

Работа со встроенным TCP/IP-стеком модулей GSM/GPRS серии SIM800

компании SIMCom Wireless Solutions

Данная статья адресована разработчикам встраиваемых решений, а именно — программистам, решающим задачу передачи данных на удаленный сервер по сети сотовой связи при помощи GSM/GPRS-модуля серии SIM800 производства SIMCom Wireless Solutions. В статье дан обзор возможностей встроенного TCP/IP-стека модулей серии SIM800, описан порядок правильной работы с ним, рассмотрены различные внештатные ситуации и способы правильной их обработки. Статья основана на официальной инструкции по применению устройств и системы AT-команд, а также включает обобщенный опыт работы с GSM-модулями в реальных сетях сотовой связи.

Батор Батуев
bator.batuev@sim.com

Компания SIMCom Wireless Solutions, ведущий разработчик и производитель GSM/GPRS, 3G, LTE и GPS/ГЛОНАСС-решений для M2M-отрасли, предлагает новую линейку GSM/GPRS-модулей серии SIM800 на замену хорошо известной в мире серии SIM900. В новых модулях улучшен ряд ключевых качеств, таких как массо-габаритные параметры, скорости передачи данных, цена. Кроме того, в новой линейке реализована поддержка Bluetooth и множества сетевых протоколов передачи данных, включая SSL.




Серия GSM/GPRS-модулей SIM800

В России линейка SIM800 (табл. 1) представлена модулями SIM800C, SIM800C-DS, SIM800H, SIM800, SIM800F [1]. Все они покрывают подавляющее большинство потребностей рынка M2M. Так, SIM800C в популярном корпусе LCC является фокусным и базовым решением для широкого спектра приложений. SIM800C-DS — самый малогабаритный в мире GSM/GPRS-модуль с под-

держкой двух SIM-карт (Dual SIM Dual Standby). Этот модуль нашел применение в охранном секторе и приложениях, где необходимо мгновенное переключение между сетями сотовой связи и требуется разделить балансы конечного потребителя и интегратора. Модуль SIM800H аналогичен модулю SIM800C по техническим характеристикам, но его основные потребители — те, кому нужна поддержка передачи данных CSD и кого не смущает корпус LGA, подразумевающий пайку в печи. Модули SIM800F и SIM800 (24×24 мм) не выделяются какими-либо особенными качествами, их основная задача — замена модулей SIM900R или SIM900 в текущих проектах и упрощение освоения серии SIM800 для новых разработок. Генеральное отличие между SIM800 и SIM800F заключается лишь в том, что первый поддерживает CSD, а второй, как SIM800C и SIM800C-SD, — нет.

Следует подчеркнуть, что GSM/GPRS-модули серии SIM800 поддерживают беспроводную технологию связи Bluetooth 3.0 Classic (профили SPP, HFP и проч.) на аппаратном уровне.

Таблица 1. Линейка GSM/GPRS-модулей серии SIM800

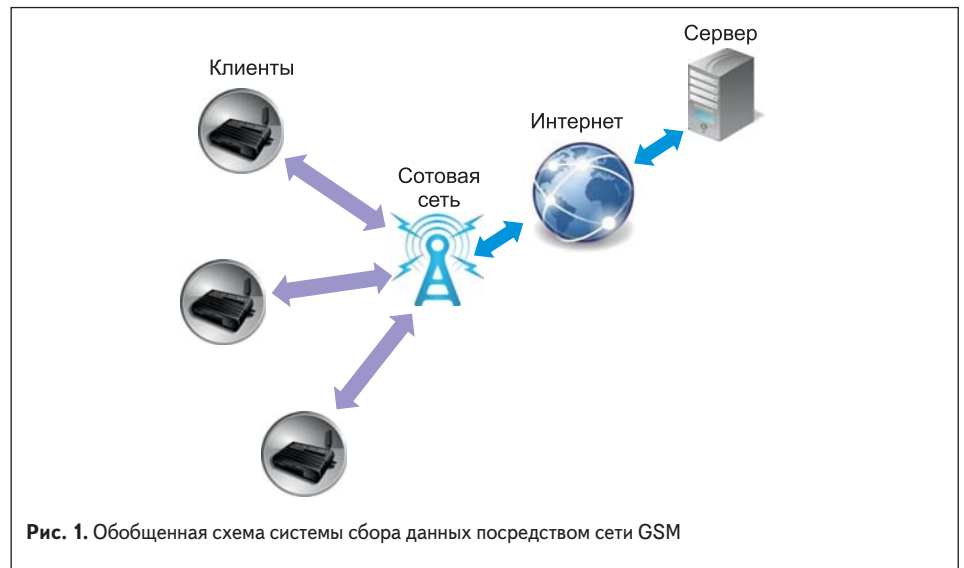
Модуль						
	SIM800C	SIM800C-DS	SIM800H	SIM800	SIM800F	
Корпус	LCC	LCC+LGA	LGA	LCC		
Размер, мм	15,7×17,6		15,8×17,8	24×24		
GPRS	85,6 кбит/с					
Bluetooth	3.0 (кроме SIM800C-DS)					
Количество SIM-карт	1	2	1			
Функции	Voice Call, USSD, SMS, CSD (только SIM800H и SIM800), DTMF, Jamming Detection, MMS, Audio R/F, Embedded AT (опционально)					
Поддерживаемые протоколы передачи данных	PPP, TCP/IP, UDP/IP, FTP, HTTP, SMTP, POP3, SSL					
Интерфейсы	2×UART, USB		2×UART, USB, I ² C, SPI			
Аудио	MIC, SPK 32 Ом	PCM, MIC, 2×SPK 8/32 Ом	PCM, 2×MIC, 2×SPK 8/32 Ом		PCM, MIC, SPK 32 Ом	
Память пользователя	Внутренняя	Внутренняя, SD-карта (<32 Гбайт)				Внутренняя
Рабочее напряжение, В	3,4–4,4					
Диапазон рабочих температур, °C	–40...+85					

Поддержка Bluetooth требует соответствующего программного обеспечения (ПО). Данная функция, несколько не увеличивая стоимость изделия, дает пользователю уникальные возможности: голосовые вызовы посредством стандартной беспроводной гарнитуры, обмен произвольными данными на расстояниях в десятки метров, файлами и контактами из записной книжки — все то, для чего предназначен Bluetooth, но в разрезе задач IoT. Также разработчику следует знать, что модули серии SIM800 поддерживают технологию Embedded AT. Она позволяет интегрировать пользовательский Си-код в операционную систему (ОС) модуля и управлять всеми его ресурсами: SMS, голосовые вызовы, выход в Интернет, управление интерфейсами UART, SPI, I²C, GPIO и проч. Это весьма полезная технология, широко применяемая, когда остро стоит вопрос о стоимости и/или размере конечного изделия. Подробнее о работе Bluetooth и Embedded AT в модулях серии SIM800 можно узнать отдельно из руководств по применению или у инженеров технической поддержки компании и дистрибьюторов. В данной же статье речь пойдет о работе со встроенным TCP/IP-стеком.

Разработчику не нужно знать/помнить принципы сетевых протоколов, тайминги, инкапсуляцию и т. д. (рис. 2), не нужно обрабатывать кадры, пакеты и сегменты данных. При работе со встроенным TCP/IP-стеком модуля хост

имеет дело лишь с потоком полезных данных (рис. 3), обмениваясь ими с сервером на прикладном уровне стека протоколов.

GSM/GPRS-модуль, управляемый хостом через последовательный порт UART, берет



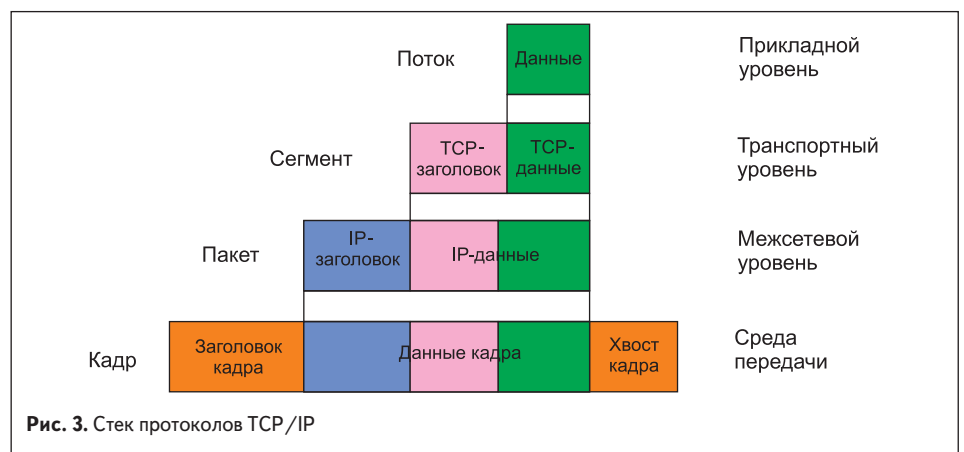
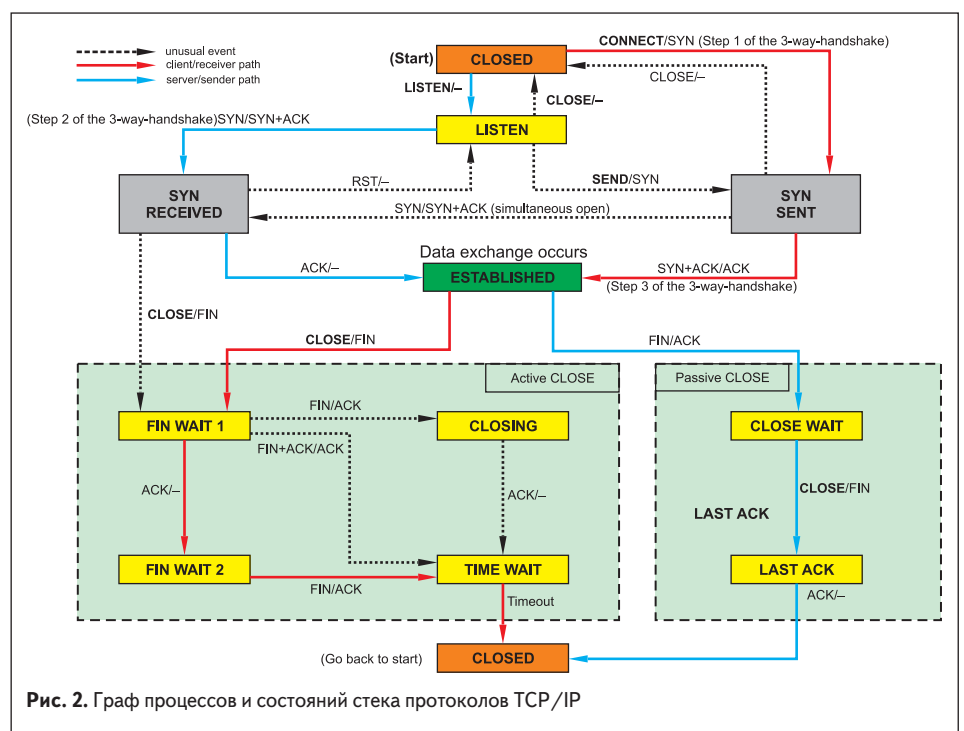
Протокол TCP/IP в M2M

Обобщенная схема любой системы сбора данных построена на принципах клиент-серверного подключения (рис. 1). То есть где-то в сети есть сервер, который ждет подключения M2M-устройств и принимает от них полезные данные (расход топлива, координаты перемещения объекта, температура, тревожный сигнал, количество потребленной электроэнергии и т. п.) в зависимости от приложения и задач, выполняемых системой. В качестве клиентов в такой схеме выступают устройства на базе GSM/GPRS-модуля и управляющий контроллер (хост): модуль предоставляет доступ в Интернет, а хост управляет этим процессом. Такие клиент-серверные соединения для передачи данных, как правило, используют протокол TCP/IP.

Опытные разработчики знают, что посредством GSM/GPRS-модуля можно выйти в Интернет и подключиться к серверу двумя способами — при помощи протокола канального уровня PPP или встроенного протокола TCP/IP. Оба варианта доступны в GSM/GPRS-модулях серии SIM800.

Когда в распоряжении клиентской части имеются продвинутые аппаратные ресурсы хоста и ОС типа Linux или Android, для выхода в Интернет часто применяют протокол PPP. Протоколы до уровня приложений при этом реализованы в самой ОС хоста. Но это относится к небольшому числу случаев. В подавляющем большинстве M2M-приложений ставятся жесткие требования к стоимости проекта, который подразумевает недорогой хост с небольшой памятью и простейшей ОС (без особых изысков). Реализация собственного стека протоколов (настройка над PPP) в таком случае часто выводит проект за временные рамки сдачи работ и рамки бюджета единицы изделия.

В таких случаях более оправданным является применение встроенного стека TCP/IP-модуля, т. к. значительно упрощается процесс отладки и удешевляется стоимость конечного изделия.



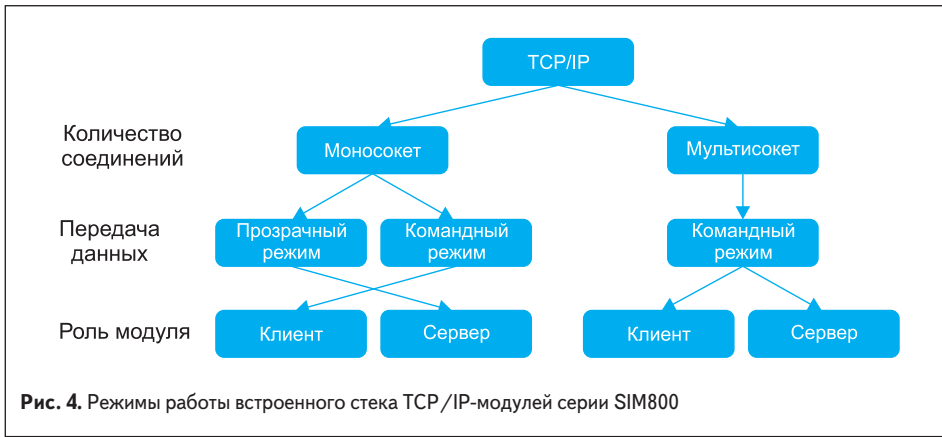


Рис. 4. Режимы работы встроенного стека TCP/IP-модулей серии SIM800

на себя общение с сетью сотовой связи и работу в IP-сети. Хосту для этого предоставляется командный интерфейс, т. е. доступ к сетевым функциям происходит посредством простой и понятной системы AT-команд [4]. При этом все сложные сетевые процессы, такие как активация контекста, открытие сессии, передача данных, закрытие сессии и т. д., скрыты от хоста.

Кстати, надо сказать, что протокол TCP/IP, наряду с другими протоколами, к примеру UDP/IP, очень удобен для систем, требующих надежности передачи данных, т. к. в его основе заложены принципы подтверждения и повтора посылок в случае потери пакетов. Прикладному уровню не нужно заботиться об организации повторного запроса данных.

Возможности встроенного стека протоколов TCP/IP в GSM/GPRS-модулях серии SIM800

Встроенный стек модуля гибок и может быть настроен на различные режимы работы в зависимости от пожеланий и фантазии разработчика. На рис. 4 схематически указаны режимы работы встроенного стека. Они определяют роль модуля по отношению к удаленной стороне (клиент/сервер), способ

передачи данных и количество одновременно открытых соединений.

Режимы работы встроенного стека настраиваются в момент инициализации, до активации контекста и установления соединения, при помощи следующих AT-команд:

- **AT+CIPMUX=<n>**, где **<n>=0** — моносокеты, **<n>=1** — мультисокеты;
- **AT+CIPMODE=<n>** где **<n>=0** — командный режим передачи данных, **<n>=1** — прозрачный.

Так, в режиме мультисокета модуль может открыть и поддерживать до шести одновременных соединений или работать только с одним соединением (моносокеты).

Как видно, есть два способа обмена данными с удаленной стороной: прозрачный и командный режимы. В прозрачном режиме данные, находящиеся на стороне клиента, во время открытой сессии передаются и принимаются в последовательный порт UART модуля в том же виде, в каком их «видит» на удаленной стороне сервер. В некоторых случаях предпочтительней командный режим, при котором GSM/GPRS-модуль настроен на прием только AT-команд, и, чтобы осуществить отправку данных на сервер или принять их, потребуется подавать данные в куле с AT-командами. Кстати,

мультисокеты исключают работу модуля в прозрачном режиме передачи данных.

На рис. 4 показано, что GSM/GPRS-модуль может быть настроен на выполнение роли клиента или сервера. Роль модуля не задается AT-командой, как количество соединений или режим обмена данными, а определяется способом открытия соединения. В дальнейшем мы будем рассматривать вариант открытия одного соединения (моносокеты) с удаленным сервером (роль модуля — клиент). Передачу данных рассмотрим как в командном, так и в прозрачном режиме. Но перед тем как перейти к практическим примерам, надо разобраться с механизмом работы встроенного стека TCP/IP.

Механизм работы встроенного стека TCP/IP описывается диаграммой состояний, показанной на рис. 5. Это диаграмма состояний для односокетного соединения. Как видно, всего состояний 10. Все они, от IP INITIAL до PDP DEACT, замыкают цикл до активации контекста и открытия соединения до закрытия соединения и деактивации контекста — именно в таком порядке, в нормальном случае. Состояние стека можно контролировать командой **AT+CIPSTATUS** (без параметров).

Начальное состояние стека после инициализации **AT+CIPMUX** и **AT+CIPMODE** — IP INITIAL, оно означает, что GPRS-контекст не настроен. Во время активации контекста информация направляется в сеть GPRS и может служить условием доступа к услугам пакетной передачи данных. Сегодня многие операторы сотовой связи дают доступ в GPRS независимо от того, какой контекст был задан. Однако этап настройки контекста пропускать не следует.

Настройка контекста осуществляется по команде **AT+CSST=<APN>,<USR>,<PASS>**, где **<APN>** — точка доступа, **<USR>** — логин, **<PASS>** — пароль. Эти параметры можно получить от поставщика мобильной связи. После этой команды встроенный стек принимает состояние IP START. Кстати, если говорить о повторном открытии сессии (питание не отключалось), то команду **AT+CSST** можно подавать без параметров. Модуль примет в исполнение ранее заданные параметры и приведет встроенный стек в верное состояние.

После успешной настройки контекста его следует активировать командой **AT+CIICR** (без параметров). С момента активации контекста модуль получает доступ в сеть GPRS и на стороне оператора сотовой связи начинается отсчет трафика. За этим этапом следует переход встроенного стека в состояние IP STATUS посредством запроса своего IP-адреса в сети командой **AT+CIFSR** (без параметров).

Итак, к примеру, мы имеем сервер с IP-адресом 192.168.123.123 и открытым портом 1234. Для подключения к этому серверу должна быть исполнена команда **AT+CIPSTART="TCP", "192.168.123.123", 1234**. Важно, чтобы состояние стека перед подачей этой команды было IP STATUS. В противном случае связь не будет установлена. Кстати, допускается вместо IP-адреса задавать доменное имя, например: **AT+CIPSTART="TCP", "www.simcomm2m.com", 1234**.

Когда соединение с сервером установлено, состояние стека приобретает статус CONNECT

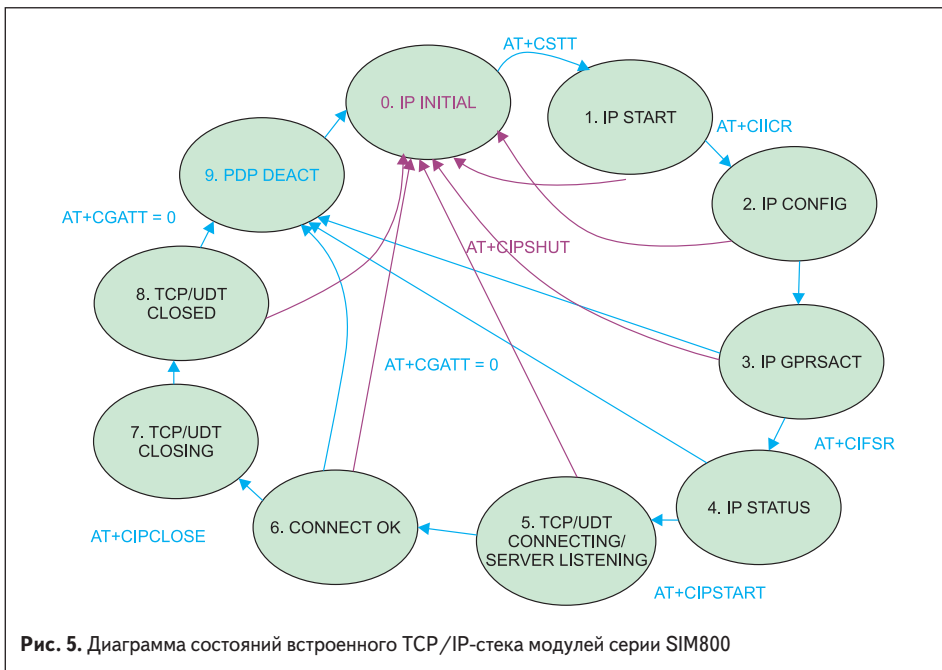


Рис. 5. Диаграмма состояний встроенного TCP/IP-стека модулей серии SIM800

ОК. Теперь между сервером и хостом установлен канал передачи данных уровня приложений. После того как обмен данными будет завершен, соединение с сервером можно будет закрыть командой *AT+CIPCLOSE*. При этом есть варианты закрытия:

- Штатное закрытие соединения при помощи команды *AT+CIPCLOSE* (без параметров) или *AT+CIPCLOSE=0*, которое проходит с отправкой командных пакетов в адрес сервера и ожиданием от сервера подтверждения закрытия (рис. 6). Это согласованное закрытие сокета.
- Быстрое закрытие при помощи команды *AT+CIPCLOSE=1*, которое подразумевает перевод состояния стека в состояние TCP CLOSED без уведомления сервера. Этот случай актуален при потере связи с сервером или с GPRS-сетью. Этот способ нужен, чтобы вернуть стек в предсказуемое состояние, не тратя время на ожидание подтверждения, которое может и не прийти.

После закрытия соединения с сервером GPRS-контекст все еще активен. Его следует закрыть командой *AT+CIPSHUT* (без параметров). После этого стек модуля переходит в начальное состояние IP INITIAL, и он готов к новой сессии.

Обработка исключительных случаев

Все команды GSM/GPRS-модулей серии SIM800 имеют время исполнения. Разработчику ПО хоста важно знать время исполнения отдельно для каждой команды, чтобы исключить бесконечное ожидание реакции на команду (открытие соединения, к примеру). Значения максимального времени исполнения задокументированы, их можно найти в системе команд GSM/GPRS-модуля [4]. В таблице 2 указаны значения максимального времени исполнения основных команд встроенного TCP/IP-стека. Как видно, некоторые команды исполняются десятки секунд. Это объясняется зависимостью этих команд от быстродействия сети и сервера.

Получается, некоторые команды могут исполняться несколько минут, прежде чем можно будет понять, что что-то пошло не так. В M2M такие задержки, конечно, недопустимы. Как же обрабатывать случаи, когда время исполнения команды затянулось, а реакции так и не последовало? Все зависит от того, на каком этапе установления соединения произошел сбой (ошибка или вышел таймаут) и в каком состоянии находится стек (рис. 6). Причин сбоя может быть несколько, и реакция может быть разная, но главное — вернуть встроенный стек в исходное состояние IP INITIAL или IP STATUS. Рассмотрим на примере несколько случаев:

Таблица 2. Максимальное время исполнения команд встроенного стека TCP/IP

Команда	Максимальное время исполнения, с
CICR	85
CIPSTART	160
CIPSEND	645
CIPCLOSE	120
CIPSHUT	65

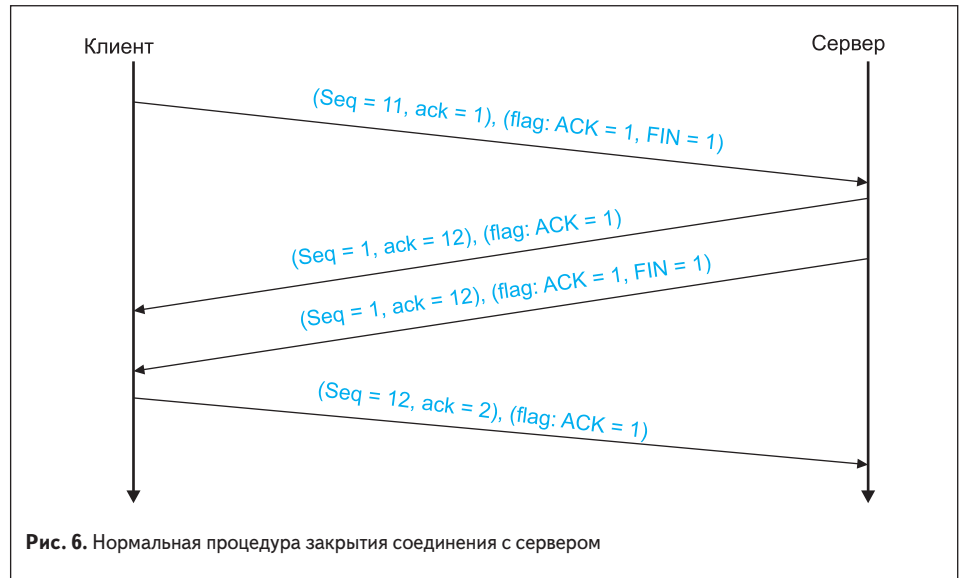


Рис. 6. Нормальная процедура закрытия соединения с сервером

1. Сервер вышел из строя или доступ в Интернет ограничен (потеря пакетов, высокий пинг и проч.). В этом случае все команды из таблицы 1 приведут к длительному времени исполнения. Чтобы повторить попытку соединения с этим или другим сервером, следует перед этим закрыть сокет командой *AT+CIPCLOSE=1*. При этом деактивировать контекст командой *AT+CIPSHUT* не обязательно.
2. Потеря связи с GSM-сетью. Такое возможно в местах плохого покрытия сети, из-за ухудшения условий приема сигнала или внезапной выемки SIM-карты из прибора. Здесь следует проверить готовность SIM-карты (*AT+CPIN?* или чтение ячейки памяти командой *AT+CMGR*), уровень сигнала (*AT+CSQ*), наличие регистрации в сети (*AT+CREG?*) и доступ к услугам GPRS (*AT+CGATT?*). Если физический доступ к GSM-сети пропадет после или во время открытия сессии командой *AT+CIPSTART*, то придется закрыть соединение (*AT+CIPCLOSE=1*), деактивировать контекст и восстанавливать соединение с начала, сразу после того как будут успешно проверены SIM-карта, уровень сигнала, регистрация в сети и доступ к услугам GPRS.
3. Истек срок жизни контекста. Когда открывается контекст, сеть выделяет определенные ресурсы на его поддержание. Операторы сотовой связи не допускают «мертвые» контексты, когда ресурс занят, а обмена данными в этом контексте нет. Если обмена данными нет, то через некоторое время оператор деактивирует контекст. У разных операторов это время разное — примерно от трех до семи минут. Модуль при этом в порт UART выдаст уведомление: +PDP DEACT. Его нужно обработать и сбросить встроенный стек в исходное состояние командой *CIPSHUT*. Однако иногда в некоторых приложениях требуется поддерживать контекст в активном состоянии. Для этого можно периодически обмениваться с сервером пустыми данными, типа эха. Но это неудобно в реализации. Взамен можно воспользоваться функцией поддержания соединения командой *AT+CIPTCPKA* [4].

4. Нагрузка на сеть GSM. Всем известно, что GPRS-услуги и голосовая связь делят общие ресурсы. GPRS всегда выделяется оператором по остаточному принципу, а у голосовых соединений наивысший приоритет. Контекст может быть деактивирован оператором принудительно. Внешне данный случай выглядит как предыдущий (п. 3), и обрабатывать его следует аналогично.

Следует предусмотреть случай, когда переинициализация соединения не дает желаемого эффекта. В этом случае рекомендуются штатное выключение/включение модуля и повторная попытка восстановить соединение с самого начала.

Работа со встроенным стеком протоколов TCP/IP в GSM/GPRS-модулях серии SIM800

Перейдем к практике применения встроенного стека TCP/IP на примере GSM/GPRS-модуля SIM800C, а точнее, на примере отладочного набора (рис. 7). В качестве сервера для на-



Рис. 7. Отладочный набор для GSM/GPRS-модуля SIM800C

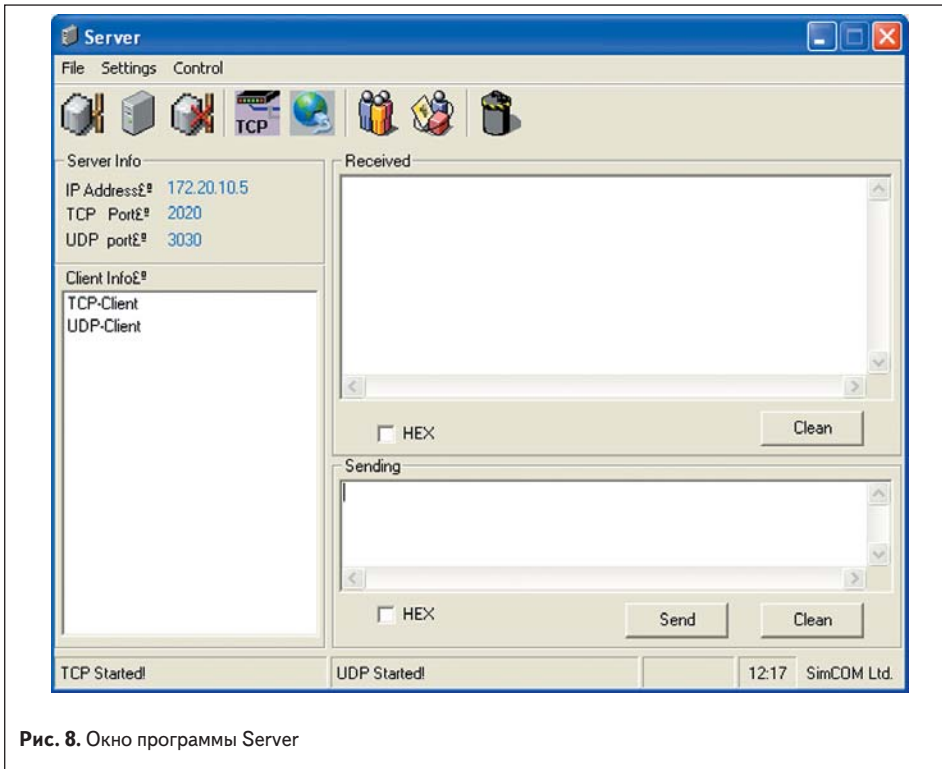


Рис. 8. Окно программы Server

глядности будем использовать ПК с внешним статическим IP-адресом и учебную программу Server (рис. 8) — примитивный TCP-сервер.

При запуске программы сервер уже готов к работе. В окне программы всегда отображается информация об IP-адресе сервера и номере порта для подключения. Эту информацию мы и используем при открытии соединения на стороне SIM800C. Также в окне программы есть поля Received (для отображения принятых данных от удаленной стороны, от SIM800C) и Sending (для ввода данных в сторону удаленной стороны, к SIM800C). Данные по умолчанию отправляются/принимаются в формате ASCII-символов. Если требуется передавать произвольные бинарные данные, то для этого нужно поставить галочку возле параметра HEX.

Итак, ниже приведен рабочий AT-лог, который можно применить на практике. После каждой AT-команды для модуля SIM800C идут ответы и сообщения от модуля. Данный лог демонстрирует инициализацию стека, настройку контекста и открытие соединения с удаленным сервером, а также различные способы обмена данными в командном и прозрачном режимах.

Инициализация

```
AT+CPIN? // Проверка готовности SIM-карты.
+CPIN: READY
OK
AT+CSQ // Уровень сигнала RSSI = 20 ед.
(примерно -73 дБм), удовлетворительный уровень.
+CSQ: 20,0
OK
AT+CREG? // Проверка наличия регистрации в сети GSM.
+CREG: 0,1
OK
AT+CGATT? // Проверка доступа к услугам пакетной
передачи данных.
+CGATT: 1
OK
```

```
AT+CIPMODE=0 // Командный режим передачи данных.
OK
AT+CIPMUX=0 // Моносокет.
OK
```

Настройка контекста и открытие соединения

```
AT+CIPSTATUS
OK
STATE: IP INITIAL
AT+CSST="internet" // Настройка точки доступа.
OK
AT+CIPSTATUS
OK
STATE: IP START
AT+CIICR // Активация контекста.
OK
AT+CIPSTATUS
OK
STATE: IP GPRSACT
AT+CIFSR
100.91.34.225
AT+CIPSTATUS
OK
STATE: IP STATUS
AT+CIPSTART="TCP", "81.95.20.18", 2020 // Открытие
соединения с удаленным сервером.
OK
CONNECT OK // Сообщение об успешном открытии соединения.
AT+CIPSTATUS
OK
STATE: CONNECT OK
```

Передача данных с подтверждением в командном режиме

```
AT+CIPSEND? // Проверка максимальный размер
данных, которые можно послать в сторону удаленной
стороны. Этот размер зависит от сети.
+CIPSEND: 1460
OK
```

```
AT+CIPQSEND? // Нормальный режим передачи данных.
В этом режиме каждая порция высланных данных под-
тверждается сообщением «SEND OK», что означает, что
сервер данные принял и подтвердил их получение.
+CIPQSEND: 0
OK
```

```
AT+CIPSEND=100 // Передача 100 байт данных.
> // Приглашение. hellohellohellohellohellohellohello
hellohellohellohellohellohellohellohellohello //
Размер данных не должен быть больше 1460 байт,
и в конце блока данных следует байт 0x1A.
SEND OK // Данные успешно переданы.
AT+CIPSEND // Передача данных произвольного раз-
мера.
> hellohellohellohellohellohellohellohellohellohello
hellohellohellohellohellohellohello // Размер данных
не должен быть больше 1460 байт, и в конце блока данных
следует байт 0x1A.
SEND OK
```

Быстрая передача данных в командном режиме

```
AT+CIPQSEND=1 // Режим быстрой передачи данных.
Этот режим подразумевает передачу данных без ожидания
от сервера подтверждения о получении.
```

```
OK
AT+CIPSEND=100 // Передача 100 байт данных.
> hellohellohellohellohellohellohellohellohellohello
llohellohellohellohellohellohellohellohellohello
DATA ACCEPT:100 // Модуль принял данные в свой
буфер и вышлет их в сторону сервера в фоновом
режиме.
AT+CIPACK // Проверка: 300 байт передано на сервер,
из них 300 байт сервером приняты и подтверждены.
+CIPACK: 300,300,0
OK
AT+CIPSEND // Передача данных произвольного раз-
мера происходит аналогичным образом.
> hellohellohellohellohellohellohellohellohellohello
llohellohellohellohellohellohellohellohellohello
DATA ACCEPT:100
AT+CIPACK
+CIPACK: 400,400,0
OK
AT+CIPQSEND=0 // Нормальный режим передачи
данных.
OK
```

Прием данных в командном режиме, автоматический вывод принятых данных

```
AT // Модуль находится в командном режиме;
OK
AT
OK
HelloHelloHelloHelloHelloHelloHelloHelloHello //
Данные, принятые от сервера, выводятся из порта
UART модуля автоматически. Данные выводятся «как
есть», и это неудобно, поэтому будут полезны следую-
щие настройки.
AT
OK
AT+CIPHEAD=1 // Перед блоком данных, принятых
от сервера, добавлять заголовок формата +IPD,<длина
блока данных>.
OK
AT+CIPSRIP=1 // При приеме данных показы-
вать уведомление в виде RECV FROM:<IP адрес
отправителя>,<порт>.
OK
```

```
AT+CIPSHOWTP=1 // Показывать тип протокола
в уведомлении +IPD,<длина блока данных>,<тип про-
токола>.
OK
RCV FROM:81.95.20.18:2020 // Блок принятых данных,
обрамленных уведомлением и заголовком с указанием
типа протокола TCP и длиной 50 байт.
+IPD,50,TCP:HelloHelloHelloHelloHelloHelloHelloHell
oHello
```

Прием данных в командном режиме, ручной вывод принятых данных

Для смены способа вывода данных требуется разорвать соединение и деактивировать контекст.

```
AT+CIPCLOSE // Закрытие соединения.
CLOSE OK
AT+CIPSTATUS
OK
STATE: TCP CLOSED
AT+CIPSHUT // Деактивация контекста.
SHUT OK
AT+CIPRXGET?
+CIPRXGET: 0 // Автоматический вывод принятых
данных.
OK
AT+CIPRXGET=1 // Настройка ручного вывода данных.
OK
AT+CSTT
OK
AT+CIICR
OK
AT+CIFSR
100.69.113.182
AT+CIPSTART="TCP","81.95.20.18",2020
OK
CONNECT OK
AT
OK
AT
OK
+CIPRXGET: 1,"81.95.20.18:2020" // Уведомление
о приеме данных от сервера.
AT
OK
AT+CIPRXGET=4 // Уточнение размера принятых
данных.
+CIPRXGET: 4,100 // Пришло 100 байт данных.
OK
```

```
AT+CIPRXGET=2,20 // Вывести 20 байт в порт UART;
+CIPRXGET: 2,20,80,"81.95.20.18:2020" // В буфере
модуля осталось 80 байт.
HelloHelloHelloHello // Запрошенные 20 байт данных.
OK
AT+CIPRXGET=2,20 // Вывести 20 байт в порт UART.
+CIPRXGET: 2,20,60,"81.95.20.18:2020" // В буфере
модуля осталось 60 байт.
HelloHelloHelloHello // Запрошенные 20 байт данных.
OK
AT+CIPRXGET=2,60 // Вывести 60 байт в порт UART.
+CIPRXGET: 2,60,0,"81.95.20.18:2020" // Приемный
буфер модуля пуст.
HelloHelloHelloHelloHelloHelloHelloHelloHelloHello
// Запрошенные 60 байт данных.
OK
AT+CIPRXGET=4 // Проверка наличия данных в буфере
модуля.
+CIPRXGET: 4,0 // Буфер пуст.
OK
```

Обмен данными с сервером в прозрачном режиме

Для смены режима передачи данных требуется разорвать соединение и деактивировать контекст.

```
AT+CIPCLOSE
CLOSE OK
AT+CIPSHUT
SHUT OK
AT+HFC=2,2 // Аппаратный контроль потока должен
быть включен, чтобы избежать потери данных.
OK
AT+CIICR
OK
AT+CIFSR
100.104.155.220
AT+CIPSTART="TCP","81.95.20.18",2020
OK
CONNECT // Соединение установлено.
HelloHelloHelloHelloHelloHelloHelloHelloHello // Пере-
дача данных на сервер как есть.
HelloHelloHelloHelloHelloHelloHelloHelloHello // При-
ем данных от сервера как есть.
AT // AT-команды будут неотвеченными и будут воспри-
няты как данные для отсылки.
AT
+++ // Эта escape-последовательность переведет
модуль в режим AT-команд, но при этом контекст и соеди-
нение сохраняются активными.
```

```
OK // В этом месте можно обработать входящие СМС,
отменить входящий голосовой вызов и проч.
AT
OK
AT
OK
AT
OK
ATO // Команда ATO возвращает модуль в режим пере-
дачи данных.
CONNECT
HelloHelloHelloHelloHelloHelloHelloHelloHelloHello
HelloHelloHelloHelloHelloHelloHelloHelloHelloHello
CLOSED // Это сообщение говорит о том, что удаленная
сторона закрыла сокет. Теперь встроенный стек TCP/IP
нужно перевести в исходное состояние.
AT+CIPSTATUS
OK
STATE: TCP CLOSED
AT+CIPSHUT
SHUT OK
AT+CIPSTATUS
OK
STATE: IP INITIAL // Стек в исходном состоянии.
```

Благодаря подробному освещению возможностей встроенного стека протоколов TCP/IP новой линейки модулей серии SIM800, выгод его применения и приведению объемного исчерпывающего примера работы со стеком в различных режимах, данная статья поможет разработчику быстро освоить материал официальных руководств по применению модулей SIMCom Wireless Solutions и послужит в разработке отправной точкой. ■

Литература

1. www.simcomm2m.com/russian
2. Батуев Б. Embedded AT: начало работы с технологией интеграции пользовательского ПО в GSM/GPRS-модуль SIM800/SIM800H // Беспроводные технологии. 2014. № 3.
3. Руководство по применению встроенного стека протоколов TCP/IP GSM/GPRS-модулей серии SIM800/SIM800. Series_TCPIP_Application Note_V1.01.pdf.
4. Система AT-команд GSM/GPRS-модулей серии SIM800. SIM800_Series_AT_Command_Manual_V1.09.pdf.