Портирование стека Keil RL-TCPNet на микроконтроллер MDR32F9Q2I

Андрей Шаронов, Равиль Бикметов (г. Пермь)

Основной задачей портирования стека является разработка драйвера физического уровня, который должен обеспечивать работу канала связи UART/RS-232 и протокола PPP.

Введение

Микроконтроллер MDR32F9Q2I можно по праву назвать единственным массовым отечественным микроконтроллером, созданным за последнее десятилетие. «Миландр» – компания-производитель постаралась на славу: стандартное микропроцессорное ядро Cortex-M3, разнообразная периферия, сравнительно низкая цена, как микроконтроллера, так и отладочного комплекта, и хорошая поддержка на интернет-странице производителя дают микроконтроллеру шанс закрепиться не только на рынке компонентов для ВПК, но и на гражданском рынке. Важен и тот факт, что поддержка данного кристалла включена в популярную среду Keil uVision.

Краткий обзор существующих стеков TCP/IP для микроконтроллеров

В современном мире любой разрабатываемый прибор должен уметь работать в глобальной или в локальной сети. Для реализации подобных возможностей разрабатываются специальные программные библиотеки – стеки TCP/IP. На данный момент, как на микроконтроллер MDR32F9Q2I, так и на другие микроконтроллеры серии 1986ВЕх компании «Миландр», портированы стеки lwIP и uIP. К сожалению, стек RL-TCPNet компании Keil для данного семейства официально не был адаптирован. Поскольку авторы имеют достаточно много наработок по данному стеку, было принято решение о его переносе на отладочную плату микроконтроллера MDR32F9Q2I.

В сравнительную таблицу 1 включены три вышеперечисленных стека. Видно, что стек RL-TCPNet хотя и обладает рядом недостатков, среди которых тесная связь с единственной средой Keil uVision и закрытый исходный код, но имеет неплохие возможности и уверенно конкурирует с другими стеками для встраиваемых приложений. Поэтому наличие порта на микроконтроллере MDR32F9Q2I будет более чем уместно.

Описание протокола PPP

Ввиду отсутствия поддержки протокола Ethernet на уровне микроконтроллера, в качестве протокола канального уровня будет использован PPP. На физическом уровне передачи данных используется последовательный интерфейс RS-232. Этот же интерфейс используется для подключения к компьютеру отладочной платы с микроконтроллером MDR32F9Q2I.

Сам протокол позволяет организовать соединение «точка-точка», поэтому в сети могут присутствовать только один клиент и один сервер. В данном случае, сервером будет выступать отладочная плата, а клиентом – персональный компьютер. Протокол позволяет реализовать контроль целостности кадра, а также два режима авторизации пользователя. Все эти возможности протокола успешно реализованы в стеке TCPNet. Включить поддержку данного протокола можно в настройках конфигурационного файла *Net_Config.c* с помощью мастера настройки среды Keil uVision. Меню мастера показано на рисунке 1.

Таблица 1. Сравнительная характеристика встраиваемых стеков TCP/IP

	uIP	lwIP	TCPNet	OpenTCP	uC/TCP-IP
Поддерживаемые платформы	AVR, MSP430, ARM и др. (адаптирован для низкопроизводительных микроконтроллеров)	ARM, MIPS, Cold-Fire и др.	ARM	AVR, MSP430, ColdFire и др. (адаптирован для низкопроизводительных микроконтроллеров)	AVR, MSP430, ARM и др.
Исходный код	Открытый	Открытый	Закрытый	Открытый	Открытый
Лицензия	Freeware	Freeware	Коммерческий	Freeware	Коммерческий
Поддержка микроконтроллеров линейки 1986ВЕх	+	+	–	–	–
Протоколы канального уровня	Ethernet, PPP, SLIP	Ethernet, PPP, SLIP	Ethernet, PPP, SLIP	Ethernet	Ethernet, PPP, SLIP
Прикладные протоколы	HTTP, FTP, SNMP	HTTP, FTP, SNMP	HTTP, FTP, SNMP, SMTP, TFTP, Telnet и др.	HTTP, FTP, SNMP, SMTP, TFTP, Telnet и др.	HTTP, FTP, SNMP, SMTP, TFTP, Telnet и др.
Интерфейс BSD-сокетов	–	+	+	+	+
Тесная интеграция с Keil uVision	–	–	+	–	–

Таблица 2. Функции драйвера последовательного интерфейса

Функция	Описание
init_serial()	Инициализация контроллера последовательного интерфейса
com_getchar()	Чтение символа из буфера приема
com_putchar()	Запись символа в буфер передачи
com_tx_active()	Определяет, активен ли передатчик в данный момент

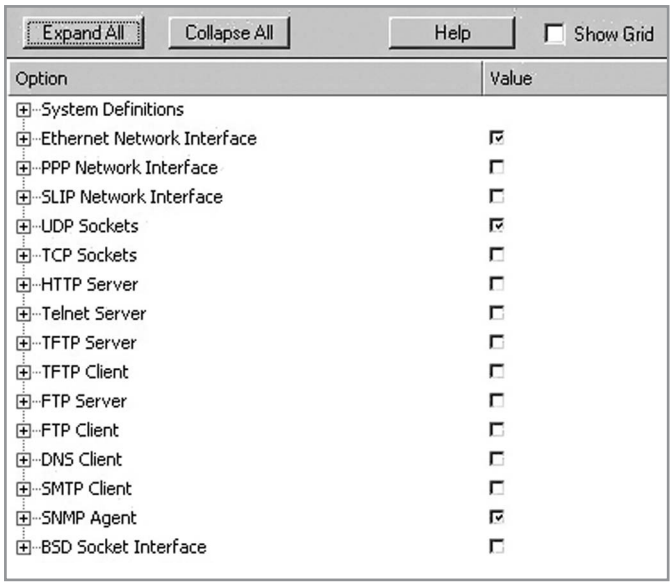

Рис.1. Меню мастера настройки файла *Net_Config.c*

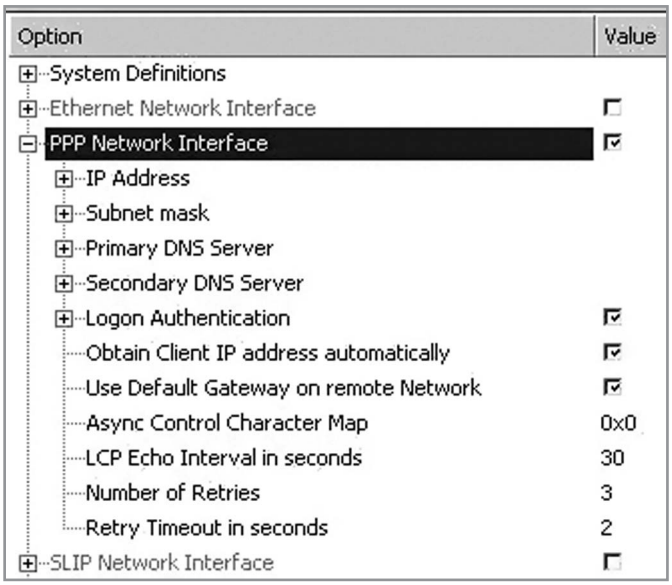

Рис.2. Настройки интерфейса PPP

Для начала при необходимости поставим галочку вместо Ethernet Network Interface напротив PPP Network Interface. Далее, открыв меню настроек интерфейса PPP, проверим все остальные параметры (см. рис. 2). Здесь можно настроить IP-адрес устройства, маску подсети, указать адреса серверов DNS, а также различные параметры авторизации клиента (логин, пароль, режим авторизации). Решить, выдавать ли клиенту автоматически IP-адрес или, если сама встраиваемая система выступает в роли клиента, получать его автоматически, установить режим управления передачей символов, число попыток соединения и другие параметры.

Для успешной работы протокола PPP необходим драйвер физического уровня. Роль такого драйвера должен исполнять файл с названием *Serial_имя семейства микроконтроллеров.c*. Такие файлы находятся в каталоге C:\каталог установки Keil\ARM\RL\TCPnet\Drivers. Как уже было сказано, аналогичный драйвер для микроконтроллера MDR32F9Q2I отсутствует. Фактически основной задачей портирования стека является разработка такого драйвера физиче-

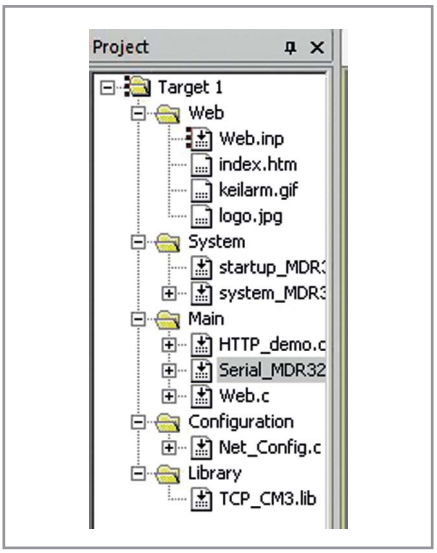

Рис.3. Структура проекта

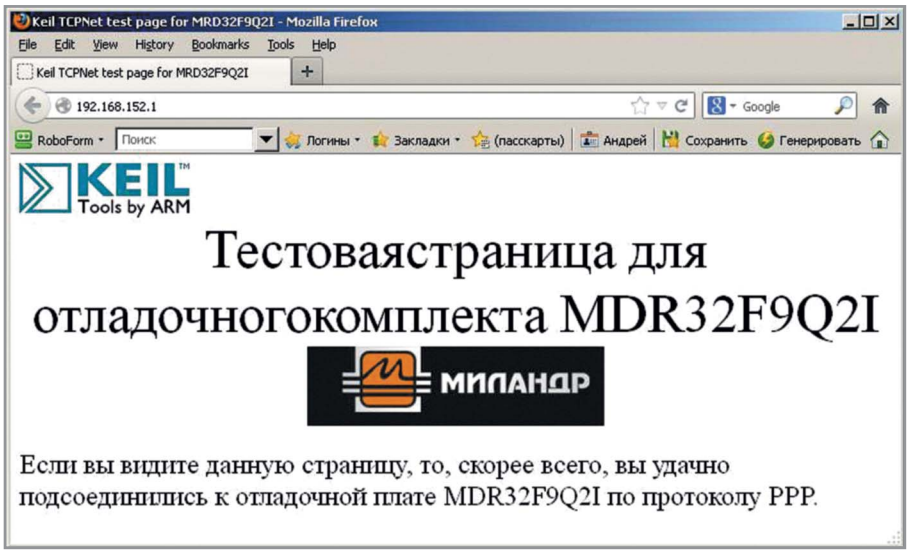

Рис.5. Тестовая страница HTTP-сервера

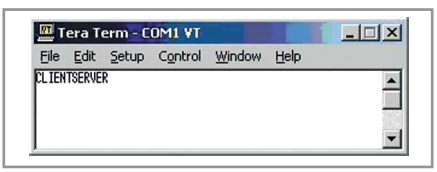

Рис.4. Окно терминальной программы при первом запуске программы

ского уровня, который должен обеспечивать следующие функции:

- инициализацию контроллера UART;
- отправку байта контроллеру UART;
- чтение байта, принятого контроллером UART;
- индикацию активности контроллера UART.

Названия функций и их краткие описания представлены в таблице 2. Более подробное описание приведено в [1].

Помимо функций, драйвер должен обрабатывать прерывания от контроллера UART, в котором будет осуществляться помещение принятых символов в буфер приема и отправка символов из буфера передачи в канал UART/RS-232.

Портирование стека TCPNet

Процедуру портирования можно разделить на три этапа:

1. настройка тактового генератора на частоту 48 МГц;
2. разработка функции генерации системных тиков с помощью системного таймера микроконтроллера;
3. разработка драйвера физического уровня.

Для начала рассмотрим функцию настройки тактового генератора. В отличие от микроконтроллеров семейства LPC23xx/24xx и LPC17xx, микроконтроллер MDR32F9Q2I не имеет визуальных средств настройки умножителя тактовой частоты. Поэтому инициализация умножителя проводится пользователем в тексте программы. Для этого требуется настроить тактовый генератор на использование внешнего кварцевого резонатора, инициализировать умножитель частоты и настроить генератор тактовых импульсов для периферийных устройств. Все эти операции проводятся над регистрами блока MDR_RST_CLK, отвечающего за тактирование микроконтроллера. При инициализации нам понадобится четыре регистра:

- HS_CONTROLL – определяет источник тактовых импульсов микроконтроллера. Здесь необходимо настроить тактирование от внешнего кварцевого резонатора;
- PLL_CONTROL – производит настройку умножителя частоты, отдельно для процессорного ядра и для контроллера USB;
- CPU_CLOCK – выбор источника тактовых импульсов для процессорного ядра микроконтроллера и периферийных устройств, не оснащенных собственным умножителем частоты;
- CLOCK_STATUS – регистр состояния, где устанавливаются флаги готов-

Листинг 1

```
void frq_init(void)
{
MDR_RST_CLK->HS_CONTROL = 0x1; // Enable HSE oscillator
while (MDR_RST_CLK->CLOCK_STATUS == 0x00) __NOP();
// wait while HSE startup
MDR_RST_CLK->PLL_CONTROL |=(6-1)*0x100;        // PLL=6 - CPU CLK 48 MHz
MDR_RST_CLK->PLL_CONTROL |=0x04;      // Enable PLL for CPU
while (!(MDR_RST_CLK->CLOCK_STATUS & 0x02))
__NOP();
MDR_RST_CLK->CPU_CLOCK = 0x106; // switch to HSE (8 MHz)
SystemCoreClockUpdate(); // Get Core Clock Frequency
}
```

Листинг 2

```
static void timer_poll ()
{
  /* System tick timer running in poll mode */

if (SysTick->CTRL & 0x10000)
{
/* Timer tick every 100 ms */
               timer_tick ();
               tick = __TRUE;
       }
}
```

ности различных узлов тактового генератора.

Каждый из перечисленных регистров подробно описан в спецификации на микроконтроллер на интернет-странице производителя [2]. Для инициализации тактового генератора на частоте 48 МГц была разработана следующая функция (см. листинг 1).

В качестве прототипа программы, демонстрирующей возможности TCPNet на микроконтроллере MDR32F9Q2I, был выбран проект для платы MCBSTM32, расположенный в каталоге C:\ каталог установки Keil \ARM\Boards\ Keil\MCBSTM32\RL\TCP-net\Http_demo. Функция формирования системного тика (timer_poll()) находится в файле *HTTP_demo.c*. Внимательное сравнение системных таймеров MCD32F9Q2I и STM32F103RB не выявило никаких различий, поэтому функция timer_poll() была оставлена без изменений (см. листинг 2).

В качестве прототипа драйвера физического уровня также был взят драйвер *Serial_STM32.c* указанной платы. Были изменены функции init_serial(), com_putchar и функция обработчика прерывания. Полный текст нового драйвера, названного *Serial_MDR32F9x.c*, приведён в Приложении 1, которое можно загрузить с интернет-страницы журнала.

В качестве прикладного приложения, как и в примере для MCBSTM32, используется HTTP-сервер. Для него была написана специальная тестовая страница. Общий состав файлов проекта приведен на рисунке 3.

После успешной компиляции программа была загружена в отладочную плату с помощью отладчика MT-Link, являющегося клоном отладчика Segger J-Link v7. Более подробно процесс настройки проекта для работы с данным отладчиком описан в [3].

Испытания

Разработанный проект встраиваемого HTTP-сервера был протестирован с помощью компьютера под управлением Windows XP. При первом старте проекта с помощью терминальной программы плате была отправлена команда CLIENT, на которую плата ответила CLIENTSERVER (см. рис. 4).

Такой обмен строками характерен при установке прямого соединения через последовательный порт компьютеров под управлением операционных систем Windows. После этого было настроено прямое подключение к плате через последовательный порт и произведено подключение по протоколу PPP. С помощью браузера была открыта страница HTTP-сервера (см. рис. 5).

Как видно из вышеприведённых рисунков, тестовая программа справляется со своими задачами. К сожалению, выявились и недостатки. Для повторного подключения к плате после разрыва связи необходимо перезапустить программу, однако данная проблема характерна и для других примеров в составе программной среды Keil uVision, и может быть решена пользователем библиотеки при разработке собственных проектов.

Литература

1. RL-ARM User's Guide. Serial Routunes. www.keil.com/support/man/docs/rlarm/rlarm_tn_ser_funcs.htm
2. Спецификация микроконтроллеров серии 1986BE9x и К1986BE9x. http://milandr.ru/uploads/Products/product_80/spec_seriya_1986BE9x.pdf
3. *Голубцов М.* Микроконтроллер MDR32F9Q2I. Часть 1. Первое знакомство с микроконтроллером и средствами разработки для него. Современная электроника, №3, 2012.