoc/appnote/all/an833/index.h tm). Как можно видеть из документации и исходных кодов, проект был разработан одним человеком за достаточно короткое время. Также положительным свойством данной реализации является то, что проект может быть собран как MPLAB C18, так и HT-SOFT PIC18 компиляторами. То есть в данном случае имеется некая переносимость и независимость от средств разработки, представляемых производителем чипов. Следует заметить, что в отличие от большинства сложных проектов, поставляющихся производителями микросхем в качестве «application note», стек протоколов от Microchip не содержит ошибок и исполняемый файл (прошивка) собирается без проблем. Исходный текст сопровождается подробным и понятным документом (ht tp://w ww.microchip.c om/download/ appnote/internet/00833b.pdf), содержащим объяснение принципов работы сетевых протоколов, описанием особенностей реализации и опций конфигурации для реализации различных сетевых протоколов и приложений. Данная статья является по сути переводом этого документа. Цель, которую преследовал автор — изучение возможностей использования языка С для реализации сложного проекта на Microchip PIC18, а не применение стека на практике. Ответ, полученный автором, — комплект средств разработки для Microchip PIC18 позволяет создать, используя язык С, хорошо структурированный и документированный проект, который может быть передан от одного разработчика другому.

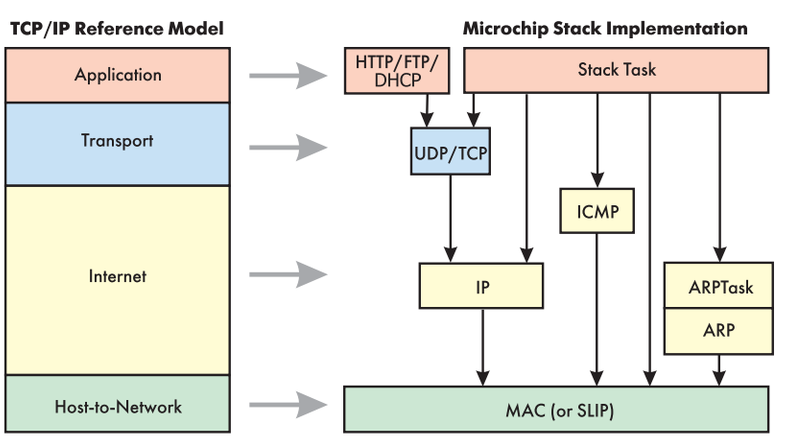

Рис. 1. Уровни TCP/IP-стека Microchip

При наличии отладочной платы PICDEM.net (ht tp://ww w.microchip.c om/download/tools/picmicro/demo/pdemnet/51240a.pdf), предоставленной фирмой «Гамма», и средств разработки: MPLAB 6.xx, MPLAB C18, HT-SOFT PIC18 — автору удалось собрать работоспособный веб-сервер, несколько отличный от прошитого в плату по умолчанию. Конечно, реализация стека TCP/IP для 8-разрядных микроконтроллеров из-за ограниченных ресурсов не может быть полнофункциональной и сопоставимой с распространенным в мире встраиваемых приложений стеком BSD. Следует заметить, что использование Linux-стека интернет-протоколов во встраиваемых системах ограничено GNU-лицензией, которая требует предоставления исходных кодов, что не всегда приемлемо. Поэтому разработчики встраиваемых систем работают со стеком BSD. Конечно, пока еще не существует 8-разрядных микроконтроллеров и средств разработки к ним, которые позволят использовать стек BSD, но, скорее всего, такие микроконтроллеры и не появятся, так как 8-разрядники ориентированы на задачи, требующие минимума ресурсов.

Описание протоколов, реализованных в стеке

Приведем постоянно встречающуюся картинку структуры стека интернет-протоколов (рис. 1).

Как обычно, в построении сетевых решений присутствуют уровни абстракции, начиная с физического уровня, который в данном случае может быть либо SLIP по встроенному последовательному порту, либо 10-мегабитным Ethernet, реализованным на чипе Realtek RTL8019AS, и заканчивая уровнями

приложений HTTP и FTP. Для выполнения конкретной задачи не требуется наличия всех уровней, например, при использовании SLIP не требуется протокол ARP. Из-за своей сложности некоторые протоколы реализованы не полностью. Также не обошлось без применения «хитростей», неизвестных автору (следует заметить, что автор не является специалистом по сетевым протоколам). Например, наряду с частичной реализацией DHCP, который служит для динамического выделения IP-адреса, предлагается более простой метод автоматического выделения адреса IP — IP Gleaning. Это нестандартное решение обеспечивает настройку IP-адреса платы PICDEM.net следующим образом: плата должна получить запрос PING, направленный на адрес платы Ethernet (MAC), IP-адрес, указанный в этом запросе, присваивается плате, и плата отвечает на полученный запрос (и все последующие обращения) с этим IP-адресом.

В рамках своих знаний об интернет-протоколах автор постарается описать функции, реализованные в стеке Microchip, и ограничения, являющиеся следствием скромных ресурсов микроконтроллера (как правило, критический ресурс в данной задаче — ОЗУ).

Описание проведем снизу вверх, рассматривая различные варианты параллельно. Всю работу со средой (MAC) обеспечивает RTL8019AS Network Interface Controller (NIC). Также в памяти сетевого контроллера хранятся пакеты и некоторые данные для протоколов верхних уровней. Альтернативный SLIP-протокол реализован не полностью и может использоваться только совместно с ПК под управлением операционной системы Windows. Следует обратить внимание, что функции, реализующие «физический» уровень (С-код), вызываемые с верхних уровней, имеют одинаковые имена и набор параметров, что позволяет реализовать механизм абстракции уровней и наглядно демонстрирует структуру стека. То есть приложение или верхние уровни стека — все, что лежит выше MAC/SLIP (рис. 1), не знают, какая среда используется для передачи данных. Таким образом, эти функции являются взаимоисключающими и линкер не позволит собрать приложение, содержащее два способа физической передачи данных. Для решения этого конфликта и выбора остальных опций конфигурации используется препроцессор. Это достаточно мощное средство управления кодом на этапе компиляции. Фактически препроцессор С изменяет исходный код перед вызовом компилятора, и при установке различных опций компилируется различный код. Для микроконтроллера с ограниченными ресурсами такой механизм конфигурации позволяет избежать дополнительных затрат времени и памяти по сравнению с проверками во время исполнения. Также это позволяет собрать все изменения, требующиеся для различных конфигураций, в одном файле. В данном проекте этот файл StackTsk.h и выбор способа доступа к среде определяется константой STACK_USE_SLIP. То есть, если присутствует директива #define STACK_USE_SLIP — стек использует SLIP, если эта директива закомментирована, то используется Ethernet. В данной статье не приводятся все конфигурационные константы, полная таблица приведена в документе к AN833.

Следующим уровнем является ARP — протокол разрешения адреса, который ставит соответствие адреса Ethernet (MAC) и IP. Результатом действия этого протокола является динамически создаваемая таблица, содержащая пары адресов. Это одно из мест, в котором может ограничиваться количество соединений, поддерживаемое стеком одновременно. Для экономии памяти предопределенное число 5 может быть уменьшено. При использовании соединения SLIP протокол ARP не нужен, что дает экономию оперативной памяти.

Следует заметить, что протокол ARP управляется автономной задачей (ARPtsk), что будет рассмотрено в описании реализации многозадачности.

Следующий протокол — ICMP — отвечает за поддержку управляющих сообщений. Например, этот протокол обрабатывает пакеты PING. ICMP тесно связан с основным протоколом стека — IP, который управляет маршрутизацией пакета и принимает решение на какой шлюз следует посылать сообщение для хоста, имеющего данный IP-адрес. Из-за ограниченного времени на подготовку статьи, автор не разобрался в том, как работает алгоритм маршрутизации в стеке от Microchip. Так как механизмы маршрутизации достаточно сложные и требуют много памяти для построения таблиц, скорее всего предлагается сильно «облегченная» реализация. Но следует предположить, что вряд ли кто-нибудь будет разрабатывать шлюз или роутер на основе Microchip PIC18. По крайней мере, в данной реализации IP-протокола создаются уникальные пакеты, которые могут передаваться по сетям с возможностью прохождения по нескольким маршрутам и дублированием.

Над IP расположены протоколы TCP и UDP, которые служат непосредственно для передачи данных и образуют «транспортный» уровень. Протокол TCP наиболее известен и часто используется в приложениях. Этот протокол создает так называемые «сокеты», через которые может осуществляться непрерывная (потоковая) передача данных. При этом протокол генерирует непрерывный поток данных независимо от того, что IP-пакеты, которыми пользуется TCP, могут теряться, или нарушается последовательность их приема. Очевидно, что чем больше длина буфера для приема таких пакетов, тем меньше требуется повторов и тем выше скорость передачи. В данной реализации может быть открыто несколько «сокетов» (что дает возможность установить несколько TCP-соединений), но из-за экономии памяти «сокеты» используют общую память и их максимальное число достаточно мало. Автору не удалось поэкспериментировать с несколькими соединениями и нарушениями последовательности IP-пакетов (то есть реально проверить работу TCP/IP), но в небольшой локальной сети, где вероятность потери пакета и нарушения порядка следования мала, стек Microchip успешно поддерживал несколько соединений. Так как в TCP существует функция таймаута, то требуется поддержка часов — для этого есть специальная функция учета времени. Более простой в реализации протокол UDP обеспечивает передачу отдельных пакетов. При этом упрощением UDP данного стека является нулевая контрольная сумма, что перекладывает задачу сохранения целостности информации на приложение. Также из-за экономии ОЗУ используются общие буферы (перекрываются с TCP), и приложения должны извлекать информацию из приемного буфера до того, как придет следующий пакет. Это ограничение не вызывает сложностей в случае использования невытесняющей многозадачности. Но в случае большого количества задач и длительного времени выполнения возможна потеря пакетов.

DHCP является протоколом высокого уровня, но непосредственно из приложений не вызывается, его предназначение — периодически получать от специального сервера информацию для работы протоколов IP и ARP. Существует отдельная задача, которая обеспечивает динамическую установку IP-адреса платы, IP-адреса шлюза и маски подсети. Для работы DHCP используется протокол UDP, который должен быть разрешен и должен поддерживать требуемое количество «сокетов».

Для работы протоколов HTTP и FTP требуется TCP. Эти протоколы реализованы с ограничениями, но для решения задач, которые могут быть поставлены перед платой PICDEM.net, они достаточны. Реализация файловой системы и поддержки CGI будет описана ниже.

Описание структуры программы

Так как реализация стека не использует какую-либо многозадачную ОС, то управление работой стека возлагается на пользователя. В качестве примера управления в составе (AN833) приводится StackTsk.c, в котором содержится менеджер задач, реализующий невытесняющую многозадачность. На приложения пользователя это накладывает ряд ограничений: нельзя использовать циклы с ожиданием события, длинные вычисления должны быть разбиты на несколько этапов и т. п. Приложение должно достаточно часто проверять наличие пакетов в буфере NIC-контроллера и обрабатывать их следующим образом:

```
If a data packet received then
    Get data packet protocol type
        If packet type is IP then
            Fetch IP header of packet
            Get IP packet type
            if IP packet type is ICMP then
                Call ICMP module
            else if IP packet type is TCP then
                Call TCP module
            else if IP packet type is UDP then
                Call UDP module
            else
                Handle not supported protocol
            End If
        End If
    Else if packet type is ARP then
        Call ARP module
    End If
End If
```

В AN833 приводятся следующие размеры модулей после компиляции (табл. 1).

Таблица 1. Использование памяти при компиляции HI-TECH PICC18

Modul	Progra Memory (words)	Data Memory (bytes)
MAC (Ethernet)	906	5[1]
SLIP	780	12[2]
ARP	392	0
ARPTask	181	11
IP	396	2
ICMP	318	0
TCP	3323	42
HTTP	1441	10
FTP Server	1063	35
DHCP Client	1228	26
IP Gleaning	20	1
MPFS[3]	304	0
Stack Manager	334[4]	12+ICMP Buffer

1. Используется память NIC RNL8019AS.
2. Не учитывается память, требуемая для приемного и передающего буфера, объем которой определяется пользователем.
3. При хранении в памяти микроконтроллера.
4. Максимально возможный. Реальный объем может быть другим.

Автор использовал компилятор MPLAB C18 и получил несколько другие данные, но требуемые размеры не превышали имеющуюся в кристалле память. Также возможно, что при различных опциях компилятора исполнимый код и требования ОЗУ будут получаться различных размеров.

Для обеспечения динамического обновления веб-страниц и взаимодействия с пользователем «хитрым» образом используется механизм CGI. Для динамического отображения информации, реализованного в AN833, в CGI-

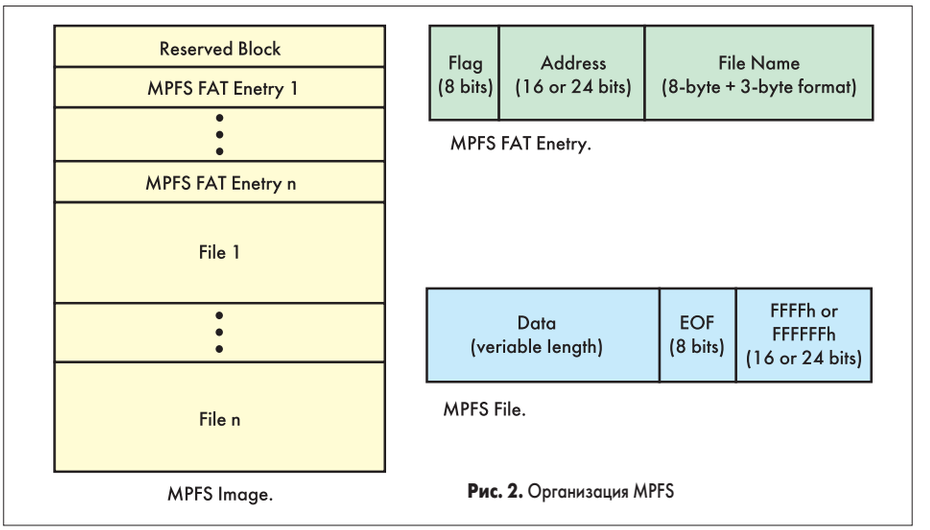

Рис. 2. Организация MPFS

файле должна быть строка %xx, где xx — номер переменной (0–99). Если HTTP-сервер встречает такую строку, то вызывает функцию, определенную пользователем (в коде приложения) HTTPGetVar. Эта функция изменяет байт (механизм передачи указателя), который затем передается HTTP-сервером клиенту. Функция позволяет передать как один байт, так и строку символов (функция вызывается сервером и передает по одному символу до тех пор, пока строка не закончится). Также механизм CGI позволяет выполнить действия в ответ на сообщение пользователя. Это делается с помощью функции HTTPExecCmd. Аргументами этой функции являются указатель на строки и число, определяющее количество строк (как в стандартной C-функции main). Это позволяет передать любое количество аргументов (ограниченное только размером ОЗУ) в функцию. В прилагаемом в AN833 примере используются оба метода.

Описание простейшей файловой системы

HTTP-сервер работает с файлами (веб-страница и ее содержание — «контент»). Microchip HTTP-сервер распознает по расширению следующие типы файлов: «.txt», «.htm», «.gif», «.cgi», «.jpg», «.cla» и «.wav». Для хранения этих файлов должна существовать файловая система. В исходном коде к рассматриваемому примеру Microchip предлагает простейшую файловую систему MPFS — она создается один раз и хранится в ПЗУ (либо во внутреннем PIC18, либо во внешнем EEPROM). Для создания файловой системы прилагается специальная утилита — mpfs, а для доступа к файлам во время исполнения кода микроконтроллером предлагаются процедуры, исходный код которых хранится в файле MPFS.c. Организация файловой системы показана на рис. 2.

Microchip FTP-сервер позволяет закачать целый образ MPFS в память платы. То есть сервер поддерживает только команду put, а доступ к отдельным файлам отсутствует. В идеале, пример должен позволить закачивать веб-странички по FTP, но автору не удалось добиться такой работоспособности.

В прилагаемом примере предлагается такая файловая система:

```
index.htm
main.htm
form.htm
commands.htm
status.cgi
led1.gif
led0.gif
mchp.gif
```

Эти файлы формируют веб-страницу и совместно с исходными кодами демонстрируют, как создавать динамически обновляемые веб-страницы. Также через предлагаемый веб-интерфейс можно управлять ресурсами платы — зажигать и гасить светодиоды на плате.

Выводы

Данный проект является хорошим учебным пособием по разработке сложных структурированных проектов на языке C для Microchip PIC18. Его также можно рекомендовать для начального освоения структуры сетевых протоколов и как пример реализации работоспособного стека TCP/IP. Заслуживает внимания алгоритм реализации невытесняющей многозадачности, используемый в проекте, который при незначительной доработке может быть использован для решения задач, требующих детерминированного времени отклика (real time). Кроме того, может оказаться полезной реализация простейшей файловой системы — как функции для ее работы, так и утилита для создания.

Можно сказать, что это удачный и полезный пример (application note) реализации контроллеров Microchip PIC18. Наверное, наиболее полезным этот пример окажется для применения во встраиваемых системах, работающих с сетью (HTTP или FTP), так как компания Microchip не запрещает использовать этот код в разработках (по крайней мере, автор не обнаружил каких-либо лицензионных ограничений на применение кода).