Phi Robotics

Product Manual

NRF905

Version 1.0

Phi Robotics Research Pvt. Ltd.

www.phi-robotics.com

Table of Contents

1 Introduction

NRF905 is a single chip radio transceiver for the 433/868/915MHz ISM band. The transceiver consists of a fully integrated frequency synthesizer, receiver chain with demodulator, power amplifier, crystal oscillator and modulator. The Shock Burst™ feature automatically handles preamble and CRC. You can easily configure the NRF905 using SPI bus. Current consumption is very low. It consumes 9 mA in transmit mode at an output power of -10dBm and 12.5 mA in receive mode. Built-in power down mode makes it easy to save power. NRF905 allows wireless communication for microcontrollers. Since the module itself handles CRC and preamble, it avoids communication errors inherently, thereby removing the need for error checking in microcontroller firmware. The module also has optional auto-retransmission. NRF905 uses addressed communication, each module can be configured runtime with an address. Since it has an SPI interface for configuration as well data transfer, it provides a structured way to handle the data. Address width (1 to 4 bytes) and data width (1 to 32 bytes) is configurable, thus providing flexibility to the user as per application.

2 Features

- Multi-channel operation
- Adjustable output power up to 10 dBm
- Automatic retransmission of data packet
- Automatic CRC and preamble handling
- Optional auto-retransmission
- Easy and compact in size
- SPI interface for command and data transfer

3 Specifications

- Operating voltage: 1.9V to 3.6V
- On-board crystal: 16 MHz
- Data rate: 50 kbps
- Data size per packet: 1 to 32 bytes
- Max transmitter output power: 10 dBm
- Operating frequency: 430 to 928 KHz
- Supply current (at 10 dBm output power): 30 mA
- Supply current in receive mode: 12.5 mA
- Supply current in power down mode: 2.5 µA

4 Hardware Connections

NRF905 module is provided with 14 pin male header. The header has power supply pins, SPI lines for communication and few control and status pins. TXE is used for selection between TX or RX mode. Logic high signal on TXE selects TX mode while logic low signal selects RX mode. CE is used to enable chip to receive and transmit the data. CD indicates status for carrier detection, AM indicates status for address match. DR is data ready pin which indicates that the module is ready to transmit or receive the data. Figure 1 below shows pin layout for NRF905 module.

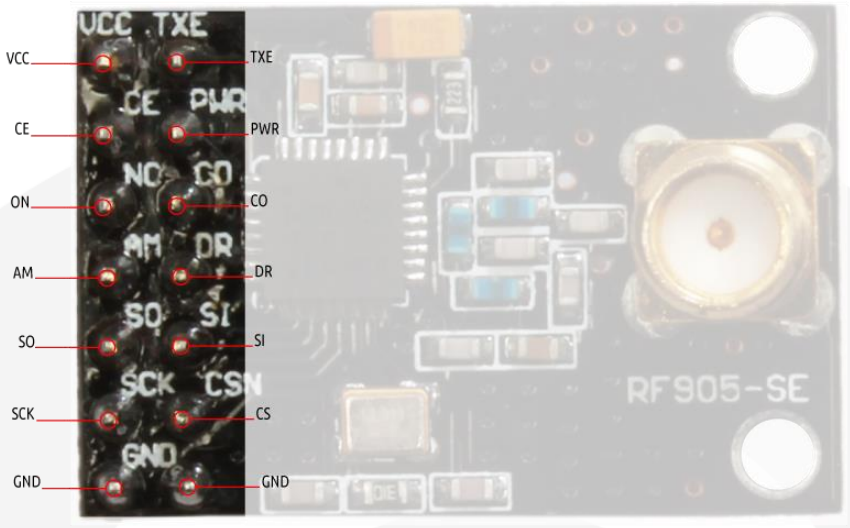

Figure 1 - NRF905 sensor pin layout

5 Pseudo Code

5.1 NRF905 Commands and Configuration Parameters

```
// NRF905 commands
// write configuration
NRF_CMD_WR_CFG = 0x00
// write transmit data
NRF_CMD_WR_TX_DATA = 0x20
// write transmit address
NRF_CMD_WR_TX_ADDR = 0x22
// read transmit data
NRF_CMD_RD_TX_ADDR = 0x23
// read receive data
NRF_CMD_RD_RX_DATA = 0x24

// NRF905 configuration parameters
// NRF905 peer address should be configured by user
NRF_ADDR = 0x12345678
// NRF905 address length can be set to 1 to 4 bytes
NRF_ADDR_LENGTH = 4
// NRF905 data length can be set to 1 to 32 bytes
NRF_DATA_LENGTH = 32
// For receive mode
NRF_TX_RX_MODE = RX
// For transmit mode
NRF_TX_RX_MODE = TX
```

```
// select centre frequency together with NRF_HFREQ_PLL
NRF_CH_NO = 108
// PLL mode = 433
NRF_HFREQ_PLL = 0
// output power = -10dBm
NRF_PA_PWR = 0
// reduced power operation disabled
NRF_RX_RED_PWR = 0
// no retransmission of data packet
NRF_AUTO_RETRAN = 0
// RX address width
NRF_RX_AFW = NRF_ADDR_LENGTH
// TX address width
NRF_TX_AFW = NRF_ADDR_LENGTH
// RX payload width
NRF_RX_PW = NRF_DATA_LENGTH
// TX payload width
NRF_TX_PW = NRF_DATA_LENGTH
// self-address
NRF_RX_ADDR = NRF_ADDR
// output clock frequency = 500KHz
NRF_UP_CLK_FREQ = 3
// external clock signal enabled
NRF_UP_CLK_EN = 1
// crystal oscillator = 16MHz
NRF_XOF = 3
// CRC check enabled
NRF_CRC_EN = 1
// 16 bit CRC mode
NRF_CRC_MODE = 1
```

5.2 Status and Control Functions

```
void nrfEnable(void)
{
      gpioSet(CE);
      gpioSet(PWR);
}

void nrfDisable(void)
{
      gpioClear(CE);
      gpioClear(PWR);
}

void nrfSetMode(uint8_t mode)
{
      if(mode == TX)    // NRF905 transmitter mode
            gpioSet(TXE);
      else              // NRF905 receiver mode
            gpioClear(TXE);
}

boolean nrfIsDataReady(void)
{
      if(gpioReadPin(DR) == 0)
            return false;
      else
            return true;
}
```

5.3 Configuring the Device

```
/*
 NRF905 configuration parameters are not divided in exact register sizes
 hence, all the parameters are assembled in a 10 byte array and
 these 10 bytes are written to device in a single write cycle
*/
void nrfInit(void)
{
      uint8_t cfg_data[10] = {0};

      // set TX/RX mode
      nrfSetMode(NRF_TX_RX_MODE);

      cfg_data[0] = (NRF_CH_NO & 0xFF);
      cfg_data[1] = ((NRF_CH_NO >> 8) & 0x01)
                  | ((NRF_HFREQ_PLL & 0x01) << 1)
                  | ((NRF_RX_RED_PWR & 0x01) << 4)
                  | ((NRF_AUTO_RETRAN & 0x01) << 5);
      cfg_data[2] = (NRF_RX_AFW & 0x07) | ((NRF_TX_AFW & 0x07) << 4);
      cfg_data[3] = (NRF_RX_PW & 0x3F);
      cfg_data[4] = (NRF_TX_PW & 0x3F);
      cfg_data[5] = (NRF_RX_ADDR & 0xFF);
      cfg_data[6] = ((NRF_RX_ADDR >> 8) & 0xFF);
      cfg_data[7] = ((NRF_RX_ADDR >> 16) & 0xFF);
      cfg_data[8] = (NRF_RX_ADDR >> 24);
      cfg_data[9] = (NRF_UP_CLK_FREQ & 0x03)
                  | ((NRF_UP_CLK_EN & 0x01) << 2)
                  | ((NRF_XOF & 0x03) << 3) | ((NRF_CRC_EN & 0x01) << 6)
                  | ((NRF_CRC_MODE & 0x01) << 7);

      // disable NRF905 before any read write access to registers
      nrfDisable();
      spiChipSelect(0);
      spiWriteByte(NRF_CMD_NRF_CMD_WR_CFG);
      spiWriteData(cfg_data, 10);
      spiChipSelect(1);
      // enable NRF905 after register read/write is complete
      nrfEnable();
}
```

5.4 Transmitting Data

```
void nrfSendData(uint32_t addr, uint8_t* data, uint8_t len)
{
	nrfWriteAddr(addr);
	nrfWriteData(data, len);
}

void nrfWriteAddr(uint32_t addr)
{
	nrfDisable();

	spiChipSelect(0);
	spiWriteByte(NRF_CMD_WR_TX_ADDR);
	spiWriteData(&addr, 4);
	spiChipSelect(1);

	nrfEnable();
}

void nrfWriteData(uint8_t* data, uint8_t len)
{
	nrfDisable();

	spiChipSelect(0);
	spiWriteByte(NRF_CMD_WR_TX_DATA);
	spiWriteData(data, len);
	spiChipSelect(1);

	nrfEnable();
}
```

5.5 Receiving Data

```
bool nrfReceiveData(uint32_t* addr, uint8_t* data, uint8_t len)
{
	// check data ready status
	if(nrfIsDataReady() == false)
		return false;

	*addr = nrdReadAddress();
	nrfReadData(data, len);
}

uint32_t nrfReadAddress(void)
{
	uint32_t addr;
	nrfDisable();
	spiChipSelect(channel, 0);
	spiWriteByte(NRF_CMD_RD_TX_ADDR);
	spiReceiveData(addr, 4);
	spiChipSelect(1);
	nrfEnable();

	return addr;
}

void nrfReadData(uint8_t* data, uint8_t len)
{
	uint8_t cmd = NRF_CMD_RD_RX_DATA;

	nrfDisable();

	spiChipSelect(0);
	spiWriteData(NRF_CMD_RD_RX_DATA);
	spiReadData(data, len);
	spiChipSelect(1);

	nrfEnable();
}
```

6 Reference

NRF905 Datasheet: http://www.nordicsemi.com/eng/Products/Sub-1-GHz-RF/nRF905

Phi Robotics Research Pvt. Ltd.

www.phi-robotics.com