

Дмитрий Поваляев (ОАО «НИИСА»)

ДОРОГА К «ОАЗИСУ», ИЛИ ПРАКТИЧЕСКИЕ ВОПРОСЫ ПЕРЕХОДА НА ОПЕРАЦИОННУЮ СИСТЕМУ OASIS R7



В статье подробно описываются проблемы беспроводного процессора Q268 производства Sierra Wireless (линейка Wavcom), возникающие при работе с приложением Open AT, а также методы и алгоритмы их устранения. Также рассказывается об обновлении Oasis R7.4, в котором устранена проблема «зависания» (freeze) при циклическом чтении из шины I²C.

В 2008 г. в инициативном порядке в ОАО «Научно-исследовательский институт систем автоматизации» (ОАО «НИИСА») начата разработка навигационно-связных комплексов, функционирующих в составе диспетчерской системы. Носимое устройство должно обеспечивать пользователя голосовой телефонией стандарта GSM, определять собственное местоположение по сигналам GPS и передавать навигационную информацию по доступным каналам связи (GSM/GPRS, УКВ) на диспетчерский пункт. Дополнительно предусмотрен канал Bluetooth для отображения данных на КПК или ноутбуке.

Мобильный комплект предназначен для установки на транспортное средство (автомобиль, корабль или катер) в исполнении для внешнего монтажа. Мобильный комплект поддерживает все функции, присущие носимому варианту (кроме голоса), но имеет двойное питание (как от бортовой сети транспортного средства, так и от встроенной АКБ). Устройство также содержит дополнительные интерфейсы для логических датчиков, каналов управления, АЦП и шины CAN.

Сердцем носимого и мобильного устройства является беспроводной процессор Sierra Wireless (ранее Wavcom) Q2687 (рис. 1). В конструкции оказались задействованы практически все интерфейсы процессора, кроме параллельной шины и DAC. В частности, UART_1 используется как внешний порт для программирования и подключения внешних устройств, UART_2 — для взаимодействия с Bluetooth модулем, SPI_1 — передатчик УКВ, SPI_2 — интерфейс CAN и, наконец, I²C обеспечивает обмен информацией с GPS-приемником.

Изначально приложение Open AT было создано под Open AT Firmware 6.63e, OS v4.24, SDK4.27. В процессе работы выяснилось, что при циклическом опросе шины I²C с периодом 0,5 с процессор как бы останавливался, замораживался (freeze) — т.е. прекращал выполнение приложения Open AT. При этом не выполнялся ни один таймер или функция, а процессор не реагировал на AT-команды, за исключением AT+WOPEN=0 (остановка встроенного приложения). В таком состоянии процессор находился до тех пор, пока его не перезагружали либо с помощью аппаратного сброса (hard reset), либо по срабатыванию внутреннего программного таймера перезапуска (software watchdog), что происходило через несколько минут после начала состояния «freeze». После перезагрузки процессор выдавал сообщение об ошибке вида



«Except RTK ...144 а 2» (рис. 2). Описание этой ошибки в документации на OAT OSv4.24 я не нашел, однако встретил в документации на OAT OSv6.20: «144: Raised if too many items are pushed in the queue». Это можно найти в разделе «ADL Queue Service», но этот сервис в данной программе не используется.

Мои попытки выяснить причину закончились на следующей мысли: возможно, всему виной искажения сигнала по шине, возникающие из-за несогласованности подключенных подтягивающих (pull-up) резисторов и результирующей емкости проводников шины I²C. Вследствие этого факта можно было наблюдать разную картину частоты «зависаний» на разных изделиях с различной топологией проводников шины I²C и, следовательно, с различной емкостью проводников шины. Картина также менялась с изменением номиналов подтягивающих резисторов, подключаемых к



Рис. 1. Беспроводной процессор Q2687

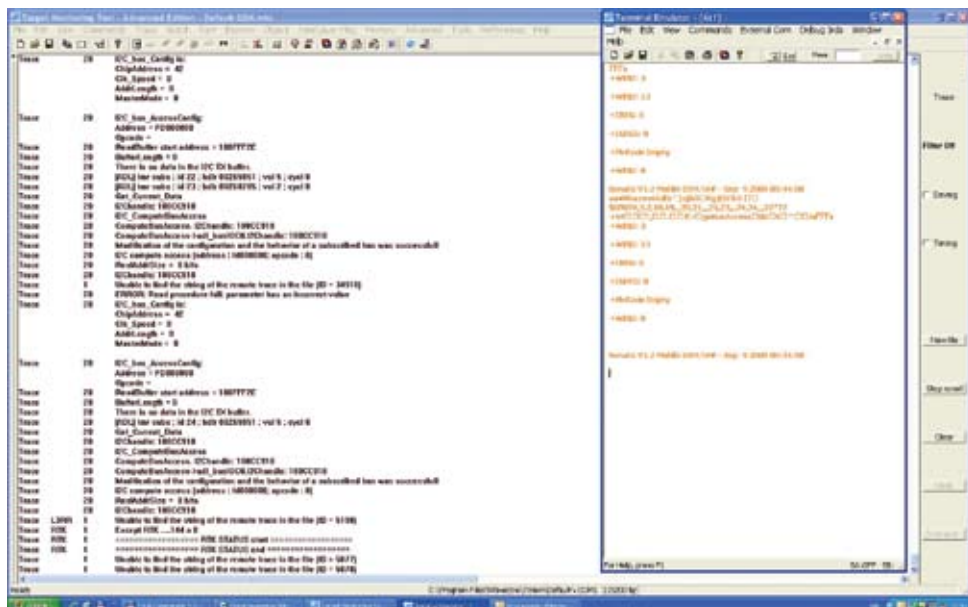


Рис. 2. Отладка программы

шине одного и того же изделия. Когда мне удалось подобрать такую комбинацию подтягивающих резисторов для каждого изделия, что чтение по шине производилось без ошибок в течение длительного времени, мы столкнулись с еще одной проблемой. Похожие симптомы «зависания» процессора наблюдались в моменты передачи голоса и данных по GSM/GPRS. Виной всему, по-видимому, были высокочастотные наводки на проводники шины. Так как локализовать причину мне не удалось, то, почитав сообщения на форуме Sierra Wireless/Wavescom на данную тему, а так же следуя рекомендациям специалистам Sierra Wireless, я все-таки решил перевести наше приложение Open AT на более новую операционную систему Open AT Firmware **R7.3**, OS v6.20, SDK2.20. Процесс включал в себя следующие шаги:

1. Установил новую оболочку Open AT IDE «Oasis 2.20», скачав ее с ftp-сервера КОМПЭЛ (доступ открывается по запросу; кстати – очень удобно, что много необходимой информации для разработчиков доступно в любое время);
2. Перепрошил процессор Wavescom под новую операционную систему, следуя описаниям на сайте производителя;
3. Собрал на основе старых исходников проект с помощью Project Wizard (Open AT может отказаться собирать проект, если пользователь Windows в своем имени использует кириллические символы);
4. Скомпилировал проект с опцией «Wismo Target» в новой среде «Oasis 2.20», и, естественно, получил в ответ много сообщений об ошибках и несобранный бинарный файл;
5. Далее открыл документацию и последовательно начал выяснять различия в описании функций.

Первое с чем я столкнулся – в новой операционной системе некоторые статические данные хранятся в регистре. Ранее для определения версии hardware использовалась функция «adl_ioGetProductType()», теперь же данные приходилось получать из регистра.

Второе – в новом OAT ADL кардинально поменялся интерфейс работы с портами GPIO, поэтому эту часть программы пришлось переделать:

- Определение самих GPIO: вместо «ADL_IO_Q2687_GPIO_21» определил порты как «ADL_IO_GPIO | 21»;
- Структуры, описывающие GPIO: структуры типа:


```
const adl_ioConfig_t Config_EN_WT [GPIO_COUNT_EN_WT] =
{
    {EN_WT_Power_GPIO, 0, ADL_IO_OUTPUT, ADL_IO_LOW}, // EN_3V3_WT - On/Off Bluetooth module WT-12
    {EN_WT_Translator_GPIO, 0, ADL_IO_OUTPUT, ADL_IO_LOW}, // EN_IO_WT - Enable for Signal Translator FXL4TD245
    {RESET_WT_GPIO, 0, ADL_IO_OUTPUT, ADL_IO_LOW} //RST_WT - Reset WT-12
};

```

 стали выглядеть так:


```
const adl_ioDefs_t* Config_EN_WT [GPIO_COUNT_EN_WT] =
{
    EN_WT_Power_GPIO | ADL_IO_DIR_OUT | ADL_IO_LEV_LOW, //EN_3V3_WT - On/Off Bluetooth module WT-12
    EN_WT_Translator_GPIO | ADL_IO_DIR_OUT | ADL_IO_LEV_LOW, //EN_IO_WT - Enable for Signal Translator FXL4TD245

```

```
RESET_WT_GPIO, 0 | ADL_IO_DIR_OUT | ADL_IO_LEV_LOW // RST_WT - Reset WT-12
};
```

- На форуме я узнал, что GPIO44 в новой операционной системе называется GPIO 0, хотя в документации в явном виде я этого факта не нашел;
- Подписка на сервисы GPIO: соответственно, изменился интерфейс функции «adl_ioSubscribe», было:

```
// Subscribe to the RUN_REQ event service
RUN_REQ_GpioEventHandle = adl_ioEventSubscribe (RUN_REQ_GpioEventHandler);
GpioHandle_RUN_REQ_Signal = adl_ioSubscribe ( GPIO_COUNT_RUN_REQ, (adl_ioConfig_t *)&Config_RUN_REQ, ADL_TMR_TYPE_100MS, CheckRUN_REQ_state_Time, RUN_REQ_GpioEventHandle );
```

стало:

```
// Subscribe to the RUN_REQ event service
RUN_REQ_GpioEventHandle = adl_ioEventSubscribe ( RUN_REQ_GpioEventHandler );
GpioHandle_RUN_REQ_Signal = adl_ioSubscribe ( GPIO_COUNT_RUN_REQ, Config_RUN_REQ, ADL_TMR_TYPE_100MS, CheckRUN_REQ_state_Time, RUN_REQ_GpioEventHandle );
```

- Так как информация о GPIO хранится в виде битовых полей, то в новой версии OAT ADL получение информации о GPIO производится с помощью масок, например: (((adl_ioDefs_t *)Param)[RING_Button] & ADL_IO_LEV_MSK) & ADL_IO_LEV_HIGH;

- Соответствующим образом изменился и интерфейс функций «adl_ioWriteSingle» и «adl_ioReadSingle».

Третье – изменения коснулись модуля работы с шиной I²C. Основное время ушло на то, чтобы понять, что теперь в адресе slave-устройства (adl_busI2CSettings_t) используется реальный адрес периферийного модуля (в нашем случае – GPS-приемника LEA-5H), в то время как в OAT OS v4.24 необходимо было указывать адрес со сдвигом влево на 1 бит. Запутанности также способствовало то, что при неверно указанном адресе slave-устройства функция «adl_busRead» («adl_busWrite») возвращает ошибку типа «ADL_RET_ERR_PARAM». Получив такую ошибку, невольно начинаешь искать проблему в структурах, описывающих шину, и в других параметрах, а об адресе устройства приходит мысль в последнюю очередь.

• Изменились структуры, описывающие работу шины:

в OAT OS v4.24 использовалась структура вида:

```

/*****/
/* I2C bus settings */
/*****/
const adl_busI2CSettings_t
I2C_bus_Config =
{
    0x84, // ChipAddress, 0x42<<1
    ADL_BUS_I2C_CLK_STD //
    Clk_Speed
};

```

в OAT OS v6.20 взято из примера драйвера для модуля памяти по шине I²C.

```

/*****/
/* I2C bus settings */
/*****/
const adl_busI2CSettings_t
I2C_bus_Config =
{
    0x42, // ChipAddress;
    ADL_BUS_I2C_CLK_STD/*, //
    Clk_Speed
    ADL_BUS_I2C_ADDR_7_BITS,
    //Remote chip address length
    configuration
    ADL_BUS_I2C_MASTER_MODE*/
    //Master or slave running mode
};

```

Хотя в документации структура `adl_busI2CSettings_t` описана с четырьмя параметрами, в настоящей версии (как и в примере) все работает и с двумя. Вероятно, параметры `ADL_BUS_I2C_ADDR_7_BITS` и `ADL_BUS_I2C_MASTER_MODE` в операционной системе как-то устанавливаются по умолчанию.

• Также изменилась структура `adl_busAccess_t`: для OAT OS v4.24 `adl_busAccess_t` определялась как:

```

static adl_busAccess_t I2C_bus_Access = { 0, 0, 0, 0, ADL_BUS_SIZE_BYTE };

```

для OAT OS v6.20 динамически назначается только два параметра — остальные значения хранятся в реестре:

```

static adl_busAccess_t I2C_bus_Access = { 0, 0 };

```

• Изменился интерфейс функции `adl_busSubscribe`:

для OAT OS v4.24 было:

```

I2CHandle = adl_busSubscribe(
ADL_BUS_I2C, (adl_busSettings_u * )&I2C_bus_Config);

```

для OAT OS v6.20:

```

I2CHandle = adl_busSubscribe(
ADL_BUS_I2C, 1, &I2C_bus_Config);

```

Четвертое — изменения в работе с шиной SPI аналогичны изменениям для шины I²C. Так же изменилась структура `adl_busSPISettings_t`, но по

окончании отладки работы по шине I²C перевод работы SPI на новую ОС не составил труда.

Пятое — изменения интерфейса функции обработки таймера (*timer hander*). В таймерах было достаточно изменить только интерфейс хандлеров. Как я понял, в OAT OS v6.20 в таймер можно передать произвольный набор данных через параметр `void * Context`, но в нашем приложении этого не было предусмотрено по той причине, что в OAT OS v4.24 такой возможности просто не было. Так как в остальном интерфейс работы с таймерами не изменился, то после замены описания хандлера с:

```

void tmrHandler_initLEA5H(u8
ID)
{
.....
}
на:
void tmrHandler_initLEA5H(u8
ID, void * Context)
{
.....
}

```

все замечательно заработало.

Примечательно, что пока в проекте использовались хандлеры со старым интерфейсом, проект компилировался без ошибок, но после загрузки бинарного файла в процессор Q2687 постоянно перезагружался до тех пор, пока сама операционная система не останавливала сбойное приложение Open AT.

Шестое — изменения в управлении зарядом батареи Wavecom Battery Charge Management. Для зарядки аккумулятора батареи (АКБ) в устройстве носимого исполнения использовался встроенный механизм заряда LiIon аккумуляторных батарей. В OAT OS v4.24 для инициализации алгоритма использовалась команда:

```

adl_atCmdCreate(
«AT+WBCM=3,2», FALSE, (adl_atRspHandler_t) NULL, NULL); //
Set the unsolicited indication «ON»

```

в OAT OS v6.20 режим «2» для команды `AT+WBCM=3` отсутствует, следовательно пришлось заменить на:

```

adl_atCmdCreate(
«AT+WBCM=3,1», FALSE, (adl_atRspHandler_t) NULL, NULL); //
Set the unsolicited indication «ON»

```

Седьмое — индикатор WIND: 3. В нашем приложении Open AT была предусмотрена возможность отключать некоторые аппаратные ресурсы процессора, например — клавиатуру, для того, чтобы использовать эти выводы (*keyboard*) в качестве GPIO. Эти процедуры производились только после получения операционной системой процессора сообщения о готовности обрабатывать AT-команды, т.е. `+WIND: 3` («the product is

ready to process AT commands»). Однако в OAT OS v6.20 индикатор `+WIND: 3` выдавался операционной системой еще до того момента, как наше приложение OpenAT стартует. Поэтому при старте приложения пришлось отключать клавиатуру и управление светодиодом индикации (*FlashLED*) непосредственно в функции `adl_main`.

Восьмое — команды аудио-тракта (*Audio Commands*). В новой операционной системе появилась возможность задавать уровни громкости сигнала динамика либо в относительных единицах, либо в децибелах `AT+WBNW=8(,1)`, соответственно пришлось незначительно переделывать интерфейс работы с аудио-интерфейсом в носимом варианте устройства.

Напоследок отмечу, что в проекте используются следующие сервисы Open AT:

- AT Commands Service;
- Timers;
- Memory Service;
- Debug Traces;
- Flash;
- FCM Service;
- GPIO Service;
- Bus Service;
- SMS Service;
- Call Service;
- WIP GPRS Service;
- AT/FCM IO Ports Service;
- ADL Audio Service.

а также:


- Wavecom Battery Charge Management;
- Audio Commands.

Проект отлаживался в среде Microsoft Visual Studio 2003.

Процессор Q2687H:
Hardware Version 4.20.
R73_00gg.Q2687H 2087432 121208
15:34

+WOPEN: 2, «AT v06.20», «AT v06.20»

Заключение

Уже разобравшись с большинством проблем возникших при переходе от OAT OS v4.24 к OAT OS v6.20, на сайте Sierra Wireless я обнаружил, что вышло обновление Oasis R7.4. В описании к нему было сообщено, что в версии R7.4 устранена проблема «зависания» (*freeze*) при циклическом чтении из шины I²C. Проблем при переходе от R7.3 к R7.4 не возникло. Уже более месяца наше программно обновленное мобильное устройство в режиме тестовой эксплуатации без проблем работает с GPS-приемником по шине I²C. 

Получение технической информации,
заказ образцов, поставка —
e-mail: wireless.vesti@compel.ru