



Валерий Жижин (ЗАО «ИПК «СТРАЖ»)

## ПРОГРАММИРОВАНИЕ РАДИОМОДУЛЯ PAN 2355

*Радиомодуль PAN 2355 компании Panasonic Industrial GMBH, реализованный на базе полудуплексного приемопередатчика CC1100 компании Texas Instruments, предназначен для построения узкополосных систем малого радиуса действия в диапазоне 868 МГц. В статье идет речь об особенностях программирования модуля на этапах конфигурации и передачи данных. Приводится описание практической реализации беспроводной системы сбора данных на базе PAN 2355 с удаленных сенсоров.*

В последнее время многие российские разработчики беспроводных сенсорных систем останавливают свой выбор на радиомодулях, использующих приемопередатчики Texas Instruments из линейки Chipcon, в частности, радиомодулях ISM диапазона PAN 23xx. Это связано с тем, что модули PAN 23xx надежны, просты в использовании, не требуют усилий в изготовлении радиочастотных цепей, характеризуются очень малыми размерами (PAN 2355 – 8x8,2x2 мм) и невысокой стоимостью.

Внутренняя архитектура и основные технические характеристики PAN 2355 подробно описаны в работе [1]. Однако в вопросах практического применения и особенно программирования радиомодулей существует определенный пробел, несмотря на появление на сайте <http://www.ti.com/> библиотеки функций для работы с CCxxx0. В статье излагается практическое применение этих функций, адаптированных для микроконтроллера MSP430F1232A.

Рассмотрим автономную беспроводную радиосистему сбора данных с двух температурных датчиков с линейным выходом по напряжению 0...3 В. Основные характеристики системы следующие:

- дальность 30 м на открытом пространстве;
- мощность передатчика 10 мВт;
- скорость передачи 115 кБит/с;
- метод модуляции – MSK;
- CRC – контроль.

На рис. 1 и рис. 2 приведены схемы передатчика и приемника соответственно.

Алгоритм работы системы заключается в следующем.

Датчики подключены к каналам АЦП А6 и А7 микроконтроллера MSP430F1232A передатчика, который попеременно их включает на 2 мс со скважностью 1:1000. Считанные из буферов АЦП А6 и А7 данные усредняются за время 30 с и результат

(1 байт/канал) передается в модуль PAN 2355, сконфигурированный в режим TX. На приемном конце аналогичный радиомодуль, сконфигурированный в режим RX, транслирует полученные данные в микроконтроллер, который через UART-порт по интерфейсу RS-232 передает информацию с датчиков в СОМ порт ПК. Радиомодули PAN 2355 подключены к микроконтроллерам через 3-проводной интерфейс SPI.

Для комфортной работы с модулем желательно использовать еще две линии – GDO0 и GDO2.

Всего в трансивере используется 47 конфигурационных регистров. Значения некоторых из них (примерно 12) можно оставить по умолчанию. Для расчета параметров конфигурации можно воспользоваться Data Sheet на микросхему CC1100, но лучше это сделать с помощью программы SmartRF\_Studio, которую можно бесплатно скачать с сайта <http://www.chipcon.com/>. При передаче сигнала по интерфейсу SPI используется аппаратный модуль микроконтроллера MSP430F1232A.

Собственно программа передачи/приема данных с использованием PAN 2355 состоит из вызовов следующих основных функций: конфигурации SPI-интерфейса MSP430F1232A, первоначального сброса модуля, функции записи конфигурационных параметров, функции записи стробов, функции передачи байта и функции приема байта. Функция установки конфигурационных параметров радиомодуля должна вызываться после функции сброса модуля. Шестнадцатеричные адреса регистров конфигурации приводятся в документации на CC1100, также можно воспользоваться заголовочным файлом TI\_CC\_CC1100 – CC2500.h, приведенным в библиотеке функций для работы с Chipcon.

Рассмотрим эти функции:

### Функция конфигурации SPY модуля MSP430F1232A

Модуль должен быть сконфигурирован в режим Master.

```
void TI_CC_SPISetup(void)
{
    TI_CC_CS_n_PxDIR |= TI_CC_CS_n_PIN; // P3.0 -> OUT
    TI_CC_CS_n_PxOUT |= TI_CC_CS_n_PIN; // CS_n = 1
    ME2 |= USPIE0;
    UCTL0 |= CHAR + SYNC + MM;
    UTCTL0 |= CKPL + SSEL1 + SSEL0 + STC; //SPI 3-pin
    mode
```

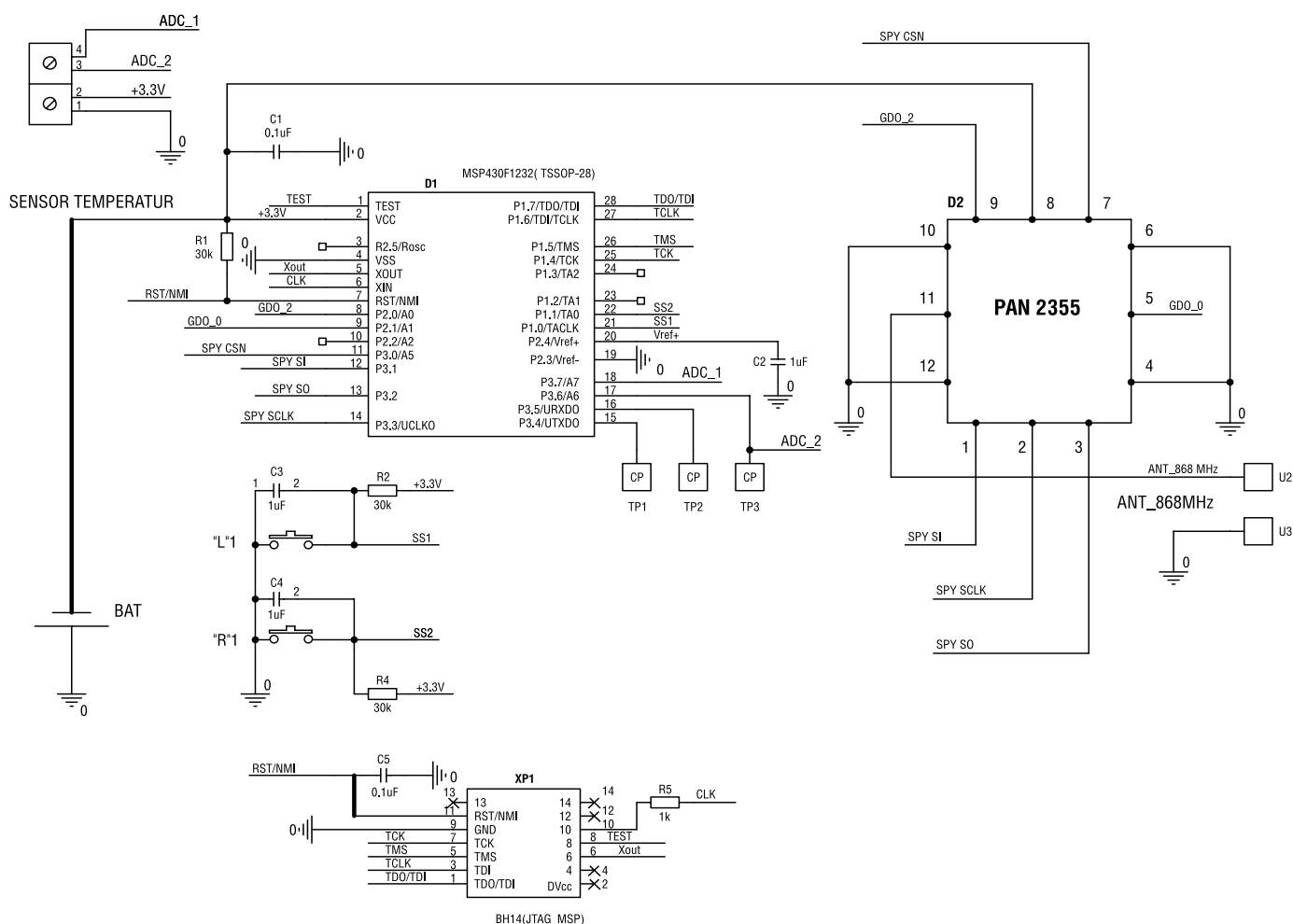


Рис. 1. Схема передатчика системы сбора данных

```
UBR00 = 0x02; // UCLK/2 -> 115kHz
UBR10 = 0x00;
UMCTL0 = 0x00;
UCTL0 &= ~SWRST;
}
```

**Функция сброса модуля**

```
void TI_CC_PowerupResetCCxxxx (void)
{
//***** 1->0->1*****
TI_CC_CS_n_PxOUT |= TI_CC_CS_n_PIN; // CS_n = 1
Delay1(1);
TI_CC_CS_n_PxOUT &= ~TI_CC_CS_n_PIN; // CS_n = 0
Delay1(1);
TI_CC_CS_n_PxOUT |= TI_CC_CS_n_PIN; // CS_n = 1
Delay1(1);
TI_CC_CS_n_PxOUT &= ~TI_CC_CS_n_PIN; // CS_n = 0
// ожидание события SOMI = 0
while(TI_CC_SPI_USART0_PxIN&TI_CC_SPI_USART0_SOMI);
U0TXBUF = TI_CCxxx0_SRES; // Строб сброса
IFG2 &= ~URXIFG0; // Сброс флага
// ожидание окончания пересылки строба
while (!(IFG2&URXIFG0));
while (TI_CC_SPI_USART0_PxIN&TI_CC_SPI_USART0_SOMI); TI_CC_CS_n_PxOUT |= TI_CC_CS_n_PIN; // CS_n = 1
}
```

**Функция конфигурации PAN 2355**

```
void writeRFSettings (void)
{
// Запись установочных значений
// GDO0 , GDO2 - вывод
```

```
TI_CC_SPIWriteReg(TI_CCxxx0_IOCFG0, 0x06);
TI_CC_SPIWriteReg(TI_CCxxx0_IOCFG0, 0x06);
// Длина пакета данных - 1 байт
TI_CC_SPIWriteReg(TI_CCxxx0_PKTLEN, 0x01);
// Автоматический контроль пакета без проверки адреса
TI_CC_SPIWriteReg(TI_CCxxx0_PKTCTRL1, 0x04);
// CRC - контроль включен
TI_CC_SPIWriteReg(TI_CCxxx0_PKTCTRL0, 0x05);
// Адрес устройства - 0
TI_CC_SPIWriteReg(TI_CCxxx0_ADDR, 0x00);
// Номер канала - 0
TI_CC_SPIWriteReg(TI_CCxxx0_CHANNR, 0x00);
// Настройка синтезатора частоты
TI_CC_SPIWriteReg(TI_CCxxx0_FSCTRL1, 0x0B);
TI_CC_SPIWriteReg(TI_CCxxx0_FSCTRL0, 0x00);
// Задание базовой несущей частоты
TI_CC_SPIWriteReg(TI_CCxxx0_FREQ2, 0x21);
TI_CC_SPIWriteReg(TI_CCxxx0_FREQ1, 0x62);
TI_CC_SPIWriteReg(TI_CCxxx0_FREQ0, 0x76);
// Настройка модема
// ФНЧ демодулятора - 203 кГц
TI_CC_SPIWriteReg(TI_CCxxx0_MDMCFG4, 0x2C);
// Скорость передачи 115 кбит/с
TI_CC_SPIWriteReg(TI_CCxxx0_MDMCFG3, 0x22);
// Длина слова синхронизации - 32 бита, MSK модуляция
TI_CC_SPIWriteReg(TI_CCxxx0_MDMCFG2, 0x73);
// Длина преамбулы 4 байта, FEC отключен
TI_CC_SPIWriteReg(TI_CCxxx0_MDMCFG1, 0x22);
// Полоса канала - 199.95 кГц
TI_CC_SPIWriteReg(TI_CCxxx0_MDMCFG0, 0xF8);
```

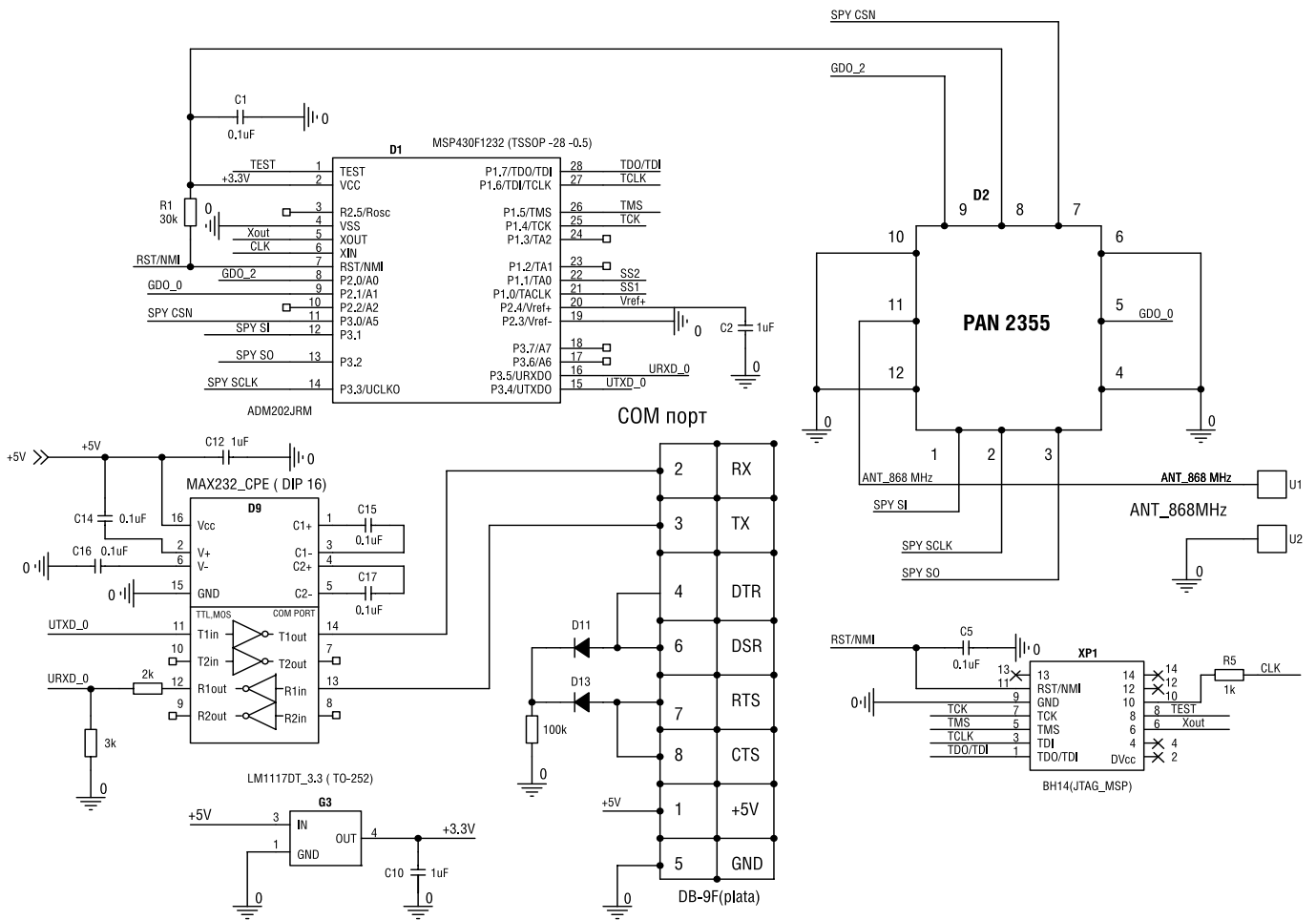


Рис. 2. Схема приемника системы сбора данных

```
// Установка мощности передатчика - для paTable[
]={0xC3} составляет 10 мВт (см. Data Sheet на мик-
росхему CC1100)
TI_CC_SPIWriteBurstReg(TI_CCxxx0_PATABLE, paTable,
1);
// Завершение конфигурации RX
TI_CC_SPIWriteReg(TI_CCxxx0_FREND1, 0xB6);
// Завершение конфигурации TX
TI_CC_SPIWriteReg(TI_CCxxx0_FREND0, 0x10);
//Функции конфигурации автомата контроля радио (по
умолчанию)
TI_CC_SPIWriteReg(TI_CCxxx0_MCSM2, 0x07);
TI_CC_SPIWriteReg(TI_CCxxx0_MCSM1, 0x3F);
TI_CC_SPIWriteReg(TI_CCxxx0_MCSM0, 0x18);
// Компенсация сдвига частоты
TI_CC_SPIWriteReg(TI_CCxxx0_FOCCFG, 0x1D);
// Конфигурация битовой синхронизации
TI_CC_SPIWriteReg(TI_CCxxx0_BSCFG, 0x1C);
// Параметры маломощных усилителей и порог компа-
ратора
TI_CC_SPIWriteReg(TI_CCxxx0_AGCCTRL2, 0xC7);
TI_CC_SPIWriteReg(TI_CCxxx0_AGCCTRL1, 0x00);
TI_CC_SPIWriteReg(TI_CCxxx0_AGCCTRL0, 0xB2);
// Параметры калибровки синтезатора
TI_CC_SPIWriteReg(TI_CCxxx0_FSCAL3, 0xEA);
TI_CC_SPIWriteReg(TI_CCxxx0_FSCAL2, 0x0A);
TI_CC_SPIWriteReg(TI_CCxxx0_FSCAL1, 0x00);
TI_CC_SPIWriteReg(TI_CCxxx0_FSCAL0, 0x11);
// Приведенные ниже настройки изменять не рекомен-
дуется
TI_CC_SPIWriteReg(TI_CCxxx0_FSTEST, 0x59);
TI_CC_SPIWriteReg(TI_CCxxx0_TEST2, 0x88);
```

```
TI_CC_SPIWriteReg(TI_CCxxx0_TEST1, 0x31);
TI_CC_SPIWriteReg(TI_CCxxx0_TEST0, 0x0B);
// Остальные настройки по умолчанию
}
```

**Функция записи строба**

```
void TI_CC_SPIStroke(char strobe)
{
TI_CC_CS_n_PxOUT &= ~TI_CC_CS_n_PIN; // CSn = 0
while (TI_CC_SPI_USART0_PxIN&TI_CC_SPI_USART0_
SOMI);
U0TXBUF = strobe;
IFG2 &= ~URXIFG0; // Сброс флага
while (!(IFG2&URXIFG0));
TI_CC_CS_n_PxOUT |= TI_CC_CS_n_PIN; // CSn = 1
}
```

**Функция передачи байта**

```
Данные передаются по адресу TI_CCxxx0_
TXFIFO (0x3F). После записи стробов целесообразно
вводить небольшие временные задержки пример-
но 35 мкс.
void RFTransmitter(char txBuffer)
{
// Сброс TX_FIFO
TI_CC_SPIStroke(TI_CCxxx0_SFTX);
Delay1(1); // задержка после записи строба
TI_CC_SPIWriteReg(TI_CCxxx0_TXFIFO, txBuffer);
Delay1(1);
// Строб включения режима TX
TI_CC_SPIStroke(TI_CCxxx0_STX);
```

```

Delay1(1);
// Ожидание окончания передачи байта
while(TI_CC_GD02_PxIN&TI_CC_GD02_PIN);
// Сброс TX_FIFO – обязательно.
TI_CC_SPIStrobe(TI_CCxxx0_SFTX);
Delay1(1);
// Выход из режима TX_FIFO
TI_CC_SPIStrobe( TI_CCxxx0_SIDLE);
}

```

### Функция записи пакета данных

Все пересылки между контроллером и модулем PAN 2355 начинаются с адресного байта, содержащего 6-битный адрес требуемого регистра, бит burst – доступа и бит чтения/записи. Байты данных следуют после адресного байта. Если burst-бит установлен, то байтов данных может быть несколько. Передача ведется старшим битом вперед.

Addr – адрес регистра, \*buffer – указатель массива байтов, count – размер массива

```

void TI_CC_SPIWriteBurstReg(char addr, char
*buffer, char count)
{
char i;
TI_CC_CSn_PxOUT &= ~TI_CC_CSn_PIN; // CSn = 0
while (TI_CC_SPI_USART0_PxIN&TI_CC_SPI_USART0_
SOMI);
// Выставляем адрес нужного регистра
U0TXBUF = addr | TI_CCxxx0_WRITE_BURST;
while (!(IFG2&UTXIFG0));
for (i = 0; i < count; i++)
{
U0TXBUF = buffer[ i ]; //Пересылка данных
while (!(IFG2&UTXIFG0));
}
IFG2 &= ~URXIFG0; // Сброс флага
while(!(IFG2&URXIFG0));
TI_CC_CSn_PxOUT |= TI_CC_CSn_PIN; // CSn = 1
}

```

### Функция записи байта по адресу addr

```

void TI_CC_SPIWriteReg(char addr, char value)
{
TI_CC_CSn_PxOUT &= ~TI_CC_CSn_PIN; // CSn = 0
while (TI_CC_SPI_USART0_PxIN&TI_CC_SPI_USART0_
SOMI);
IFG2 &= ~URXIFG0;
// Выставляем адрес нужного регистра
U0TXBUF = addr;
while (!(IFG2&URXIFG0));
IFG2 &= ~URXIFG0; // Сброс флага
U0TXBUF = value;
while (!(IFG2&URXIFG0));
TI_CC_CSn_PxOUT |= TI_CC_CSn_PIN; // CSn = 1
}

```

### Функция приема байта

Байт данных считывается из RX\_FIFO по адресу

**addr = TI\_CCxxx0\_RXFIFO (0xBF).**

```

char RFRceiver(char addr)
{
char z;
// Сброс RX_FIFO
TI_CC_SPIStrobe(TI_CCxxx0_SFRX);
Delay1(1); // задержка после записи строба
// Строб включения режима RX
TI_CC_SPIStrobe(TI_CCxxx0_SRX);
Delay1(1);
// Ожидание приема байта(GD02 -> 1)

```

```

while(!(TI_CC_GD02_PxIN&TI_CC_GD02_PIN));
// Считывание байта данных
z =TI_CC_SPIReadReg(addr);
// Сброс RX_FIFO – обязательно.
TI_CC_SPIStrobe(TI_CCxxx0_SFRX);
Delay1(1);
// Выход из режима RX_FIFO
TI_CC_SPIStrobe( TI_CCxxx0_SIDLE);
return z;
}

```

### Функция считывания байта

Сначала передается адрес регистра, из которого нужно прочесть данные, затем посылается нулевой байт, чтобы считать данные.

```

char TI_CC_SPIReadReg(char addr1)
{
char x;
TI_CC_CSn_PxOUT &= ~TI_CC_CSn_PIN; // CSn = 0
while (TI_CC_SPI_USART0_PxIN&TI_CC_SPI_USART0_
SOMI);
// Установка адреса
U0TXBUF = (addr1 | TI_CCxxx0_READ_SINGLE);
while (!(IFG2&URXIFG0));
IFG2 &= ~URXIFG0; // Сброс флага
// Передача нулевого байта
U0TXBUF = 0;
while (!(IFG2&URXIFG0));
// Считывание байта из буфера
x = U0RXBUF;
TI_CC_CSn_PxOUT |= TI_CC_CSn_PIN; // CSn = 1
return x;
}

```

Определение переменных, используемых в приведенных функциях (применительно к микроконтроллеру MSP430F1232A):

```

#define TI_CC_CSn_PxOUT P3OUT
#define TI_CC_CSn_PxDIR P3DIR
#define TI_CC_CSn_PIN 0x01 //P3.0
#define TI_CC_SPI_USART0_PxIN P3IN
#define TI_CC_SPI_USART0_SIMO 0x02 //P3.1
#define TI_CC_SPI_USART0_SOMI 0x04 //P3.2
#define TI_CC_SPI_USART0_UCLK 0x08 //P3.3
#define TI_CCxxx0_PATABLE 0x3E
#define TI_CC_GD02_PIN 0x01 //P2.0
#define TI_CC_GD02_PxIN P2IN
#define TI_CCxxx0_PATABLE 0x3E

```

Программа для дистанционной системы сбора данных, созданная с использованием приведенных выше функций, показала устойчивую работу в реальных устройствах и может быть использована для работы с трансиверами CC11xx – CC2500. При разработке программы были также использованы материалы[2].

### Литература

1. Приемопередатчики PAN 2355./Беспроводные технологии №4, 2007 г.
2. Семейство микроконтроллеров MSP430x1xx. Руководство пользователя./«Библиотека Компэла».

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