**Владимир Бобрович** (КОМПЭЛ)

QUECTEL Wireless Module Expert

# НАЧИНАЕМ РАБОТУ С СОТОВЫМ МОДУЛЕМ MC60 ОТ QUECTEL

*Новый **сотовый модуль MC60** производства компании **Quectel** с приемником **GPS-GNSS** поддерживает технологию открытого ПО OpenCPU, то есть на базе него, источника питания и GPS- и GNSS-антенн можно построить полноценный **трекер для отслеживания координат** транспортного средства. Как это сделать, описано в предлагаемой статье.*

Компания **Quectel Wireless Solutions** в 2016 году выпустила новый сотовый модуль **MC60,** внешний вид которого приведен на рисунке 1, а основные особенности перечислены ниже:

- 4-диапазонный GSM-GPRS-модуль 850/900/1800/1900 МГц;
- подключение двух SIM-карт в режиме Dual SIM Single Standby;
- встроенный интерфейс Bluetooth версии 3.0;
- встроенный приемник GPS-GNSS (GPS/GLONASS/QZSS);
- поддержка технологии OpenCPU;
- размеры 18,7x16,0x2,1 мм.

Наличие в составе модуля приемника GPS-GNSS и поддержка технологии OpenCPU позволяют создавать простые устройства определения географических координат точки, в которой находится модуль MC60. То есть, можно создать, например, трекер, в котором будут только источник питания, модуль и две антенны, GSM и GNSS. Программа такого трекера пишется на языке Си и использует API OpenCPU модуля. Рассмотрим простой пример такой программы – «GPS-to-SMS». Эта программа была написана в учебных целях и может быть отнесена к примерам типа «Hello, World!». Она работает следующим образом: после подачи питания на модуль и его регистрации в сотовой сети программа включает приемник GPS-GNSS и далее ожидает поступления SMS на номер SIM-карты в модуле. После получения SMS программа сравнивает номер ее отправителя с номером, заданным в программе, и, при их совпадении, считывает с приемника GPS-GNSS строку с NMEA RMC-сообщением и отправляет ее в ответной SMS. Если номер не совпадает, то ответная SMS не отправляется, а просто продолжается цикл ожидания.

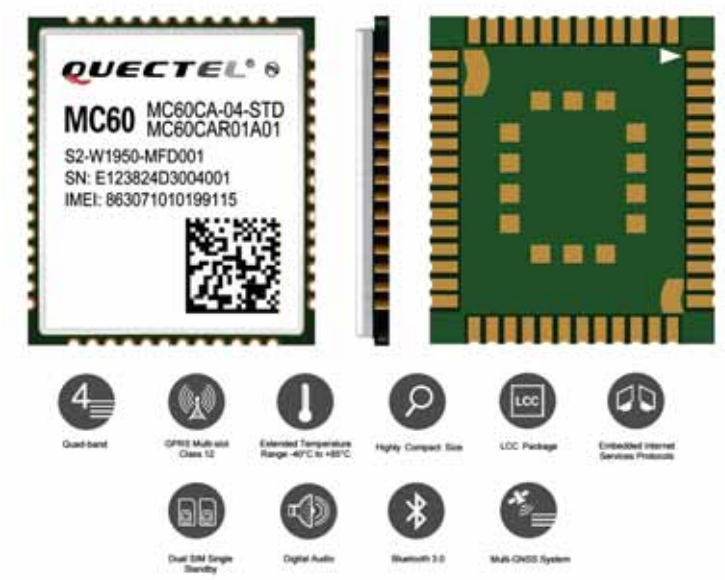


Рис. 1. **Внешний вид сотового модуля MC60 производства Quectel Wireless Solutions**

**ТЕКСТ ПРОГРАММЫ:**

```
  //07.10.2016 BVV
  //GPS to SMS primer
  //Программа ожидает SMS с заданного номера и
отсылает ответ со строкой NMEA RMC.

  #include «ril.h»
  #include «ril_util.h»
  #include «ril_sms.h»
  #include «ril_system.h»
  #include «ril_gps.h»
  #include «ql_stdlib.h»
  #include «ql_error.h»
  #include «ql_trace.h»
  #include «ql_system.h»
  #include «ql_memory.h»
  #include «ql_type.h»

  //Номер телефона, с которого должна прийти
SMS
  char strPhoneNumber[] = «+7xxxxxxxxxx»;

  //Функция, вызываемая при получении новой
SMS
  static void Hdlr_RecvNewSMS(u32 nIndex)
  {
    u32 uMsgRef = 0;
    ST_RIL_SMS_TextInfo *pTextInfo = NULL;
    ST_RIL_SMS_DeliverParam *pDeliverTextInfo =
NULL;
    char      aPhNum[RIL_SMS_PHONE_NUMBER_MAX_
LEN] = {0,};
    pTextInfo   =   Ql_MEM_Alloc(sizeof(ST_RIL_
SMS_TextInfo));
    Ql_memset(pTextInfo,  0x00,  sizeof(ST_RIL_
SMS_TextInfo));
    RIL_SMS_ReadSMS_Text(nIndex,      LIB_SMS_
CHARSET_GSM, pTextInfo);
```

```
    pDeliverTextInfo  =  &((pTextInfo->param).
deliverParam);
    Ql_strcpy(aPhNum, pDeliverTextInfo->oa);
    Ql_MEM_Free(pTextInfo);

    //Отправка SMS, если номер отправителя из
пришедшей SMS совпадает с номером в программе
    if (Ql_strstr(aPhNum, strPhoneNumber))
    {
      u8 rdBuff[160]; // буфер для GPS
      u8 sendBuff[160]; // буфер для текста от-
правляемой SMS
      char item[]= «RMC»; // из GPS будет про-
читана строка: +QGNSSRD: $GNRMC...
      Ql_memset(rdBuff, 0, sizeof(rdBuff)); //
заполнение буфера GPS нулями
      Ql_memset(sendBuff, 0, sizeof(sendBuff));
// заполнение буфера SMS нулями
      RIL_GPS_Read(item, rdBuff);  // чтение из
GPS
      // Копирование GPS NMEA RMC строки в текст
SMS, кроме символов «\r\n+QGNSSRD: $»
      Ql_strncpy(sendBuff,          rdBuff+13,
sizeof(sendBuff));
      // Отправка SMS
      Ql_Debug_Trace(«Sending  SMS  with  the
text:\r\n»);
      Ql_Debug_Trace(«%s\r\n», sendBuff);
      RIL_SMS_SendSMS_Text(aPhNum,         Ql_
strlen(aPhNum),   LIB_SMS_CHARSET_GSM,   (u8*)
sendBuff, Ql_strlen(sendBuff), &uMsgRef);
    }
    else // Входящая СМС пришла с неизвестного
номера
    {
      Ql_Debug_Trace («Received SMS came from
unknown phone number: %s\r\n», aPhNum);
    }
    return;
  }

   //Главная функция программы
  void proc_main_task(s32 iTaskID)
  {
    ST_MSG taskMsg;
    Ql_Debug_Trace(«MC60  OpenCPU  GPS-to-SMS
primer\r\n»);
    Ql_Debug_Trace(«Waiting   for   incoming
SMS...\r\n»);
    while (TRUE)
    {
      Ql_memset(&taskMsg,   0x0,   sizeof(ST_
MSG));
      Ql_OS_GetMessage(&taskMsg);
      switch (taskMsg.message)
      {
            case MSG_ID_RIL_READY:
            {
                  Ql_RIL_Initialize();
                  // Включение питания GPS
                  RIL_GPS_Open(1);
                  break;
            }
            case MSG_ID_URC_INDICATION:
                  switch (taskMsg.param1)
                  {
                        case     URC_SYS_INIT_
STATE_IND:
                        {
                              if   (SYS_STATE_
SMSOK == taskMsg.param2)
                              // Получено «SMS
Ready» - модуль готов к работе
                              {
                                    RIL_SMS_
DeleteSMS(0, RIL_SMS_DEL_ALL_MSG);
                              }
                              break;
                        }
                        case URC_NEW_SMS_IND:
                        // Получена новая SMS
                        {
                              Ql_Debug_
Trace(«New SMS received!\r\n»);
                              Hdlr_
RecvNewSMS(taskMsg.param2);
                              RIL_SMS_
DeleteSMS(0, RIL_SMS_DEL_ALL_MSG);
                              break;
                        }
                        default:
                              break;
                  }
                  break;
      default:
            break;
      }
    }
  }
```

Вначале, как обычно, подключаются необходимые заголовочные файлы. Файлы `ril.h` и `ril_xxx.h`, относятся к модулю Quectel OpenCPU RIL (*Radio Interface Layer*). Функции, реализованные в этом модуле – выполнение AT-команд, первичная обработка URC (*Unsolicited Result Code*), прием и отправка SMS, управление GPS-GNSS-приемником и другие. Особенностью модуля RIL является то, что он имеет открытый исходный код и позволяет пользователю добавлять собственные программные функции. Подключаемые файлы `ql_stdlib.h` и `ql_trace.h` необходимы для вывода отладочной информации в DEBUG UART-порт. Файл `ql_error.h` содержит все коды ошибок, возвращаемых API-функциями OpenCPU, а файл `ql_system.h` необходим для организации главного цикла программы.

В символьном массиве `strPhoneNumber[]` должен быть записан номер телефона, с которого будут отправляться входящие SMS.

Функция `Hdlr_RecvNewSMS(u32 nIndex)` вызывается при получении новой SMS, параметр nIndex передает в функцию порядковый номер принятой SMS. Содержимое принятой SMS считывается RIL-функцией `RIL_SMS_ReadSMS_Text`

в переменную с указателем pTextInfo. Текст SMS никак не анализируется, из нее только извлекается номер отправителя в переменную aPhNum. Далее функция Ql_strstr(aPhNum, strPhoneNumber) производит сравнение телефонных номеров, и если они совпадают, то будет выполнена отправка SMS с NMEA RMC-строкой. Если номера не совпадают – SMS отправлена не будет, а в отладочный порт будет выведено сообщение о недопустимом номере: Ql_Debug_Trace(«Received SMS came from unknown phone number: %s\r\n», aPhNum).

Если будет осуществлена отправка SMS, то сначала производится считывание строки из приемника GPS-GNSS, для этого вызывается функция RIL_GPS_Read(item, rdBuff). Передаваемый параметр item определяет тип читаемой NMEA-строки, а rdBuff является указателем на буфер, в который будет помещена строка. Если параметру item присвоить значение «ALL», то будет считан целый кадр NMEA-сообщения. Для MC60 это:

```
+QGNSSRD: $GNRMC,060654.000,A,5543.5146,N,03
739.1604,E,0.00,35.05,050916,,,D*40
$GNVTG,35.05,T,,M,0.00,N,0.00,K,D*15
$GNGGA,060654.000,5543.5146,N,03739.1604,E,
2,17,0.66,145.6,M,14.4,M,,*7D
$GPGSA,A,3,29,02,32,19,31,25,06,24,14,12,,,
0.95,0.66,0.67*0C
$GLGSA,A,3,77,78,70,88,87,86,71,,,,,,0.95,0
.66,0.67*1C
$GPGSV,3,1,12,12,78,128,35,25,60,288,41,02,
56,120,21,06,40,061,30*78
$GPGSV,3,2,12,29,30,242,36,49,20,217,35,24,
19,182,22,14,17,300,24*7C
$GPGSV,3,3,12,31,16,322,16,19,15,065,24,32,
10,273,31,03,03,001,16*70
$GLGSV,3,1,10,87,82,057,24,77,59,174,38,78,
55,311,21,88,43,224,45*6F
$GLGSV,3,2,10,86,23,046,18,70,17,012,21,79,
14,325,,71,10,061,17*69
$GLGSV,3,3,10,76,10,155,,69,03,321,*69
$GNGLL,5543.5146,N,03739.1604,E,060654.000,
A,D*4F
```

Чтобы не усложнять программу, в ней не производится никакой обработки NMEA-сообщений, а просто читается одна строка, и ее содержимое передается в отправляемую SMS. Для этого параметру item присваивается значение «RMC».

Вот пример RMC-строки, которая читается из приемника, если сигнал со спутников еще не захвачен:

```
+QGNSSRD: $GNRMC,000100.710,V,,,,,0.00,0.00,
010104,,,N*50
```

Если захвачен сигнал только с одного спутника или с нескольких, но координаты еще не определены:

```
+QGNSSRD: $GNRMC,053816.304,V,,,,,0.40,309.5
5,050916,,,N*58
```

А такая строка выдается, если сигнал захвачен и координаты определены:

```
+QGNSSRD: $GNRMC,065536.000,A,5543.5135,N,03
739.1593,E,0.00,357.42,280916,,,A*75
```

RMC означает минимальный рекомендованный набор данных GPS, он содержит UTC-время определения координат (065536.000), признак их валидности (A – валидны, V – невалидны), сами координаты (5543.5135,N,03739.1593,E), скорость (0.00) и направление движения (357.42), а также дату (280916). Параметр, который стоит непосредственно перед символом «*», показывает режим определения координат:

- «A» – автономный режим;
- «D» – дифференциальный режим;
- «N» – недостоверные данные.

Символ «*» означает конец данных, а «75» – это контрольная сумма. В начале и конце строки передаются еще символы «\r\n», то есть перевод строки и новая строка.

После того как строка RMC прочитана, она копируется в буфер текста отправляемой SMS: Ql_strncpy(sendBuff,rdBuff+13,sizeof(sendBuff)). Для уменьшения размера SMS начальные символы «\r\n+QGNSSRD: $» в буфер отправки не копируются, для этого к начальному значению адреса буфера RMC прибавляется число 13. Отправка SMS производится вызовом функции RIL_SMS_SendSMS_Text. В DEBUG-порт выводится сообщение об отправке и текст отправляемой SMS. Для этого два раза вызывается функция Ql_Debug_Trace.

### ГЛАВНАЯ ФУНКЦИЯ ПРОГРАММЫ, PROC_MAIN_TASK(S32 ITASKID)

В начале главной функции в DEBUG-порт выводятся строки, свидетельствующие о старте программы, и создается структура taskMsg, в которую будут передаваться системные сообщения по мере их возникновения. Далее организуется цикл выборки сообщений и их обработка. Здесь можно отметить, что Quectel OpenCPU является многозадачной системой реального времени, в которой может исполняться одна главная задача и до 10 субзадач. Для каждой задачи в программе должна быть создана своя структура типа ST_MSG и организован свой цикл выборки сообщений. Программа «GPS-to-SMS» – однозадачная, и в ней только один такой цикл. В самом начале цикла производится обнуление всех полей стандартной структуры ST_MSG, а затем вызывается функция, осуществляющая выборку (считывание) сообщения из очереди сообщений: Ql_OS_GetMessage(&taskMsg).

Первым в цикле проверяется сообщение о готовности модуля RIL, это MSG_ID_RIL_READY. Если такое сообщение появилось, то вызывается функция инициализации модуля – Ql_RIL_Initialize(). Только после инициализации RIL программа может использовать RIL API-функции, и в данном случае программа вызывает функцию RIL_GPS_Open(1), которая включает питание GPS-GNSS-приемника.

Далее в цикле проверяется системное сообщение MSG_ID_URC_INDICATION, оно появляется при получении URC. Если имеется такое системное сообщение, далее тестируется поле param1 структуры taskMsg, которое определяет тип полученного URC. В модуле MC60 может быть множество URC-сообщений. Те из них, которые обрабатываются модулем RIL, пе-

речислены в файле `ril.h`. В поле param2 структуры taskMsg содержатся дополнительные параметры для пришедшего URC. В цикле нашей программы проверяется приход только двух типов URC, это `URC_SYS_INIT_STATE_IND` и `URC_NEW_SMS_IND`.

URC-сообщение `URC_SYS_INIT_STATE_IND` генерируется в момент инициализации MC60 при его включении или перезагрузке. Если при этом параметр param2 структуры `taskMsg` равен `SYS_STATE_SMSOK`, это означает, что модуль MC60 зарегистрировался в сети и готов к приему и отправке SMS. Для исключения ситуации с переполнением памяти SMS в программе вызывается функция удаления всех SMS, принятых ранее: `RIL_SMS_DeleteSMS`.

URC-сообщение `URC_NEW_SMS_IND` генерируется при получении новой SMS. Параметр param2 структуры `taskMsg` при этом содержит номер (индекс) принятой SMS, по которому производится ее чтение. Программа вызывает функцию-обработчик `Hdlr_RecvNewSMS(taskMsg.param2)`, которая выполняет все необходимые действия – читает пришедшую SMS, читает RMC-строку из приемника GPS-GNSS и отправляет ответ. После возврата из функции-обработчика опять же вызывается функция удаления SMS.

## ПРИМЕЧАНИЯ

Впрограмме никак не обрабатываются коды ошибок, которые возвращают почти все API-функции OpenCPU. Это сделано для упрощения программы и уменьшения ее текста. В программе не используются режимы энергосбережения приемника GPS-GNSS и трансивера GSM, и средний ток потребления достигает 95 мА при проверке с использованием фирменного отладочного комплекта Quectel GSMEVB-KIT и активной антенны GPS-GLONASS из Quectel MC60-TE-A Kit.