Особенности режимов энергосбережения в контроллерах W5200 и W7200 компании WIZnet

Сергей ШЕРСТНЁВ
shsa@efo.ru

Уже довольно известный на российском рынке производитель аппаратных Ethernet-решений — WIZnet — выпустил в начале 2011 года сетевой контроллер W5200. А в начале этого года компания запустила в производство микросхему W7200, сочетающую в одном корпусе управляющий контроллер серии Cortex-M3 и W5200. Они предназначены, прежде всего, для использования во встраиваемых приложениях с повышенными требованиями к энергопотреблению. Реализованные в контроллерах режимы Power Down и Wake-On-LAN дают возможность более гибкой настройки разрабатываемого устройства при реализации энергосбережения. О них и пойдет речь в данной статье.

Введение

Микросхема W5200 (рис. 1а) — это однокристальный Ethernet-контроллер, аппаратно реализующий стек протоколов TCP/IP для сетей 10BaseT/100BaseTX и имеющий возможность работы в качестве MAC-RAW канала для программного стека протоколов (Web, E-mail, FTP, Telnet) на внешнем хосте. Одновременно W5200 может поддерживать до восьми соединений, имеет 32 кбайт внутренней памяти для обмена данными (внутренняя TX/RX-память), автоматическое определение вида передачи данных (полнодуплексное или полудуплексное) и функцию MDI/MDIX опознания типа кабеля (прямой или кроссовер).

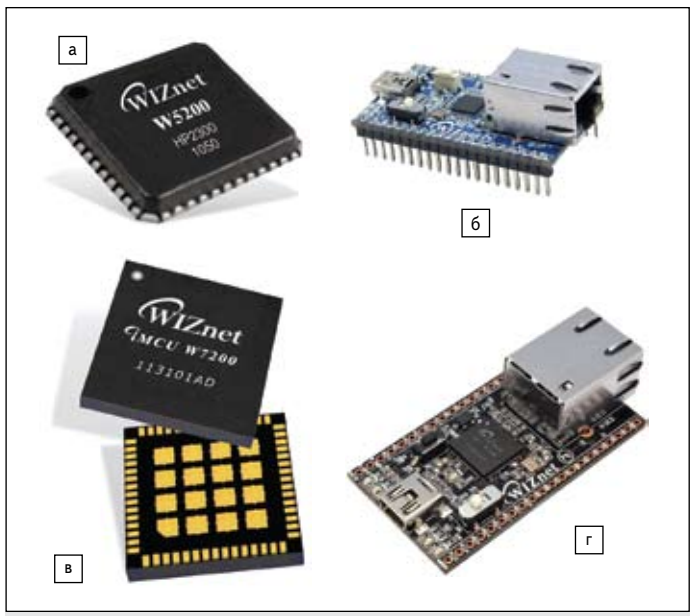

Рис. 1. Микросхемы WIZnet с режимами энергосбережения и тестовые модули: а) W5200; б) W5200E01-M3; в) W7200; г) iMCU7200 EVB

Отличия W5200 от микросхем предыдущих семейств:

- Компактный корпус 48-QFN.
- Наличие только последовательного интерфейса SPI (mode 0, 3), работающего на частоте до 80 МГц.
- Режимы энергосбережения (Power down и Wake-On-LAN).

Для тестирования микросхемы W5200 разработан модуль W5200E01-M3 (рис. 1б). В качестве управляющего контроллера используется ARM Cortex-M3 STM32F103C8, функции передачи по сети Ethernet выполняет W5200, а последовательный интерфейс связи реализован аппаратным USB-мостом FT232R. Также на плате установлены разъемы RJ45 со встроенным трансформатором и mini-USB B-типа, кнопка программного сброса, переключатель режимов работы «программирование – запуск», светодиоды, отражающие состояние работы модуля, и два светодиода для нужд пользователя. Внешние выводы контроллера STM32 соединены с двумя внешними штыревыми 20-выводными разъемами J1 и J2. Питание модуля может обеспечиваться по шине USB или через один из выводов разъема J1. На сайте компании [1] выложены готовые примеры для реализации различных сетевых приложений. Программирование Cortex-M3 осуществляется по интерфейсу UART-USB с помощью специальной утилиты [2].

Микросхема W7200 (рис. 1в) — это двухкристальный сетевой контроллер, объединяющий в одном корпусе 32-разрядный ARM-процессор Cortex-M3 и Ethernet-мост W5200.

Для тестирования возможностей микросхемы W7200 разработан модуль iMCU7200EVB (рис. 1г). Функции управляющего и сетевого контроллера исполняет W7200, а последовательный интерфейс связи реализован аппаратным USB-мостом FT232R. Также на плате установлены разъемы RJ45 со встроенным трансформатором и mini-USB B-типа, кнопка программного сброса, переключатель режимов работы «программирование – запуск», светодиоды, отражающие состояние работы модуля, и два светодиода для нужд пользователя. Внешние выводы контроллера W7200 соединены с двумя внешними штыревыми 20-выводными разъемами J1 и J2. Питание модуля может обеспечиваться по шине USB или через один из выводов разъема J1. Программирование и примеры аналогичны W5200E01-M3.

Применяемые средства

Для реализации режимов воспользуемся отладочными модулями W5200E01-M3 либо iMCU7200EVB. Эти модули имеют сходную структуру, отличающуюся лишь в деталях. В W5200E01-M3 в качестве управляющего используется контроллер STM32F103C8, имеющий 64 кбайт флэш-памяти для команд, а iMCU7200EVB содержит процессор STM32F103CB со 128 кбайт флэш-памяти. Линия прерывания nINT от W5200 подведена в W5200E01-M3 к выводу 0 порта B, а в iMCU7200EVB стыкуется с выводом 13 порта C контроллера STM32. Поэтому в программе будут отличаться и обработчики прерываний от Ethernet-контроллера. Для платы W5200E01-M3 определяется вектор по прерыванию на линии 0 EXTI0, а для iMCU7200EVB вектор по прерыванию находится на линиях с 10-й по 15-ю EXTI15_10. И наконец, светодиоды LED3 и LED4 управляются нулевым и первым выводом порта A W5200E01-M3, а у iMCU7200EVB для этого задействован порт B.

На сайте WIZnet [1] можно найти примеры программного обеспечения (ПО) для этого модуля, мы же реализуем HTTP-сервер (HyperText Transfer Protocol, протокол передачи гипертекста), добавим режимы пониженного электропитания и оценим степень энергосбережения.

Встроенный HTTP-сервер, часто также называемый web-сервером, является очень серьезным и перспективным дополнением любого сетевого устройства. Возможности такого сервера позволяют удаленно настраивать и контролировать работу оборудования с помощью стандартного web-браузера. HTTP сейчас используется повсеместно, поэтому доступ к устройству с web-интерфейсом возможен практически с любого компьютера и мобильного устройства. Реализуемый web-сервер позволяет настроить сетевые параметры (IP-адрес, маску и шлюз) и светодиоды LED3 и LED4 на отладочных платах. Это ПО мы выбрали еще и потому, что в этом случае модуль может работать автономно от ПК, что дает возможность оценить его электропотребление. В противном же случае питание модуля осуществляется через интерфейс USB.

Все web-страницы сервера в скомпонованном виде (файл *wizweb.rom*) находятся в директории ***…\binhex_util*** проекта W5200E01-M3_HTTPS. Образ со страницами располагается во флэш-памяти STM32 сразу за программным кодом, поэтому после каждого внесения изменений в программу контроллера необходимо корректировать начальный адрес расположения web-данных. Делается это следующим образом:

- Созданный проект компилируется и собирается, затем определяется размер полученного в директории ***…\Work\App\Debug\Exe*** двоичного файла *W5200EVB_App.bin*.
- Полученный размер переводится в шестнадцатеричный код и суммируется с адресом обращения к флэш-памяти 0x08000000. Полученная сумма меняется в файле *romfile.h* для макроопределения:

```
#define FLASH_ROMFILE_START_ADDRESS 0x08005578
//This value is determined by the size of f/w
```

- После внесения изменений проект необходимо пересобрать и полученный двоичный файл *W5200EVB_App.bin* скопировать в директорию ***…\binhex_util***, где уже находится образ с web-страницами *wizweb.rom*.
- Для объединения файлов необходимо запустить утилиту *allbin.bat*. В результате она создает скомпонованный файл *All_YYYYMMDD.bin*, где YYYY — текущий год, MM — текущий месяц, DD — текущий день.

Загрузка полученного программного кода в контроллер выполняется утилитой STM Flash Loader Demonstrator, которую можно взять с сайта компании-производителя [2]. Реализованный проект W5200E01-M3_HTTPS_POWER и программу отсылки magic-пакетов *ether_wake.exe* можно скачать по ссылке [3]. В зависимости от используемого модуля в файле *Types.h* нужно выбрать одно из макроопределений:

```
#define __DEF_W5200__
```

или

```
#define __DEF_W7200__
```

В проекте удалены ссылки на стандартную библиотеку периферии CMSIS, так как выполнение происходило в среде разработки IAR Embedded Workbench for ARM версии не ниже 6.2x, которая использует собственные библиотеки. При использовании кода на более ранних версиях EWARM следует руководствоваться документом [4].

Пробуждение по сети

Пробуждение по сети (Wake-On-LAN, WOL) — это технология, позволяющая удаленно включить компьютер посредством отправки через локальную сеть специального пакета данных (так называемого magic packet — «волшебного пакета»). Magic packet — это специальная последовательность байтов, рассылаемых широковещательно. Кроме стандартной информации в заголовке, пакет должен содержать в своем теле последовательность синхронизации (6 байт FF) и следующие за ней подряд 16 MAC-адресов устройства, для которого он предназначен.

Контроллер STM32 поддерживает три режима пониженного энергопотребления [5]:

- Режим сна (Sleep mode). В этом режиме останавливается только центральное процессорное устройство (ЦПУ). Все периферийные узлы продолжают работать в обычном режиме и могут пробуждать ЦПУ с помощью прерываний/событий. В этом режиме с включенными периферийными устройствами и тактированием с частотой 72 МГц от внешнего осциллятора HSE через фазовую автоподстройку частоты (ФАПЧ) энергопотребление составляет около 14,4 мА.
- Режим останова (Stop mode) обеспечивает наименьшее энергопотребление при сохранении содержимого оперативной памяти и регистров. Останавливаются все тактовые генераторы для блоков, которые питаются напряжением 1,8 В; ФАПЧ, кварцевые генераторы HSI RC и HSE RC выключаются. Стабилизатор напряжения может находиться в нормальном режиме или режиме пониженного энергопотребления. Выход из режима останова может осуществляться по сигналу контроллера внешних событий/прерываний EXTI (External interrupt/event controller). Источником EXTI могут быть одна из 16 линий внешних прерываний, часы реального времени, USB и узлы детектирования напряжения питания. В этом режиме энергопотребление падает до 24 мкА. Дальнейшее снижение энергопотребления до уровня порядка 14 мкА достигается за счет перевода встроенного регулятора напряжения в специальный экономичный режим.
- Режим ожидания (Standby mode) используется для перевода микроконтроллеров в состояние с наименьшим энергопотреблением. При этом отключается внутренний стабилизатор, выключая все блоки, питаемые от шины 1,8 В. ФАПЧ, HSI RC и HSE RC также переводятся в неактивное состояние. После входа в режим ожидания теряется информация в оперативной памяти и регистрах, за исключением информации, хранящейся в области резервирования. Из этого режима можно выйти по прерыванию от часов реального времени, по внешнему сбросу контроллера либо по нарастающему фронту на выводе 0 порта A. Этот вывод должен быть сконфигурирован для перевода микроконтроллера в активный режим (как вывод WKUP). Энергопотребление составляет около 2 мкА.

При возникновении события, требующего оповещения управляющего контроллера, W5200 сигнализирует об этом убывающим фронтом на выводе W5200_nINT (40-я ножка). В рассматриваемых модулях вывод прерывания W5200_nINT подведен к внешним портам микросхемы STM32 (к порту PB0 в W5200E01-M3 и к PC13 — в iMCU7200EVB). Поэтому самым экономичным режимом, из которого сетевой контроллер способен вывести STM32, является состояние останова.

Дополнения в основной файл программы *main.c*:

1. Настройка портов ввода/вывода общего назначения GPIO (General-Purpose Input/Output).

В функции ***GPIO_Configuration*** добавляется информация о параметрах выводов, к которым подключены светодиоды и сигнал прерывания от Ethernet-контроллера, а также происходит настройка контроллера EXTI.

Структура для инициализации светодиодов содержит следующие параметры:

- ***GPIO_Speed_50MHz*** — скорость переключения выводов;
- ***GPIO_Mode_Out_PP*** — выход с двумя состояниями (англ. Push-Pull).

Далее эти параметры инициализируются для того порта, на выводах которого подключены светодиоды.

```
    GPIO_InitStructure.GPIO_Pin |= LED3 | LED4;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
#ifdef __DEF_W5200__
    GPIO_Init(GPIOA, &GPIO_InitStructure);
    GPIO_SetBits(GPIOA, LED3);// led off
    GPIO_SetBits(GPIOA, LED4);// led off
#endif
#ifdef __DEF_W7200__
    GPIO_Init(GPIOB, &GPIO_InitStructure);
    GPIO_SetBits(GPIOB, LED3);// led off
    GPIO_SetBits(GPIOB, LED4);// led off
#endif
```

Структура для инициализации вывода для сигнала прерывания содержит следующие параметры:

- ***GPIO_Pin*** — номер вывода порта:
  0 для W5200E01-M3
  и 13 для iMCU7200EVB;
- ***GPIO_Speed_50MHz*** — скорость переключения выводов;
- ***GPIO_Mode_IPU*** — вход с подтяжкой к питанию (Pull-up).

Далее эти параметры инициализируются для того порта, на выводах которого подключен сигнал прерывания.

```
    // WIZ Interrupt
#ifdef __DEF_W5200__
    #define WIZ_INTGPIO_Pin_0
#endif
#ifdef __DEF_W7200__
    #define WIZ_INTGPIO_Pin_13
#endif

    GPIO_InitStructure.GPIO_Pin = WIZ_INT;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
    GPIO_Init(GPIOB, &GPIO_InitStructure);
#ifdef __DEF_W5200__
    GPIO_Init(GPIOB, &GPIO_InitStructure);
#endif
#ifdef __DEF_W7200__
    GPIO_Init(GPIOC, &GPIO_InitStructure);
#endif
```

Следующая структура инициализирует контроллер внешних прерываний/событий EXTI:

- ***EXTI_Mode_Interrupt*** — сигналом для срабатывания контроллера является внешнее прерывание.
- ***EXTI_Trigger_Falling*** — прерывания генерируются при спаде сигнала на входе EXTI.

```
#ifdef __DEF_W5200__
    GPIO_EXTILineConfig(GPIO_PortSourceGPIOB, GPIO_
PinSource0);
    EXTI_InitStructure.EXTI_Line = EXTI_Line0 ;
#endif
#ifdef __DEF_W7200__
    GPIO_EXTILineConfig(GPIO_PortSourceGPIOC, GPIO_
PinSource13);
    EXTI_InitStructure.EXTI_Line = EXTI_Line13;
#endif
    EXTI_InitStructure.EXTI_Mode = EXTI_Mode_Interrupt;
    EXTI_InitStructure.EXTI_Trigger = EXTI_Trigger_Falling;
    EXTI_InitStructure.EXTI_LineCmd = ENABLE;
    EXTI_Init(&EXTI_InitStructure);
```

2. Настройка контроллера вложенных векторных прерываний NVIC (Nested Vectored Interrupt Controller).

В функции ***NVIC_Configuration*** в качестве источника для срабатывания NVIC устанавливаем контроллер внешних событий/прерываний EXTI. Приоритет назначается нулевой, так как в данном примере это единственный сигнал, который может вывести STM32 из режима останова.

```
    /* Enable global Interrupt */
#ifdef __DEF_W5200__
    NVIC_InitStructure.NVIC_IRQChannel = EXTI0_IRQn;
#endif
#ifdef __DEF_W7200__
    NVIC_InitStructure.NVIC_IRQChannel = EXTI15_10_IRQn;
#endif
    NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0;
    NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
    NVIC_Init(&NVIC_InitStructure);
```

3. Переход в режим останова.

В случае изменения состояния светодиодов в меню **Digital Output→LED** через web-браузер на LED3-вкл., LED4-выкл. модуль переходит в состояние пониженного энергопотребления. Функция ***Enter_to_stop_mode*** переводит сетевой контроллер в состояние WOL мониторинга поступления magic-пакета по сети, а управляющий — в режим останова.

```
void ProcessWebSever(u_char ch)
{
…
        /* Check LEDs status */
        if ((chk_led & LED3) && !(chk_led & LED4)) // If LED3=1 &
LED4=0
            Enter_to_stop_mode();
…
}

void Enter_to_stop_mode (void)
{
    Delay_ms(1000);
    printf("\nStatus: STOP MODE, WOL ON \r\n");
    IINCHIP_WRITE(IMR, IR_MAGIC); // Enable the Magic Packet
Interrupt
    IINCHIP_WRITE(MR,MR_WOL); //Enter into the WOL mode

    PWR_EnterSTOPMode(PWR_Regulator_LowPower, PWR_
STOPEntry_WFI);
```

В Ethernet-контроллерах W5200 для управления функциями пробуждения по сети используются три регистра (рис. 2):

- регистр состояния MR;
- регистр прерываний IR;
- регистр маскирования прерываний IMR.

Установкой бита 4 (IR_MAGIC = 0x10) в состояние «лог. 1» регистра IMR разрешаем прерывание при обнаружении в сети адресованного устройству magic-пакета. Затем переводим сетевой контроллер в режим WOL, устанавливая бит 5 регистра MR в состояние «лог. 1» (MR_WOL = 0x20). Теперь, заметив предназначенный ему пакет пробуждения, контроллер W5200 просигнализирует об этом спадом сигнала на выводе nINT и установкой «лог. 1» в бите 4 регистра IR.

Перевод STM32 в режим останова осуществляется функцией ***PWR_EnterSTOPMode***, которая входит в стандартную библиотеку периферийных драйверов этого контроллера ***StdPeriph_Driver*** и реализована в файле *stm32f10x_pwr.c*.

Для входа в режим выполняются следующие действия:

- Устанавливается бит SLEEPDEEP в регистре управления питанием Cortex.
- Сбрасывается бит Power Down Deep Sleep (PDDS) в регистре управления питанием STM32.

MR (Mode Register) — Регистр состояния

7	6	5	4	3	2	1	0
RST		WOL	PB	PPPoE			

IR (Interrupt Register) — Регистр прерываний

7	6	5	4	3	2	1	0
CONFLICT		PPPoE	WOL				

IMR (Interrupt Mask Register) — Регистр маскирования прерываний

7	6	5	4	3	2	1	0
IR_IP		IR_PPP	IR_MAGIC				

PHYSTATUS (PHY ststus Register) — Регистр управления PHY-узлом

7	6	5	4	3	2	1	0
		LINK		PWDN			

Рис. 2. Регистры W5200 для работы с WOL

- Выполнение команды ***WFI*** (Wait For Interrupt) переводит контроллер в режим сниженного энергопотребления с возможностью вывода из него по прерыванию.

Добавление в файл обработки прерываний *stm32f10x_it.c*.

В обработчике EXTI0_IRQHandler реализован процесс вывода устройства из режима останова для модуля W5200E01-M3, а в обработчике EXTI 15_10_IRQHandler — для модуля iMCU7200EVB. Так как реакция на это событие должна происходить идентичным образом, то далее рассматривается только EXTI0_IRQHandler.

Функция *EXTI_GetITStatus* проверяет появление событий в регистре ожидания EXTI_PR от внешнего прерывания EXTI0. В случае положительного результата определяется причина. Наличие единицы в бите 4 регистра IR (IR_MAGIC = 0x08) контроллера W5200 означает, что поступил сигнал на пробуждение. При выходе из режима останова по сигналу прерывания кварцевый генератор HSI RC устанавливается в качестве источника тактирования внутренних узлов STM32. Поэтому необходимо изменить структуру тактирования контроллера в соответствии с настройками при начальной инициализации модуля, что и выполняется в функции ***RCC_reInit***. Сброс сигнала прерывания осуществляется обнулением регистра IMR сетевого контроллера W5200. Далее происходит перезапуск и инициализация W5200, а затем очищается флаг прерывания EXTI0 в регистре ожидания EXTI_PR микросхемы STM32.

```
void EXTI0_IRQHandler (void)
{
    wiz_NetInfo netinfo;

    if (EXTI_GetITStatus(EXTI_Line0) != RESET)
    {
        /* If WOL interrupt occurs */
        if ((IINCHIP_READ(IR)&IR_MAGIC) == IR_MAGIC)
        {
            RCC_reInit();
            IINCHIP_WRITE(IMR,0x00);   // reset interrupt
            printf("\r\nInterrupt reason = WOL");
            GetNetInfo(&netinfo);
            Reset_W5200();
            wizInit();
            SetNetInfo(&netinfo);
            printf("\r\nStatus: RUN MODE, WOL OFF");
        }
        EXTI_ClearITPendingBit(EXTI_Line0);
    }
}
```

Снижение электропитания сетевого контроллера

Самым энергоемким узлом сетевого контроллера является PHY, выполняющий преобразование данных в физические величины для передачи по каналам связи. W5200 поддерживает режим пониженного электропитания (power down mode). В этом режиме PHY-узел полностью отключается, а все остальные блоки продолжают функционировать. Таким образом, прекращается любое сетевое взаимодействие, но сохраняется возможность работы с внутренними регистрами и памятью контроллера. Это дает существенную экономию электропитания. Ведь в режиме активной передачи данных в сетях Fast Ethernet потребление составляет 160–175 мА, а при отключении PHY-узла оно снижается до 2–4 мА.

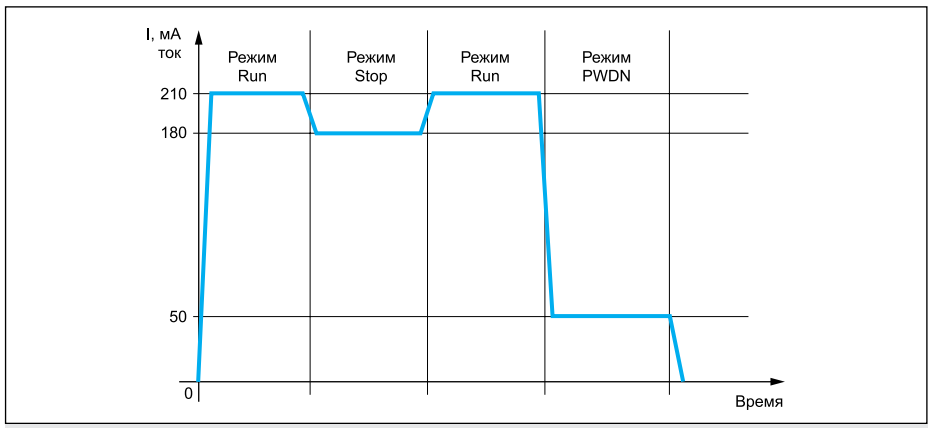

Рис. 3. Энергопотребление модуля в различных режимах

Управление режимом осуществляется выводом PWDN (45-я ножка) либо состоянием бита 3 регистра PHYSTATUS (рис. 2). Подача логической единицы на PWDN включает режим пониженного электропитания, логического нуля — отключает. То же самое и с регистром: установка бита 3 включает режим, а сброс — отключает.

Изменение состояния светодиодов через web-браузер в меню **Digital Output→LED** на LED3-выкл., LED4-вкл. переводит сетевой контроллер в состояние пониженного энергопотребления.

Функция ***Enter_to_pwrdown_mode*** выполняет установку логической единицы на выводе порта PB9 (WIZ_PWDN = GPIO_Pin_9), который подведен к порту PWDN сетевого контроллера. Чтение регистра PHYSTATUS должно подтвердить перевод состояния W5200 в режим POWER DOWN. Значение 0x08 означает, что бит 3 установлен и режим активирован:

```
void ProcessWebSever(u_char ch)
{
…
/* Check LEDs status */
if (!(chk_led & LED3) && (chk_led & LED4)) // If LED3=0 & LED4=1
                Enter_to_pwrdown_mode();
…
}

void Enter_to_pwrdown_mode (void)
{
    Delay_ms(1000);
    printf("\nStatus: POWER DOWN: ");
    if ((IINCHIP_READ(PHY)&0x08)==0x00) // If Power down disable
    {
        GPIO_SetBits(GPIOB,WIZ_PWDN);
        Delay_ms(500);
        if ((IINCHIP_READ(PHY)&0x08)==0x08)
            printf("ON");
    }
}
```

Заключение

Измерения потребляемого в этих режимах тока показали результаты, отраженные графиком на рис. 3. При полноценной работе в режиме HTTP-сервера (режим Run) модули потребляют 205–215 мА. Перевод модуля в режим останова (режим Stop) снижает энергопотребление до 180–190 мА. В этом случае регулятор напряжения переводится в состояние энергосбережения, останавливаются все тактовые генераторы. Пробуждение STM32 происходит по прерыванию от сетевого контроллера, обнаружившего посланный на его MAC-адрес magic-пакет. Процедура восстановления полноценной работы модуля составляет время, необходимое на пробуждение STM32 по прерыванию, равное согласно документации 7,21 мкс, перенастройку его линий тактирования и инициализацию W5200. В режиме пониженного электропитания сетевого контроллера (режим PWDN) суммарное энергопотребление модуля падает до 50 мА.

Из реализованных способов энергосбережения самым экономичным для модуля является переход в режим пониженного электропитания сетевого контроллера. Этот режим будет полезен при использовании модуля как основного управляющего устройства, где необходима непрерывная работа основных вычислительных узлов, а связь по сети нужна только в определенные моменты.

Но функция пробуждения по сети может быть очень эффективна для разрабатываемых изделий, к которым требуется организовать доступ по сети, например используя протокол HTTP. Модуль в этом случае будет играть роль расширения коммуникационных способностей устройства. Для экономии электропитания прибор переводится в режим останова с возможностью пробуждения в любое удобное время для проведения административных действий.

Литература

1. www.wiznet.co.kr
2. http://www.st.com/internet/mcu/product/216817.jsp
3. http://mymcu.ru/content/articles/Wiznet/WIZNET_POWER.rar
4. TN0830 Technical Note. How to use EWARM 6.2x with projects built with EWARM 6.1 and previous versions.
5. AN2629 Application Note. STM32F101xx, STM32 F102xx and STM32F103xx low-power modes.