# THE DATASHEET OF

# NRF24LE1-O17Q32-T

www.win-source.net 0086-755-83957316

# nRF24LE1 OTP

## Ultra-low Power Wireless System On-Chip Solution

# Product Specification v1.2

### Key Features

- nRF24L01+ 2.4 GHz transceiver (250 kbps, 1 Mbps and 2 Mbps air data rates)
- Fast microcontroller (8051 compatible)
- 16 kB program memory (on-chip OTP)
- 1 kB data memory (on-chip RAM)
- 1 kB OTP data memory
- AES encryption HW accelerator
- 16-32 bit multiplication/division co-processor (MDU)
- 6–12 bit ADC
- High flexibility IOs
- Serves a set of power modes from ultra low power to a power efficient active mode
- Several versions in various QFN packages:
  - 4×4 mm QFN24
  - 5×5 mm QFN32
  - 7×7 mm QFN48

### Applications

- Computer peripherals
  - Mouse
  - Keyboard
  - Remote control
  - Gaming
- Advanced remote controls
  - Audio/Video
  - Entertainment centers
  - Home appliances
- Goods tracking and monitoring:
  - Active RFID
  - Sensor networks
- Security systems
  - Payment
  - Alarm
  - Access control
- Health, wellness and sports
  - Watches
  - Mini computers
  - Sensors
- Remote control toys

All rights reserved.
Reproduction in whole or in part is prohibited without the prior written permission of the copyright holder.
September 2010

## Liability disclaimer

Nordic Semiconductor ASA reserves the right to make changes without further notice to the product to improve reliability, function or design. Nordic Semiconductor ASA does not assume any liability arising out of the application or use of any product or circuits described herein.

All application information is advisory and does not form part of the specification.

## Limiting values

Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the specifications are not implied. Exposure to limiting values for extended periods may affect device reliability.

### Life support applications

Nordic Semiconductor's products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Nordic Semiconductor ASA customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nordic Semiconductor ASA for any damages resulting from such improper use or sale.

| Data sheet status | |
|---|---|
| Objective product specification | This product specification contains target specifications for product development. |
| Preliminary product specification | This product specification contains preliminary data; supplementary data may be published from Nordic Semiconductor ASA later. |
| Product specification | This product specification contains final product specifications. Nordic Semiconductor ASA reserves the right to make changes at any time without notice in order to improve design and supply the best possible product. |

## Contact details

For your nearest dealer, please see www.nordicsemi.com

**Main office:**
Otto Nielsens veg 12
7004 Trondheim
Phone: +47 72 89 89 00
Fax: +47 72 89 89 89
www.nordicsemi.com

DNV
NORWEGIAN ACCREDITATION
No. S03
NS-EN ISO 9001 CERTIFICATED FIRM

# Revision History

| Date | Version | Description |
|---|---|---|
| October 2009 | 1.0 | Product specification |
| May 2010 | 1.1 | Updated section 2.1 on page 12, Table 14. on page 58, Table 15. on page 59, Figure 34. on page 77, Table 34. on page 77, Table 57. on page 106, Table 113. on page 177, chapter 12 on page 112, chapter 24 on page 171. |
| September 2010 | 1.2 | Updated sections 2.4 on page 16, 6.3.6.1 and 6.3.6.2 on page 75, Table 56. on page 104, Table 57. on page 106, Table 87. on page 146 , chapter 24 on page 171 and chapter 28 on page 182. |

## RoHS statement

Nordic Semiconductor's products meet the requirements of Directive 2002/95/EC of the European Parliament and of the Council on the Restriction of Hazardous Substances (RoHS). Complete hazardous substance reports as well as material composition reports for all active Nordic products can be found on our web site www.nordicsemi.com.

# Contents


**1 Introduction ..... 11**
1.1 Prerequisites ..... 11
1.2 Writing conventions ..... 11
**2 Product overview ..... 12**
2.1 Features ..... 12
2.2 Block diagram ..... 14
2.3 Pin assignments ..... 15
2.3.1 24-pin 4×4 QFN-package variant ..... 15
2.3.2 32-pin 5×5 QFN-package variant ..... 15
2.3.3 48-pin 7×7 QFN-package variant ..... 16
2.4 Pin functions ..... 16
**3 RF transceiver ..... 17**
3.1 Features ..... 17
3.2 Block diagram ..... 18
3.3 Functional description ..... 18
3.3.1 Operational Modes ..... 18
3.3.1.1 State diagram ..... 18
3.3.1.2 Power down mode ..... 19
3.3.1.3 Standby modes ..... 19
3.3.1.4 RX mode ..... 20
3.3.1.5 TX mode ..... 20
3.3.1.6 Operational modes configuration ..... 21
3.3.1.7 Timing information ..... 21
3.3.2 Air data rate ..... 22
3.3.3 RF channel frequency ..... 22
3.3.4 Received Power Detector measurements ..... 22
3.3.5 PA control ..... 22
3.3.6 RX/TX control ..... 23
3.4 Enhanced ShockBurst™ ..... 23
3.4.1 Features ..... 23
3.4.2 Enhanced ShockBurst™ overview ..... 23
3.4.3 Enhanced Shockburst™ packet format ..... 24
3.4.3.1 Preamble ..... 24
3.4.3.2 Address ..... 24
3.4.3.3 Packet Control Field (PCF) ..... 25
3.4.3.4 Payload ..... 25
3.4.3.5 CRC (Cyclic Redundancy Check) ..... 26
3.4.4 Automatic packet assembly ..... 27
3.4.5 Automatic packet disassembly ..... 28
3.4.6 Automatic packet transaction handling ..... 29
3.4.6.1 Auto Acknowledgement ..... 29
3.4.6.2 Auto Retransmission (ART) ..... 29
3.4.7 Enhanced ShockBurst™ flowcharts ..... 31

3.4.7.1 PTX operation ..... 31
3.4.7.2 PRX operation ..... 33
3.4.8 MultiCeiver™ ..... 34
3.4.9 Enhanced ShockBurst™ timing ..... 36
3.4.10 Enhanced ShockBurst™ transaction diagram ..... 39
3.4.10.1 Single transaction with ACK packet and interrupts ..... 40
3.4.10.2 Single transaction with a lost packet ..... 40
3.4.10.3 Single transaction with a lost ACK packet ..... 41
3.4.10.4 Single transaction with ACK payload packet ..... 41
3.4.10.5 Single transaction with ACK payload packet and lost packet ..... 42
3.4.10.6 Two transactions with ACK payload packet and the first ACK packet lost ..... 42
3.4.10.7 Two transactions where max retransmissions is reached ..... 43
3.4.11 Compatibility with ShockBurst™ ..... 43
3.4.11.1 ShockBurst™ packet format ..... 43
3.5 Data and control interface ..... 44
3.5.1 SFR registers ..... 44
3.5.2 SPI operation ..... 45
3.5.2.1 SPI commands ..... 45
3.5.3 Data FIFO ..... 47
3.5.4 Interrupt ..... 48
3.6 Register map ..... 49
3.6.1 Register map table ..... 49
**4 MCU ..... 55**
4.1 Block diagram ..... 56
4.2 Features ..... 56
4.3 Functional description ..... 57
4.3.1 Arithmetic Logic Unit (ALU) ..... 57
4.3.2 Instruction set summary ..... 57
4.3.3 Opcode map ..... 61
**5 Memory and I/O organization ..... 63**
5.1 PDATA memory addressing ..... 64
5.2 MCU Special Function Registers ..... 64
5.2.1 Accumulator - ACC ..... 64
5.2.2 B Register – B ..... 64
5.2.3 Program Status Word Register - PSW ..... 65
5.2.4 Stack Pointer – SP ..... 65
5.2.5 Data Pointer – DPH, DPL ..... 65
5.2.6 Data Pointer 1 – DPH1, DPL1 ..... 66
5.2.7 Data Pointer Select Register – DPS ..... 66
5.2.8 PCON register ..... 66
5.2.9 Special Function Register Map ..... 67
5.2.10 Special Function Registers reset values ..... 68
**6 OTP memory ..... 71**
6.1 Features ..... 71
6.2 Block diagram ..... 71

6.3 Functional description ..... 71
6.3.1 Using the NV data memory ..... 72
6.3.2 OTP memory configuration ..... 72
6.3.2.1 InfoPage content ..... 72
6.3.2.2 Memory configuration SFR ..... 73
6.3.3 Brown-out ..... 73
6.3.4 Temperature restriction for OTP programming ..... 74
6.3.5 OTP programming from the MCU ..... 74
6.3.5.1 MCU operations in the main block ..... 74
6.3.6 OTP programming through SPI ..... 74
6.3.6.1 PROG pin requirements ..... 75
6.3.6.2 SPI slave interface ..... 75
**7 Random Access memory (RAM) ..... 79**
7.1 SRAM configuration ..... 79
**8 Timers/counters ..... 81**
8.1 Features ..... 81
8.2 Block diagram ..... 81
8.3 Functional description ..... 82
8.3.1 Timer 0 and Timer 1 ..... 82
8.3.1.1 Mode 0 and Mode 1 ..... 82
8.3.1.2 Mode 2 ..... 83
8.3.1.3 Mode 3 ..... 84
8.3.2 Timer 2 ..... 84
8.3.2.1 Timer 2 description ..... 85
8.3.2.2 Timer mode ..... 85
8.3.2.3 Event counter mode ..... 85
8.3.2.4 Gated timer mode ..... 85
8.3.2.5 Timer 2 reload ..... 86
8.4 SFR registers ..... 86
8.4.1 Timer/Counter control register – TCON ..... 86
8.4.2 Timer mode register - TMOD ..... 87
8.4.3 Timer 0 – TH0, TL0 ..... 87
8.4.4 Timer 1 – TH1, TL1 ..... 87
8.4.5 Timer 2 control register – T2CON ..... 88
8.4.6 Timer 2 – TH2, TL2 ..... 88
8.4.7 Compare/Capture enable register – CCEN ..... 89
8.4.8 Capture registers – CC1, CC2, CC3 ..... 89
8.4.9 Compare/Reload/Capture register – CRCH, CRCL ..... 90
8.5 Real Time Clock - RTC ..... 90
8.5.1 Features ..... 90
8.5.2 Functional description of SFR registers ..... 90
**9 Interrupts ..... 94**
9.1 Features ..... 94
9.2 Block diagram ..... 94
9.3 Functional description ..... 95
9.4 SFR registers ..... 95

9.4.1 Interrupt Enable 0 Register – IEN0 ..... 96
9.4.2 Interrupt Enable 1 Register – IEN1 ..... 96
9.4.3 Interrupt Priority Registers – IP0, IP1 ..... 96
9.4.4 Interrupt Request Control Registers – IRCON ..... 97
**10 Watchdog ..... 98**
10.1 Features ..... 98
10.2 Block diagram ..... 98
10.3 Functional description ..... 98
**11 Power and clock management ..... 100**
11.1 Block diagram ..... 100
11.2 Modes of operation ..... 100
11.3 Functional description ..... 105
11.3.1 Clock control ..... 105
11.3.2 Power down control – PWRDWN ..... 107
11.3.3 Operational mode control - OPMCON ..... 109
11.3.4 Reset result – RSTREAS ..... 109
11.3.5 Wakeup configuration register – WUCON ..... 110
11.3.6 Pin wakeup configuration ..... 110
**12 Power supply supervisor ..... 112**
12.1 Features ..... 112
12.2 Block diagram ..... 112
12.3 Functional description ..... 112
12.3.1 Power-on reset ..... 112
12.3.2 Brown-out reset ..... 113
12.3.3 Power-fail comparator ..... 113
12.4 SFR registers ..... 114
**13 On-chip oscillators ..... 115**
13.1 Features ..... 115
13.2 Block diagrams ..... 115
13.3 Functional description ..... 116
13.3.1 16 MHz crystal oscillator ..... 116
13.3.2 16 MHz RC oscillator ..... 117
13.3.3 External 16 MHz clock ..... 117
13.3.4 32.768 kHz crystal oscillator ..... 117
13.3.5 32.768 kHz RC oscillator ..... 118
13.3.6 Synthesized 32.768 kHz clock ..... 118
13.3.7 External 32.768 kHz clock ..... 118
**14 MDU – Multiply Divide Unit ..... 119**
14.1 Features ..... 119
14.2 Block diagram ..... 119
14.3 Functional description ..... 119
14.4 SFR registers ..... 119
14.4.1 Loading the MDx registers ..... 120
14.4.2 Executing calculation ..... 121
14.4.3 Reading the result from the MDx registers ..... 121
14.4.4 Normalizing ..... 121

14.4.5 Shifting ..... 121
14.4.6 The mdef flag ..... 121
14.4.7 The mdov flag ..... 122
**15 Encryption/decryption accelerator ..... 123**
15.1 Features ..... 123
15.2 Block diagram ..... 123
15.3 Functional description ..... 123
**16 Random number generator ..... 125**
16.1 Features ..... 125
16.2 Block diagram ..... 125
16.3 Functional description ..... 125
16.4 SFR registers ..... 126
**17 General purpose IO port and pin assignments ..... 127**
17.1 Block diagram ..... 127
17.2 Functional description ..... 128
17.2.1 General purpose IO pin functionality ..... 128
17.2.2 PortCrossbar functionality ..... 129
17.2.2.1 Dynamic allocation of pins ..... 129
17.2.2.2 Dynamic pin allocation for digital blocks ..... 129
17.2.2.3 Dynamic pin allocation for analog blocks ..... 129
17.2.2.4 Default pin allocation ..... 129
17.3 IO pin maps ..... 130
17.3.1 Pin assignments in package 24 pin 4×4 mm ..... 131
17.3.2 Pin assignments in package 32 pin 5×5 mm ..... 132
17.3.3 Pin assignments in package 48 pin 7×7 mm ..... 133
17.3.4 Programmable registers ..... 135
**18 SPI ..... 143**
18.1 Features ..... 143
18.2 Block diagram ..... 143
18.3 Functional description ..... 144
18.3.1 SPI master ..... 144
18.3.2 SPI slave ..... 146
18.3.3 Slave SPI timing ..... 147
**19 Serial port (UART) ..... 150**
19.1 Features ..... 150
19.2 Block diagram ..... 150
19.3 Functional description ..... 150
19.3.1 Serial port 0 control register – S0CON ..... 151
19.3.2 Serial port 0 data buffer – S0BUF ..... 152
19.3.3 Serial port 0 reload register – S0RELH, S0RELL ..... 152
19.3.4 Serial port 0 baud rate select register - ADCON ..... 153
**20 2-Wire ..... 154**
20.1 Features ..... 154
20.2 Functional description ..... 154
20.2.1 Recommended use ..... 154
20.2.2 Master transmitter/receiver ..... 154

20.2.2.1 TX mode ..... 154
20.2.2.2 RX mode ..... 155
20.2.3 Slave transmitter/receiver ..... 155
20.2.3.1 2-Wire timing ..... 156
20.3 SFR registers ..... 157
**21 ADC ..... 160**
21.1 Features ..... 160
21.2 Block diagram ..... 160
21.3 Functional description ..... 160
21.3.1 Activation ..... 160
21.3.2 Input selection ..... 161
21.3.3 Reference selection ..... 161
21.3.4 Resolution ..... 161
21.3.5 Conversion modes ..... 161
21.3.6 Output data coding ..... 162
21.3.7 Driving the analog input ..... 163
21.3.8 SFR registers ..... 164
**22 Analog comparator ..... 166**
22.1 Features ..... 166
22.2 Block diagram ..... 166
22.3 Functional description ..... 166
22.3.1 Activation ..... 166
22.3.2 Input selection ..... 166
22.3.3 Reference selection ..... 167
22.3.4 Output polarity ..... 167
22.3.5 Input voltage range ..... 167
22.3.6 Configuration examples ..... 167
22.3.7 Driving the analog input ..... 167
22.3.8 SFR registers ..... 168
**23 PWM ..... 169**
23.1 Features ..... 169
23.2 Block diagram ..... 169
23.3 Functional description ..... 169
**24 Absolute maximum ratings ..... 171**
**25 Operating condition ..... 172**
**26 Electrical specifications ..... 173**
26.1 Power consumption ..... 178
**27 Mechanical specifications ..... 180**
**28 Reference circuits ..... 182**
28.1 nRF24LE1 OTP, 7×7 mm QFN48 ..... 182
28.1.1 Schematics ..... 182
28.1.2 Layout ..... 183
28.1.3 Bill of Materials (BOM) ..... 183
28.2 nRF24LE1 OTP, 5×5 mm QFN32 ..... 184
28.2.1 Schematics ..... 184
28.2.2 Layout ..... 185

28.2.3 Bill Of Materials (BOM) ..... 185
28.3 nRF24LE1 OTP, 4×4 mm QFN24 ..... 186
28.3.1 Schematics ..... 186
28.3.2 Layout ..... 187
28.3.3 Bill Of Materials (BOM) ..... 187
**29 Ordering information ..... 188**
29.1 Package marking ..... 188
29.1.1 Abbreviations ..... 188
29.2 Product options ..... 188
29.2.1 RF silicon ..... 188
29.2.2 Development tools ..... 189
**30 Glossary ..... 190**

# 1 Introduction

The nRF24LE1 OTP is a member of the low-cost, high-performance family of intelligent 2.4 GHz RF transceivers with embedded microcontrollers. The nRF24LE1 OTP is optimized to provide a single chip solution for ULP wireless applications. The combination of processing power, memory, low power oscillators, real-time counter, AES encryption accelerator, random generator and a range of power saving modes provides an ideal platform for implementation of RF protocols. Benefits of using nRF24LE1 OTP include tighter protocol timing, security, lower power consumption and improved co-existence performance. For the application layer the nRF24LE1 OTP offers a rich set of peripherals including: SPI, 2-wire, UART, 6 to 12 bit ADC, PWM and an ultra low power analog comparator for voltage level system wake-up.

The nRF24LE1 OTP comes in different package variants:

- nRF24LE1: An ultra compact 4×4mm 24 pin QFN (7 generic I/O pins)
- nRF24LE1: A compact 5×5mm 32 pin QFN (15 generic I/O pins)
- nRF24LE1: A 7×7mm 48 pin QFN (31 generic I/O pins)

The 4×4mm 24 pin QFN is ideal for low I/O count applications where small size is key. Examples include wearable sports sensors and watches. The 5×5mm 32 pin QFN is ideal for medium I/O count applications such as wireless mouse, remote controls and toys. The 7×7mm 48 pin QFN is designed for high I/O count products like wireless keyboards.

## 1.1 Prerequisites

In order to fully understand the product specification, a good knowledge of electronic and software engineering is necessary.

## 1.2 Writing conventions

This product specification follows a set of typographic rules to make the document consistent and easy to read. The following writing conventions are used:

- Commands, bit state conditions, and register names are written in `Courier New`.
- Pin names and pin signal conditions are written in **`Courier New bold`**.
- Cross references are underlined and highlighted in blue.

# 2 Product overview

## 2.1 Features

Features of the nRF24LE1 OTP include:

- Fast 8-bit microcontroller:
  - Intel MCS 51 compliant instruction set
  - Reduced instruction cycle time, up to 12 times compared to legacy 8051
  - 32 bit multiplication – division unit
- Memory:
  - Program memory: 16 kB of OTP memory with security features
  - Data memory: 1 kB of on-chip RAM memory
  - Non-volatile data memory: 1 kB
- A number of on-chip hardware resources are available through programmable multi purpose input/ output pins (7–31 pins dependent on package variant):
  - GPIO
  - SPI master
  - SPI slave
  - 2-Wire master/ slave
  - Full duplex serial port
  - PWM
  - ADC
  - Analog comparator
  - External interrupts
  - Timer inputs
  - 32.768 kHz crystal oscillator
- High performance 2.4 GHz RF transceiver
  - True single chip GFSK transceiver
  - Enhanced ShockBurst™ link layer support in HW:
    - Packet assembly/disassembly
    - Address and CRC computation
    - Auto ACK and retransmit
  - On the air data rate 250 kbps, 1 Mbps or 2 Mbps
  - Digital interface (SPI) speed 0–8 Mbps
  - 125 RF channels operation, with 79 (2.402 GHz –2.480 GHz) channels within 2.400–2.4835 GHz
  - Short switching time enable frequency hopping
  - Fully RF compatible with nRF24LXX
  - RF compatible with nRF2401A, nRF2402, nRF24E1, nRF24E2 in 250 kbps and 1 Mbps mode
- A/D converter:
  - 6, 8, 10 or 12 bit resolution
  - 14 input channels
  - Single ended or differential input
  - Full-scale range set by internal reference, external reference or VDD
  - Single step mode with conversion time down to 3 µs
  - Continuous mode with 2, 4, 8 or 16 kbps sampling rate
  - Low current consumption; only 0.1mA at 2 ksps
  - Mode for measuring supply voltage

- Analog comparator:
  - Used as wakeup source
  - Low current consumption (0.75µA typical)
  - Differential or single-ended input
  - Single-ended threshold programmable to 25%, 50%, 75% or 100% of VDD or an arbitrary reference voltage from pin
  - 14-channel input multiplexer
  - Rail-to-rail input voltage range
  - Programmable output polarity
- Encryption/decryption accelerator
  - Utilize time and power effective AES firmware
- Random number generator:
  - Non-deterministic architecture based on thermal noise
  - No seed value required
  - Non-repeating sequence
  - Corrector algorithm ensures uniform statistical distribution
  - Data rate up to 10 kB per second
  - Operational while the processor is in standby
- System reset and power supply monitoring:
  - On-chip power-on and brown-out reset
  - Watchdog timer reset
  - Reset from pin
  - Power-fail comparator with programmable threshold and interrupt to MCU
- On-chip timers:
  - Three16-bit timers/counters operating at the system clock (sources from the 16 MHz on-chip oscillators)
  - One 16-bit timer/counter operating at the low frequency clock (32.768 kHz)
- On-chip oscillators:
  - 16 MHz crystal oscillator XOSC16M
  - 16 MHz RC-oscillator RCOSC16M
  - 32.768 kHz crystal oscillator XOSC32K
  - 32.768 kHz RC-oscillator RCOSC32K
- Power management function:
  - Low power design supporting fully static stop/ standby
  - Programmable MCU clock frequency from 125 kHz to 16 MHz
  - On chip voltage regulators supporting low power mode
  - Watchdog and wakeup functionality running in low power mode
- Complete firmware platform available:
  - Hardware abstraction layer (HAL) Functions
  - Library functions
  - Gazell Wireless protocol
  - Application examples

## 2.2 Block diagram

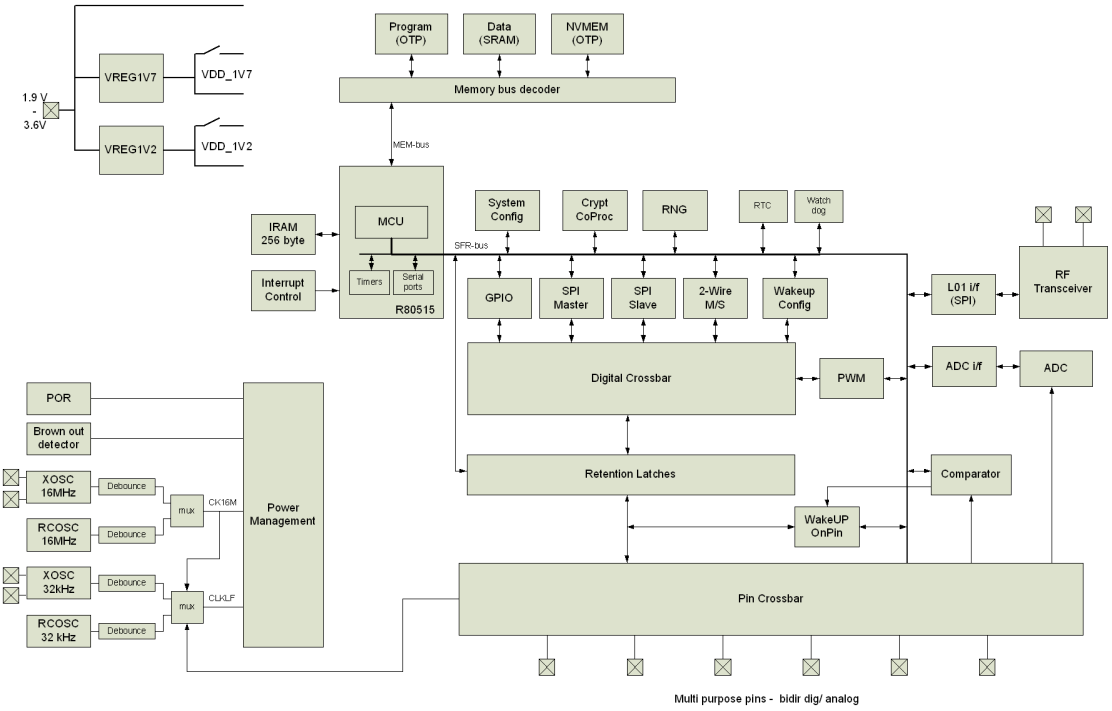



*Figure 1. nRF24LE1 OTP block diagram*

To find more information on the blocks shown in Figure 1, see Table 1. below:

| Name | Reference |
|---|---|
| Memory (Program, Data, NVMEM) | Chapter 5 on page 63 |
| Power management | Chapter 11 on page 100 |
| RF transceiver | Chapter 3 on page 17 |
| 2-Wire | Chapter 20 on page 154 |
| SPI (Master and Slave) | Chapter 18 on page 143 |
| GPIO | Chapter 17 on page 127 |
| PWM | Chapter 23 on page 169 |
| Watchdog | Chapter 10 on page 98 |

*Table 1. Block diagram cross references*

## 2.3 Pin assignments

### 2.3.1 24-pin 4×4 QFN-package variant

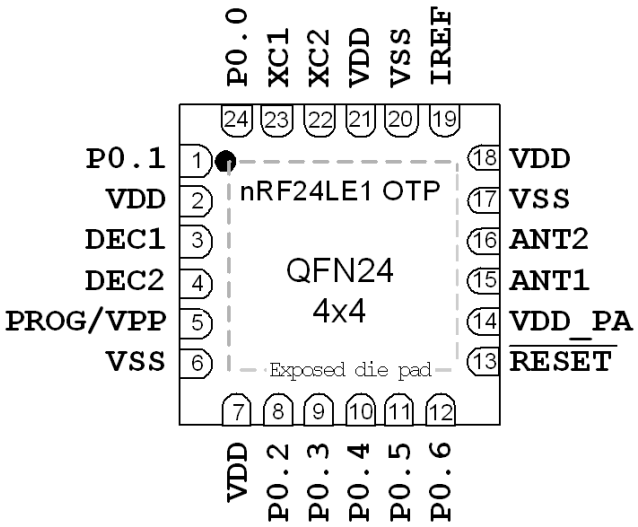



*Figure 2. nRF24LE1 OTP pin assignment (top view) for a QFN24 4×4 mm package*

### 2.3.2 32-pin 5×5 QFN-package variant

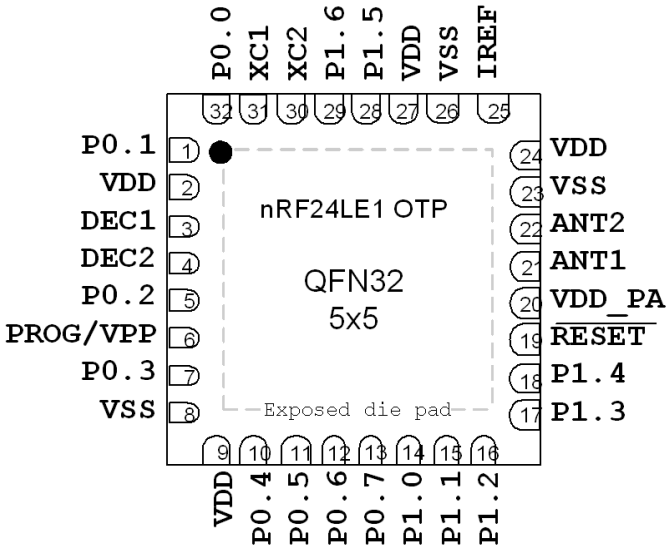



*Figure 3. nRF24LE1 OTP pin assignment (top view) for a QFN32 5×5 mm package*

### 2.3.3 48-pin 7×7 QFN-package variant

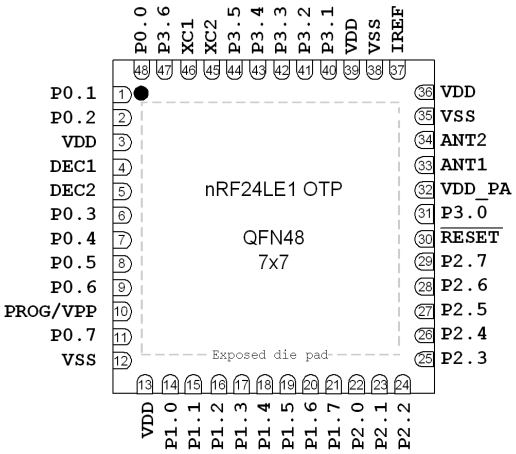



*Figure 4. nRF24LE1 OTP pin assignment (top view) for a QFN48 7×7 mm package*

## 2.4 Pin functions

| Name | Type | Description |
|---|---|---|
| **VDD** | Power | Power supply (+1.9V to +3.6V DC) |
| **VSS** | Power | Ground (0V) |
| **DEC1**<br>**DEC2** | Power | Power supply outputs for de-coupling purposes (100nF for DEC1, 33nF for DEC2) |
| **P0.0 – P3.6** | Digital or analog I/O | General purpose I/O pins. Number of I/O available depends on package type. |
| **PROG/VPP** | Digital Input,High Voltage | Input to enable OTP programming. This pin requires an external pull-down resistor, or must be connected to ground if external programming is not needed. |
| **RESET** | Digital Input | Reset for microcontroller, active low |
| **IREF** | Analog Input | Device reference current output. To be connected to reference resistor on PCB. |
| **VDD_PA** | Power Output | Power supply output (+1.8V) for on-chip RF Power amplifier |
| **ANT1**, **ANT2** | RF | Differential antenna connection (TX and RX) |
| **XC1, XC2** | Analog Input | Crystal connection for 16 MHz crystal |
| Exposed die pad | Power/heat relief | For the nRF24LE1 OTP QFN48 7×7mm and QFN32 5×5mm connect the die pad to GND. For nRF24LE1 OTP QFN24 4×4mm do not connect the die pad to GND. |

*Table 2. nRF24LE1 OTP pin functions*

# 3 RF transceiver

The nRF24LE1 OTP uses the same 2.4 GHz GFSK RF transceiver with embedded protocol engine (Enhanced ShockBurst™) that is found in the nRF24L01+ single chip RF transceiver. The RF transceiver is designed for operation in the world wide ISM frequency band at 2.400–2.4835 GHz and is very well suited for ultra low power wireless applications.

The RF transceiver module is configured and operated through the RF transceiver map. This register map is accessed by the MCU through a dedicated on-chip Serial Peripheral interface (SPI) and is available in all power modes of the RF transceiver module.

The embedded protocol engine (Enhanced ShockBurst™) enables data packet communication and supports various modes from manual operation to advanced autonomous protocol operation. Data FIFOs in the RF transceiver module ensure a smooth data flow between the RF transceiver module and the nRF24LE1 OTP MCU.

The rest of this chapter is written in the context of the RF transceiver module as the core and the rest of the nRF24LE1 OTP as external circuitry to this module.

## 3.1 Features

Features of the RF transceiver include:

- General
  - Worldwide 2.4 GHz ISM band operation
  - Common antenna interface in transmit and receive
  - GFSK modulation
  - 250 kbps, 1 and 2 Mbps on air data rate
- Transmitter
  - Programmable output power: 0, -6, -12 or -18dBm
  - 11.1mA at 0dBm output power
- Receiver
  - Integrated channel filters
  - 13.3mA at 2 Mbps
  - -82 dBm sensitivity at 2 Mbps
  - -85 dBm sensitivity at 1 Mbps
  - -94 dBm sensitivity at 250 kbps
- RF Synthesizer
  - Fully integrated synthesizer
  - 1 MHz frequency programming resolution
  - Accepts low cost ±60ppm 16 MHz crystal
  - 1 MHz non-overlapping channel spacing at 1 Mbps
  - 2 MHz non-overlapping channel spacing at 2 Mbps
- Enhanced ShockBurst™
  - 1 to 32 bytes dynamic payload length
  - Automatic packet handling (assembly/disassembly)
  - Automatic packet transaction handling (auto ACK, auto retransmit)
- 6 data pipe MultiCeiver™ for 6:1 star networks

## 3.2 Block diagram

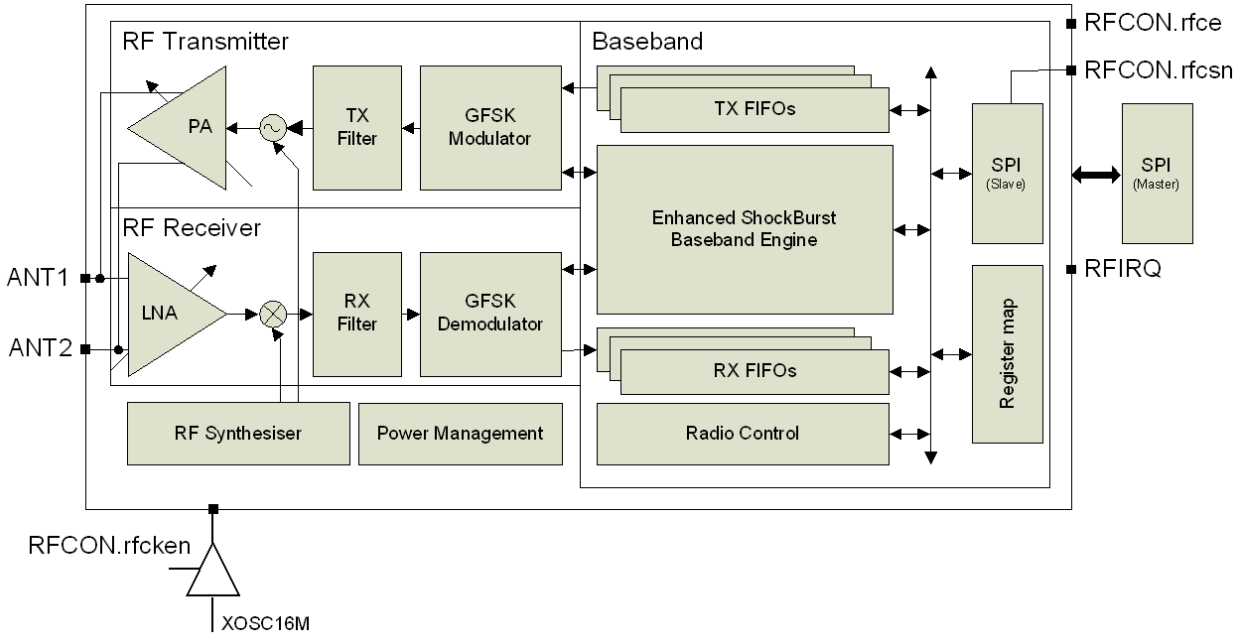



*Figure 5. RF transceiver block diagram*

## 3.3 Functional description

This section describes the different operating modes of the RF transceiver and the parameters used to control it.

The RF transceiver module has a built-in state machine that controls the transitions between the different operating modes. The state machine is controlled by SFR register `RFCON` and RF transceiver register `CONFIG`, see section 3.5 for details.

### 3.3.1 Operational Modes

You can configure the RF transceiver to power down, standby, RX and TX mode. This section describes these modes in detail.

#### 3.3.1.1 State diagram

The state diagram (Figure 6.) shows the operating modes of the RF transceiver and how they function. At the end of the reset sequence the RF transceiver enters Power Down mode. When the RF transceiver enters Power Down mode the MCU can still control the module through the SPI and the `rfcsn` bit in the `RFCON` register.

There are three types of distinct states highlighted in the state diagram:

- **Recommended operating mode:** is a recommended state used during normal operation.
- **Possible operating mode:** is a possible operating state, but is not used during normal operation.
- **Transition state:** is a time limited state used during start up of the oscillator and settling of the PLL.

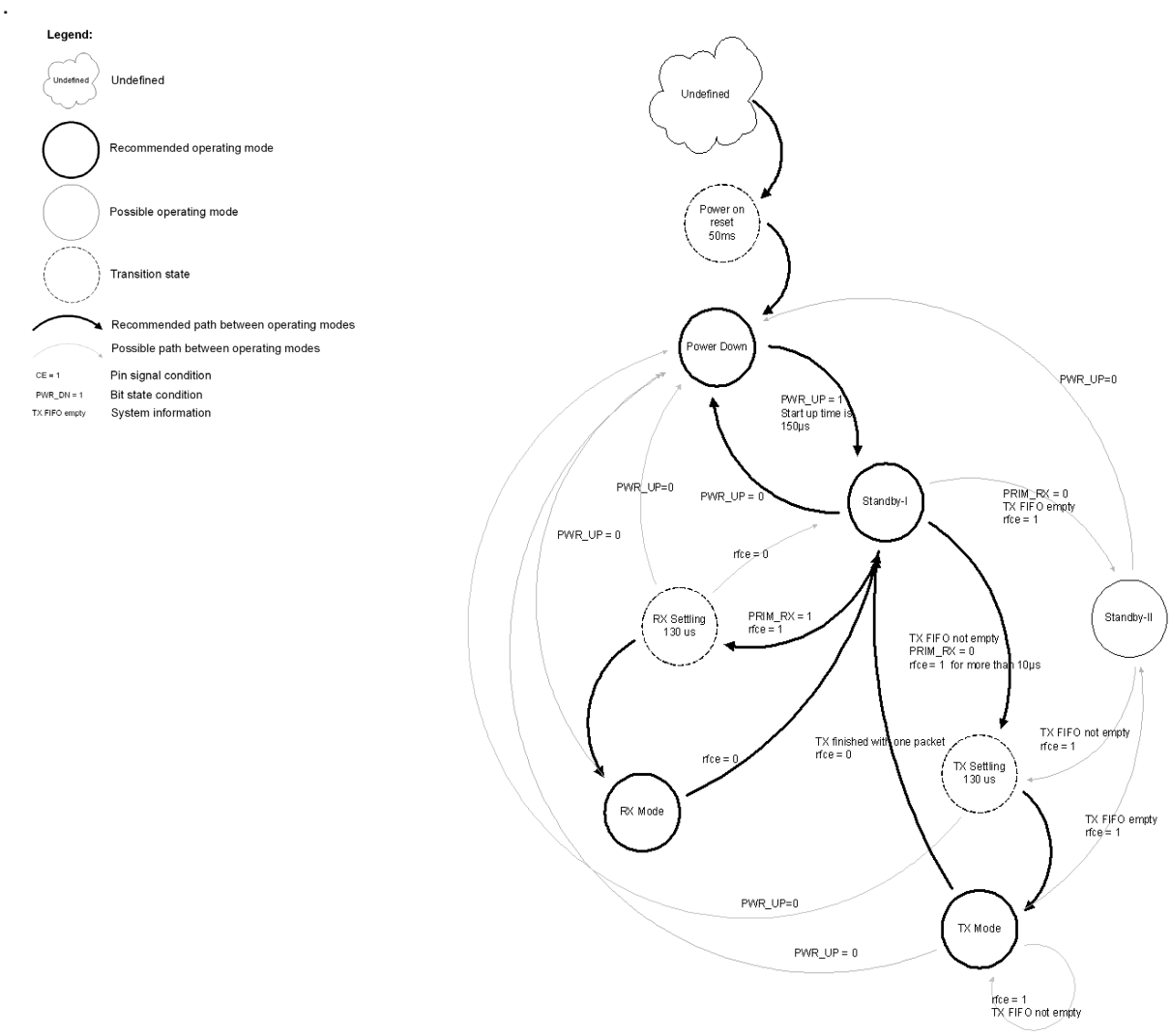



*Figure 6. Radio control state diagram*

### 3.3.1.2 Power down mode

In power down mode the RF transceiver is disabled with minimal current consumption. All the register values available from the SPI are maintained and the SPI can be activated. For start up times see Table 5. on page 23. Power down mode is entered by setting the `PWR_UP` bit in the `CONFIG` register low.

### 3.3.1.3 Standby modes

**Standby-I mode**

By setting the `PWR_UP` bit in the `CONFIG` register to 1, the RF transceiver enters standby-I mode. Standby-I mode is used to minimize average current consumption while maintaining short start up times. Change to the active mode only happens if the `rfce` bit is enabled and when it is not enabled, the RF transceiver returns to standby-I mode from both the TX and RX modes.

**Standby-II mode**

In standby-II mode extra clock buffers are active and more current is used compared to standby-I mode. The RF transceiver enters standby-II mode if the `rfce` bit is held high on a PTX operation with an empty TX FIFO. If a new packet is downloaded to the TX FIFO, the PLL immediately starts and the packet is transmitted after the normal PLL settling delay (130 µs).

The register values are maintained and the SPI can be activated during both standby modes. For start up times see Table 5. on page 23.

#### 3.3.1.4 RX mode

The RX mode is an active mode where the RF transceiver is used as a receiver. To enter this mode, the RF transceiver must have the `PWR_UP` bit, `PRIM_RX` bit and the `rfce` bit is set high.

In RX mode the receiver demodulates the signals from the RF channel, constantly presenting the demodulated data to the baseband protocol engine. The baseband protocol engine constantly searches for a valid packet. If a valid packet is found (by a matching address and a valid CRC) the payload of the packet is presented in a vacant slot in the RX FIFOs. If the RX FIFOs are full, the received packet is discarded.

The RF transceiver remains in RX mode until the MCU configures it to standby-I mode or power down mode. However, if the automatic protocol features (Enhanced ShockBurst™) in the baseband protocol engine are enabled, the RF transceiver can enter other modes in order to execute the protocol.

In RX mode a Received Power Detector (RPD) signal is available. The RPD is a signal that is set high when a RF signal higher than -64 dBm is detected inside the receiving frequency channel. The internal `RPD` signal is filtered before presented to the `RPD` register. The RF signal must be present for at least 40µs before the `RPD` is set high. How to use the RPD is described in Section 3.3.4 on page 22.

#### 3.3.1.5 TX mode

The TX mode is an active mode for transmitting packets. To enter this mode, the RF transceiver must have the `PWR_UP` bit set high, `PRIM_RX` bit set low, a payload in the TX FIFO and a high pulse on the `rfce` bit for more than 10 µs.

The RF transceiver stays in TX mode until it finishes transmitting a packet. If `rfce` = 0, RF transceiver returns to standby-I mode. If `rfce` = 1, the status of the TX FIFO determines the next action. If the TX FIFO is not empty the RF transceiver remains in TX mode and transmits the next packet. If the TX FIFO is empty the RF transceiver goes into standby-II mode. The RF transceiver transmitter PLL operates in open loop when in TX mode. It is important never to keep the RF transceiver in TX mode for more than 4 ms at a time. If the Enhanced ShockBurst™ features are enabled, RF transceiver is never in TX mode longer than 4 ms.

#### 3.3.1.6 Operational modes configuration

The following table (Table 3.) describes how to configure the operational modes.

| Mode | PWR_UP register | PRIM_RX register | rfce | FIFO state |
|---|---|---|---|---|
| RX mode | 1 | 1 | 1 | - |
| TX mode | 1 | 0 | 1 | Data in TX FIFO. Will empty all levels in TX FIFO[a]. |
| TX mode | 1 | 0 | Minimum 10 µs high pulse | Data in TX FIFO.Will empty one level in TX FIFO[b]. |
| Standby-II | 1 | 0 | 1 | TX FIFO empty |
| Standby-I | 1 | - | 0 | No ongoing packet transmission |
| Power Down | 0 | - | - | - |

a. If the rfce bit is held high the TX FIFO is emptied and all necessary ACK and possible retransmits are carried out. The transmission continues as long as the TX FIFO is refilled. If the TX FIFO is empty when the rfce bit is still high, the RF transceiver enters standby-II mode. In this mode the transmission of a packet is started as soon as the rfcsn is set high after an upload (UL) of a packet to TX FIFO.

b. This operating mode pulses the rfce bit high for at least 10 µs. This allows one packet to transmit. This is the normal operating mode. After the packet is transmitted, the RF transceiver enters standby-I mode.

*Table 3. RF transceiver main modes*

#### 3.3.1.7 Timing information

The timing information in this section relates to the transitions between modes and the timing for the rfce bit. The transition from TX mode to RX mode or vice versa is the same as the transition from the standby modes to TX mode or RX mode (130 µs), as described in Table 4.

| Name | RF transceiver | Max. | Min. | Comments |
|---|---|---|---|---|
| Tpd2stby | Power Down ➔ Standby mode | 1 µs[a] | | |
| Tstby2a | Standby modes ➔ TX/RX mode | 130 µs | | |
| Thce | Minimum rfce high | | 10 µs | |
| Tpece2csn | Delay from rfce pos. edge to rfcsn low | | 4 µs | |

a. This presupposes that the XO is running. Please refer to CLKLFCTRL for bit 3 in Table 58. on page 107.

*Table 4. Operational timing of RF transceiver*

**Note:** If **VDD** is turned off, or if the nRF24LE1 OTP enters Deep Sleep or Memory Retention mode, the register values are lost and you must configure the RF transceiver before entering the TX or RX modes.

### 3.3.2 Air data rate

The air data rate is the modulated signaling rate the RF transceiver uses when transmitting and receiving data. It can be 250 kbps, 1 Mbps or 2 Mbps. Using lower air data rate gives better receiver sensitivity than higher air data rate. But, high air data rate gives lower average current consumption and reduced probability of on-air collisions.

The air data rate is set by the `RF_DR` bit in the `RF_SETUP` register. A transmitter and a receiver must be programmed with the same air data rate to communicate with each other.

The RF transceiver is fully compatible with nRF24L01. For compatibility with nRF2401A, nRF2402, nRF24E1, and nRF24E2 the air data rate must be set to 250 kbps or 1 Mbps.

### 3.3.3 RF channel frequency

The RF channel frequency determines the center of the channel used by the RF transceiver. The channel occupies a bandwidth of less than 1 MHz at 250 kbps and 1 Mbps and a bandwidth of less than 2 MHz at 2 Mbps. The RF transceiver can operate on frequencies from 2.400 GHz to 2.525 GHz. The programming resolution of the RF channel frequency setting is 1 MHz.

At 2 Mbps the channel occupies a bandwidth wider than the resolution of the RF channel frequency setting. To ensure non-overlapping channels in 2 Mbps mode, the channel spacing must be 2 MHz or more. At 1 Mbps and 250 kbps the channel bandwidth is the same or lower than the resolution of the RF frequency.

The RF channel frequency is set by the `RF_CH` register according to the following formula:

$F_0 = 2400 +$ `RF_CH` $MHz$

You must program a transmitter and a receiver with the same RF channel frequency to communicate with each other.

### 3.3.4 Received Power Detector measurements

Received Power Detector (RPD), located in register 09, bit 0, triggers at received power levels above -64 dBm that are present in the RF channel you receive on. If the received power is less than -64 dBm, RDP = 0.

The RPD can be read out at any time while the RF transceiver is in receive mode. This offers a snapshot of the current received power level in the channel. The RPD is latched whenever a packet is received or when the MCU sets rfce low.

The status of RPD is correct when RX mode is enabled and after a wait time of Tstby2a +Tdelay_AGC= 130 µs + 40 µs. The RX gain varies over temperature which means that the RPD threshold also varies over temperature. The RPD threshold value is reduced by - 5dB at T = -40°C and increased by + 5dB at 85°C.

### 3.3.5 PA control

The PA (Power Amplifier) control is used to set the output power from the RF transceiver power amplifier. In TX mode PA control has four programmable steps, see Table 5. on page 23

The PA control is set by the RF_PWR bits in the RF_SETUP register.

| SPI RF-SETUP (RF_PWR) | RF output power | DC current consumption |
|---|---|---|
| 11 | 0 dBm | 11.1mA |
| 10 | -6 dBm | 8.8mA |
| 01 | -12 dBm | 7.3mA |
| 00 | -18 dBm | 6.8mA |

Conditions: **VDD** = 3.0V, **VSS** = 0V, $T_A$ = 27ºC, Load impedance = 15Ω+j88Ω.

*Table 5. RF output power setting for the RF transceiver*

### 3.3.6 RX/TX control

The RX/TX control is set by PRIM_RX bit in the CONFIG register and sets the RF transceiver in transmit/ receive.

## 3.4 Enhanced ShockBurst™

Enhanced ShockBurst™ is a packet based data link layer that features automatic packet assembly and timing, automatic acknowledgement and retransmissions of packets. Enhanced ShockBurst™ enables the implementation of ultra low power and high performance communication. The Enhanced ShockBurst™ features enable significant improvements of power efficiency for bi-directional and uni-directional systems, without adding complexity on the host controller side.

### 3.4.1 Features

The main features of Enhanced ShockBurst™ are:

- 1 to 32 bytes dynamic payload length
- Automatic packet handling
- Auto packet transaction handling
  - Auto Acknowledgement
  - Auto retransmit
- 6 data pipe MultiCeiver™ for 1:6 star networks

### 3.4.2 Enhanced ShockBurst™ overview

Enhanced ShockBurst™ uses ShockBurst™ for automatic packet handling and timing. During transmit, ShockBurst™ assembles the packet and clocks the bits in the data packet for transmission. During receive, ShockBurst™ constantly searches for a valid address in the demodulated signal. When ShockBurst™ finds a valid address, it processes the rest of the packet and validates it by CRC. If the packet is valid the payload is moved into a vacant slot in the RX FIFOs. All high speed bit handling and timing is controlled by ShockBurst™.

Enhanced ShockBurst™ features automatic packet transaction handling for the easy implementation of a reliable bi-directional data link. An Enhanced ShockBurst™ packet transaction is a packet exchange between two transceivers, with one transceiver acting as the Primary Receiver (PRX) and the other transceiver acting as the Primary Transmitter (PTX). An Enhanced ShockBurst™ packet transaction is always initiated by a packet transmission from the PTX, the transaction is complete when the PTX has

received an acknowledgment packet (ACK packet) from the PRX. The PRX can attach user data to the ACK packet enabling a bi-directional data link.

The automatic packet transaction handling works as follows:

1. You begin the transaction by transmitting a data packet from the PTX to the PRX. Enhanced ShockBurst™ automatically sets the PTX in receive mode to wait for the ACK packet.
2. If the packet is received by the PRX, Enhanced ShockBurst™ automatically assembles and transmits an acknowledgment packet (ACK packet) to the PTX before returning to receive mode.
3. If the PTX does not receive the ACK packet immediately, Enhanced ShockBurst™ automatically retransmits the original data packet after a programmable delay and sets the PTX in receive mode to wait for the ACK packet.

In Enhanced ShockBurst™ it is possible to configure parameters such as the maximum number of retransmits and the delay from one transmission to the next retransmission. All automatic handling is done without the involvement of the MCU.

### 3.4.3 Enhanced Shockburst™ packet format

The format of the Enhanced ShockBurst™ packet is described in this section. The Enhanced ShockBurst™ packet contains a preamble field, address field, packet control field, payload field and a CRC field. Figure 7. shows the packet format with MSB to the left.

*Figure 7. An Enhanced ShockBurst™ packet with payload (0-32 bytes)*

#### 3.4.3.1 Preamble

The preamble is a bit sequence used to synchronize the receivers demodulator to the incoming bit stream. The preamble is one byte long and is either 01010101 or 10101010. If the first bit in the address is 1 the preamble is automatically set to 10101010 and if the first bit is 0 the preamble is automatically set to 01010101. This is done to ensure there are enough transitions in the preamble to stabilize the receiver.

#### 3.4.3.2 Address

This is the address for the receiver. An address ensures that the correct packet is detected by the receiver. The address field can be configured to be 3, 4 or, 5 bytes long with the **AW** register.

**Note:** Addresses where the level shifts only one time (that is, 000FFFFFFF) can often be detected in noise and can give a false detection, which may give a raised Packet-Error-Rate. Addresses as a continuation of the preamble (hi-low toggling) raises the Packet-Error-Rate.

### 3.4.3.3 Packet Control Field (PCF)

Figure 8. shows the format of the 9 bit packet control field, MSB to the left.

*Figure 8. Packet control field (PCF)*

The packet control field contains a 6 bit payload length field, a 2 bit PID (Packet Identity) field and a 1 bit `NO_ACK flag`.

**Payload length**

This 6 bit field specifies the length of the payload in bytes. The length of the payload can be from 0 to 32 bytes.

Coding: 000000 = 0 byte (only used in empty ACK packets.) 100000 = 32 byte, 100001 = Don't care.

This field is only used if the Dynamic Payload Length function is enabled.

**PID (Packet identification)**

The 2 bit PID field is used to detect if the received packet is new or retransmitted. PID prevents the PRX operation from presenting the same payload more than once to the MCU. The PID field is incremented at the TX side for each new packet received through the SPI. The PID and CRC fields (see section 3.4.3.5 on page 26) are used by the PRX operation to determine if a packet is retransmitted or new. When several data packets are lost on the link, the PID fields may become equal to the last received PID. If a packet has the same PID as the previous packet, the RF transceiver compares the CRC sums from both packets. If the CRC sums are also equal, the last received packet is considered a copy of the previously received packet and discarded.

**No Acknowledgment flag (NO_ACK)**

The Selective Auto Acknowledgement feature controls the **NO_ACK** flag.

This flag is only used when the auto acknowledgement feature is used. Setting the flag high, tells the receiver that the packet is not to be auto acknowledged.

### 3.4.3.4 Payload

The payload is the user defined content of the packet. It can be 0 to 32 bytes wide and is transmitted on-air when it is uploaded (unmodified) to the device.

Enhanced ShockBurst™ provides two alternatives for handling payload lengths; static and dynamic.

The default is static payload length. With static payload length all packets between a transmitter and a receiver have the same length. Static payload length is set by the **RX_PW_Px** registers on the receiver side. The payload length on the transmitter side is set by the number of bytes clocked into the TX_FIFO and must equal the value in the **RX_PW_Px** register on the receiver side.

Dynamic Payload Length (DPL) is an alternative to static payload length. DPL enables the transmitter to send packets with variable payload length to the receiver. This means that for a system with different payload lengths it is not necessary to scale the packet length to the longest payload.

With the DPL feature the nRF24L01+ can decode the payload length of the received packet automatically instead of using the `RX_PW_Px` registers. The MCU can read the length of the received payload by using the `R_RX_PL_WID` command.

> **Note:** Always check if the packet width reported is 32 bytes or shorter when using the `R_RX_PL_WID` command. If its width is longer than 32 bytes then the packet contains errors and must be discarded. Discard the packet by using the `Flush_RX` command.

In order to enable DPL the `EN_DPL bit in the FEATURE register must be enabled. In RX mode the DYNPD` register must be set. A PTX that transmits to a PRX with DPL enabled must have the `DPL_P0` bit in `DYNPD` set.

#### 3.4.3.5 CRC (Cyclic Redundancy Check)

The CRC is the error detection mechanism in the packet. It may either be 1 or 2 bytes and is calculated over the address, Packet Control Field and Payload.

The polynomial for 1 byte CRC is $X^8 + X^2 + X + 1$. Initial value 0xFF.

The polynomial for 2 byte CRC is $X^{16} + X^{12} + X^5 + 1$. Initial value 0xFFFF.

No packet is accepted by Enhanced ShockBurst™ if the CRC fails.

### 3.4.4 Automatic packet assembly

The automatic packet assembly assembles the preamble, address, packet control field, payload and CRC to make a complete packet before it is transmitted.

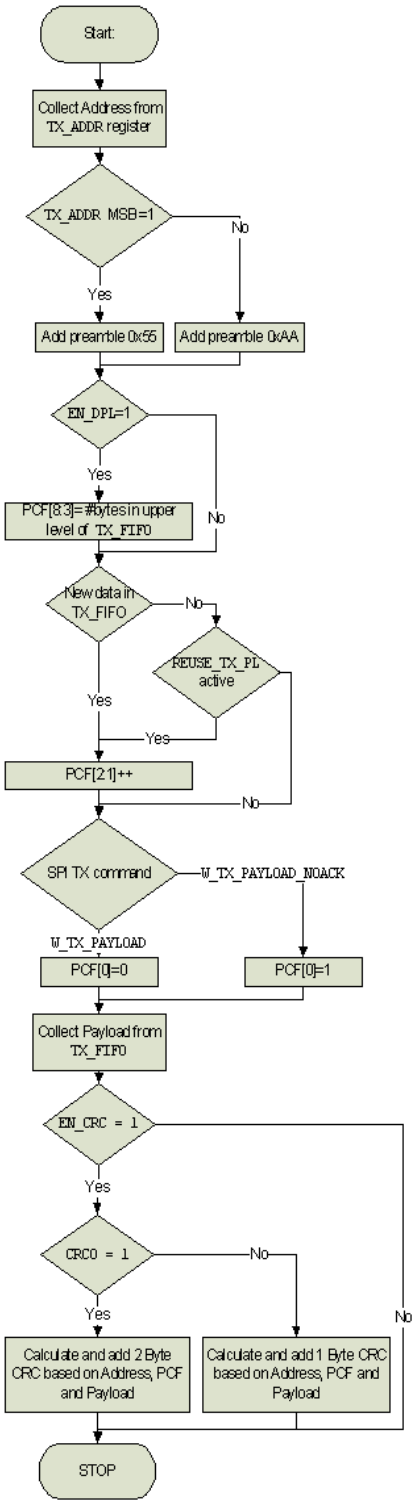



*Figure 9. Automatic packet assembly*

### 3.4.5 Automatic packet disassembly

After the packet is validated, Enhanced ShockBurst™ disassembles the packet and loads the payload into the RX FIFO, and asserts the **RX_DR** IRQ.

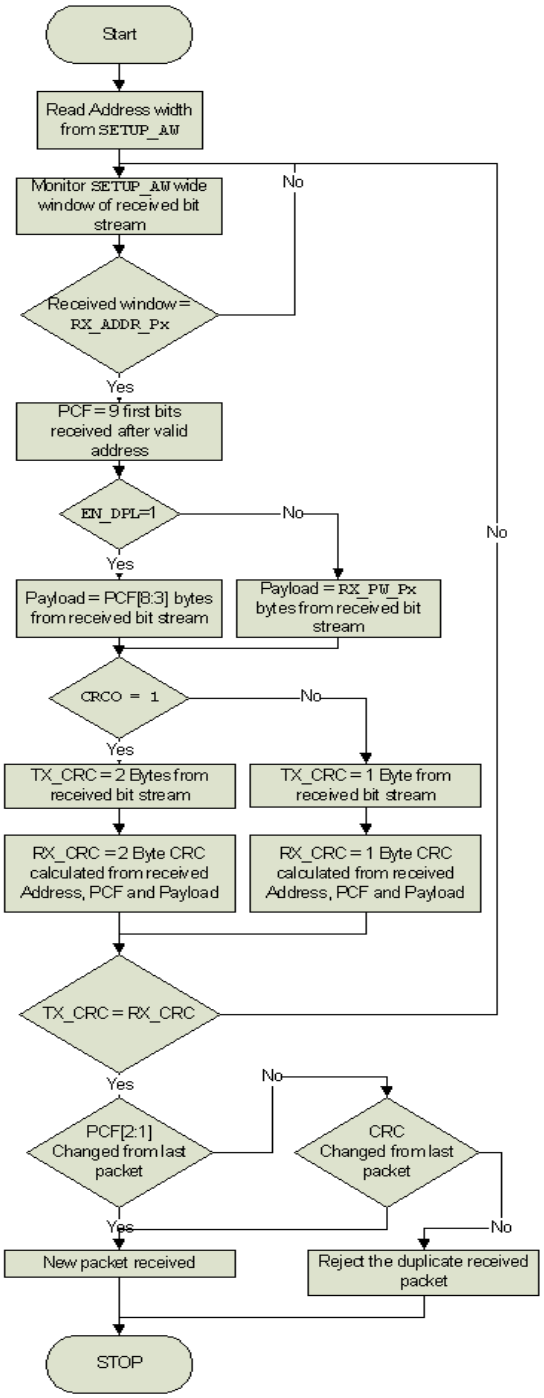



*Figure 10. Automatic packet disassembly*

### 3.4.6 Automatic packet transaction handling

Enhanced ShockBurst™ features two functions for automatic packet transaction handling; auto acknowledgement and auto re-transmit.

#### 3.4.6.1 Auto Acknowledgement

Auto Acknowledgment is a function that automatically transmits an ACK packet to the PTX after it has received and validated a packet. The Auto Acknowledgement function reduces the load of the system MCU and reduces average current consumption. The Auto Acknowledgement feature is enabled by setting the `EN_AA` register.

**Note:** If the received packet has the `NO_ACK` flag set, auto acknowledgement is not executed.

An ACK packet can contain an optional payload from PRX to PTX. In order to use this feature, the Dynamic Payload Length (DPL) feature must be enabled. The MCU on the PRX side has to upload the payload by clocking it into the TX FIFO by using the `W_ACK_PAYLOAD` command. The payload is pending in the TX FIFO (PRX) until a new packet is received from the PTX. The RF transceiver can have three ACK packet payloads pending in the TX FIFO (PRX) at the same time.

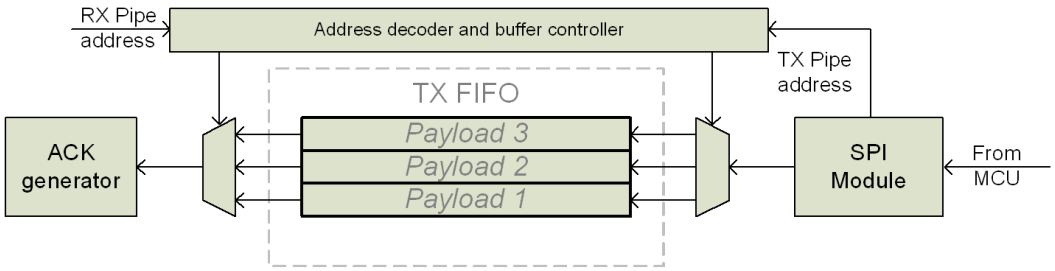



*Figure 11. TX FIFO (PRX) with pending payloads*

Figure 11. shows how the TX FIFO (PRX) is operated when handling pending ACK packet payloads. From the MCU the payload is clocked in with the `W_ACK_PAYLOAD` command. The address decoder and buffer controller ensure that the payload is stored in a vacant slot in the TX FIFO (PRX). When a packet is received, the address decoder and buffer controller are notified with the PTX address. This ensures that the right payload is presented to the ACK generator.

If the TX FIFO (PRX) contains more than one payload to a PTX, payloads are handled using the first in – first out principle. The TX FIFO (PRX) is blocked if all pending payloads are addressed to a PTX where the link is lost. In this case, the MCU can flush the TX FIFO (PRX) by using the `FLUSH_TX` command.

In order to enable Auto Acknowledgement with payload the `EN_ACK_PAY bit in the FEATURE register must be set.`

#### 3.4.6.2 Auto Retransmission (ART)

The Auto Retransmission is a function that retransmits a packet if an ACK packet is not received. It is used in an Auto Acknowledgement system on the PTX. When a packet is not acknowledged, you can set the number of times it is allowed to retransmit by setting the ARC bits in the `SETUP_RETR` register. PTX enters RX mode and waits a time period for an ACK packet each time a packet is transmitted. The amount of time the PTX is in RX mode is based on the following conditions:

- Auto Retransmit Delay (ARD) elapsed.
- No address match within 250 µs.
- After received packet (CRC correct or not) if address match within 250 µs.

The RF transceiver asserts the TX_DS IRQ when the ACK packet is received.

The RF transceiver enters standby-I mode if there is no more untransmitted data in the TX FIFO and the rfce bit in the RFCON register is low. If the ACK packet is not received, the RF transceiver goes back to TX mode after a delay defined by ARD and retransmits the data. This continues until acknowledgment is received, or the maximum number of retransmits is reached.

Two packet loss counters are incremented each time a packet is lost, ARC_CNT and PLOS_CNT in the OBSERVE_TX register. The ARC_CNT counts the number of retransmissions for the current transaction. You reset ARC_CNT by initiating a new transaction. The PLOS_CNT counts the total number of retransmissions since the last channel change. You reset PLOS_CNT by writing to the RF_CH register. It is possible to use the information in the OBSERVE_TX register to make an overall assessment of the channel quality.

The ARD defines the time from the end of a transmitted packet to when a retransmit starts on the PTX. ARD is set in **SETUP_RETR** register in steps of 250 µs. A retransmit is made if no ACK packet is received by the PTX.

There is a restriction on the length of ARD when using ACK packets with payload. The ARD time must never be shorter than the sum of the startup time and the time on-air for the ACK packet.

- For 2 Mbps data rate and 5-byte address; 15 byte is maximum ACK packet payload length for ARD=250 µs (reset value).
- For 1 Mbps data rate and 5-byte address; 5 byte is maximum ACK packet payload length for ARD=250 µs (reset value).

ARD=500 µs is long enough for any ACK payload length in 1 or 2 Mbps mode.

- For 250kbps data rate and 5-byte address the following values apply:

| ARD | ACK packet size (in bytes) |
|---|---|
| 1500µs | All ACK payload sizes |
| 1250µs | ≤ 24 |
| 1000µs | ≤ 16 |
| 750µs | ≤ 8 |
| 500µs | Empty ACK with no payload |

*Table 6. Maximum ACK payload length for different retransmit delays at 250 kbps*

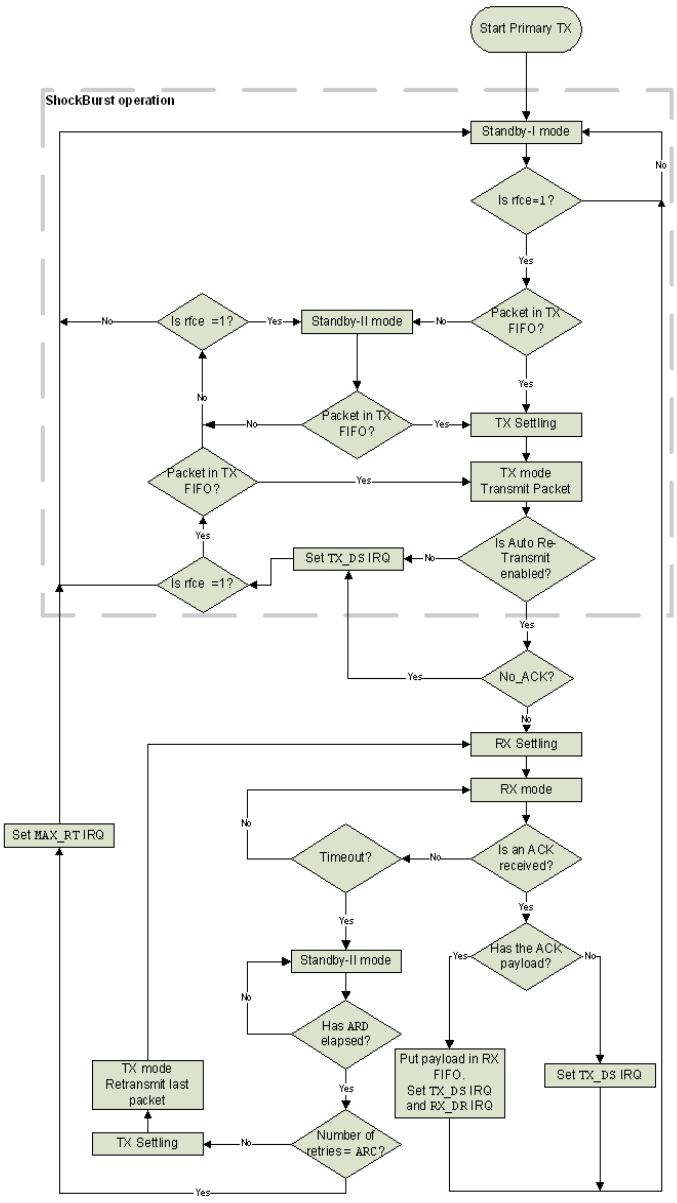
As an alternative to Auto Retransmit it is possible to manually set the RF transceiver to retransmit a packet a number of times. This is done by the REUSE_TX_PL command. The MCU must initiate each transmission of the packet with a pulse on the CE pin when this command is used.

### 3.4.7 Enhanced ShockBurst™ flowcharts

This section contains flowcharts outlining PTX and PRX operation in Enhanced ShockBurst™.

#### 3.4.7.1 PTX operation

The flowchart in Figure 12. outlines how a RF transceiver configured as a PTX behaves after entering standby-I mode.




**Note:** ShockBurst™ operation is outlined with a dashed square.

*Figure 12. PTX operations in Enhanced ShockBurst™*

Activate PTX mode by setting the rfce bit in the RFCON register high. If there is a packet present in the TX FIFO the RF transceiver enters TX mode and transmits the packet. If Auto Retransmit is enabled, the state machine checks if the NO_ACK flag is set. If it is not set, the RF transceiver enters RX mode to receive an ACK packet. If the received ACK packet is empty, only the TX_DS IRQ is asserted. If the ACK packet contains a payload, both TX_DS IRQ and RX_DR IRQ are asserted simultaneously before the RF transceiver returns to standby-I mode.

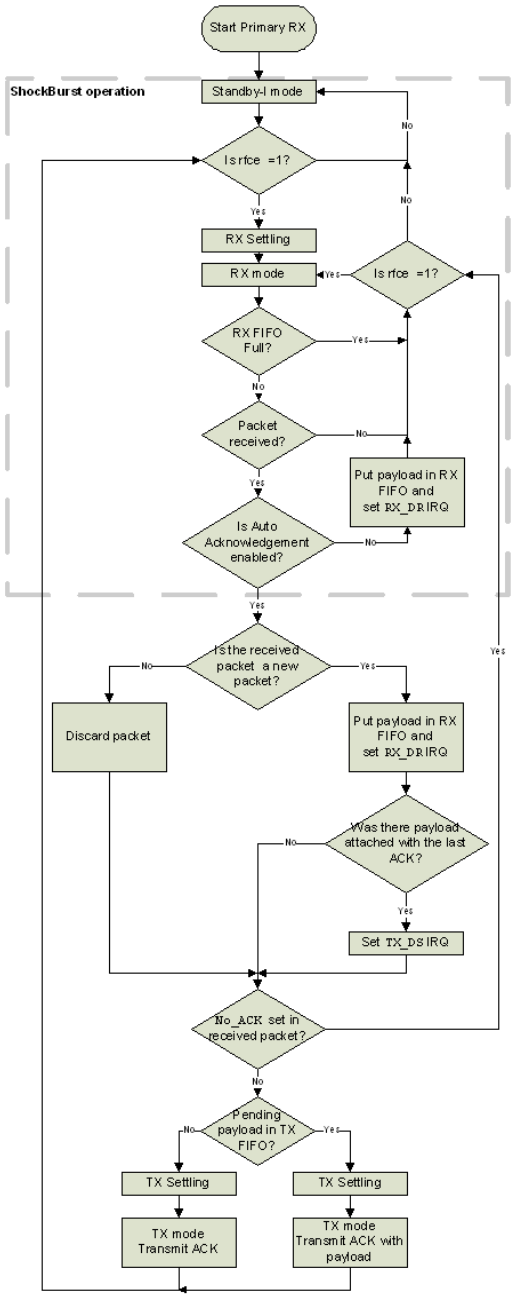
If the ACK packet is not received before timeout occurs, the RF transceiver returns to standby-II mode. It stays in standby-II mode until the ARD has elapsed. If the number of retransmits has not reached the ARC, the RF transceiver enters TX mode and transmits the last packet once more.

While executing the Auto Retransmit feature, the number of retransmits can reach the maximum number defined in ARC. If this happens, the RF transceiver asserts the MAX_RT IRQ and returns to standby-I mode.

If the rfce bit in the RFCON register is high and the TX FIFO is empty, the RF transceiver enters Standby-II mode.

### 3.4.7.2 PRX operation

The flowchart in Figure 13. outlines how a RF transceiver configured as a PRX behaves after entering standby-I mode.




**Note:** ShockBurst™ operation is outlined with a dashed square.

*Figure 13. PRX operations in Enhanced ShockBurst™*

Activate PRX mode by setting the `rfce` bit in the `RFCON` register high. The RF transceiver enters RX mode and starts searching for packets. If a packet is received and Auto Acknowledgement is enabled, the RF transceiver decides if the packet is new or a copy of a previously received packet. If the packet is new the

payload is made available in the RX FIFO and the RX_DR IRQ is asserted. If the last received packet from the transmitter is acknowledged with an ACK packet with payload, the TX_DS IRQ indicates that the PTX received the ACK packet with payload. If the No_ACK flag is not set in the received packet, the PRX enters TX mode. If there is a pending payload in the TX FIFO it is attached to the ACK packet. After the ACK packet is transmitted, the RF transceiver returns to RX mode.

A copy of a previously received packet might be received if the ACK packet is lost. In this case, the PRX discards the received packet and transmits an ACK packet before it returns to RX mode.

### 3.4.8 MultiCeiver™

MultiCeiver™ is a feature used in RX mode that contains a set of six parallel data pipes with unique addresses. A data pipe is a logical channel in the physical RF channel. Each data pipe has its own physical address (data pipe address) decoding in the RF transceiver.

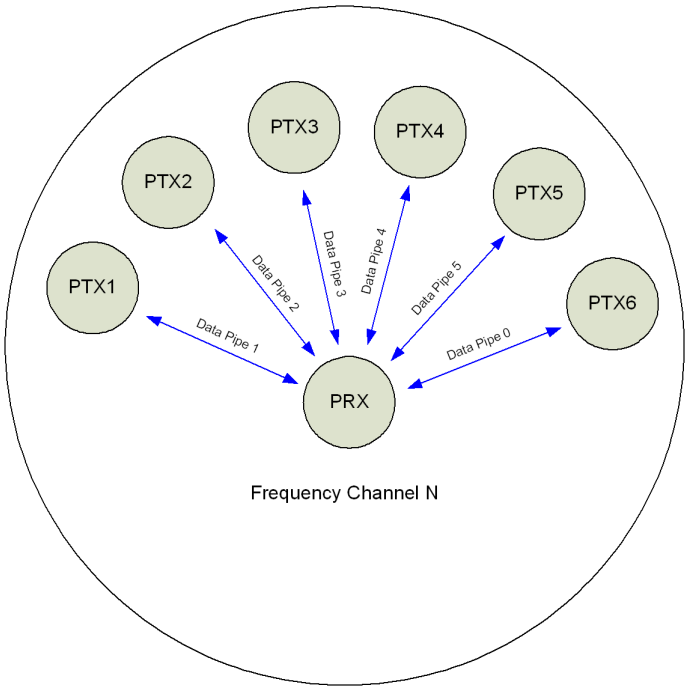



*Figure 14. PRX using MultiCeiver™*

The RF transceiver configured as PRX (primary receiver) can receive data addressed to six different data pipes in one frequency channel as shown in Figure 14. Each data pipe has its own unique address and can be configured for individual behavior.

Up to six RF transceivers configured as PTX can communicate with one RF transceiver configured as PRX. All data pipe addresses are searched for simultaneously. Only one data pipe can receive a packet at a time. All data pipes can perform Enhanced ShockBurst™ functionality.

The following settings are common to all data pipes:

- CRC enabled/disabled (CRC always enabled when Enhanced ShockBurst™ is enabled)
- CRC encoding scheme
- RX address width
- Frequency channel

- Air data rate
- LNA gain

The data pipes are enabled with the bits in the EN_RXADDR register. By default only data pipe 0 and 1 are enabled. Each data pipe address is configured in the RX_ADDR_PX registers.

**Note:** Always ensure that none of the data pipes have the same address.

Each pipe can have up to a 5 byte configurable address. Data pipe 0 has a unique 5 byte address. Data pipes 1–5 share the four most significant address bytes. The LSByte must be unique for all six pipes. Figure 15. is an example of how data pipes 0-5 are addressed.

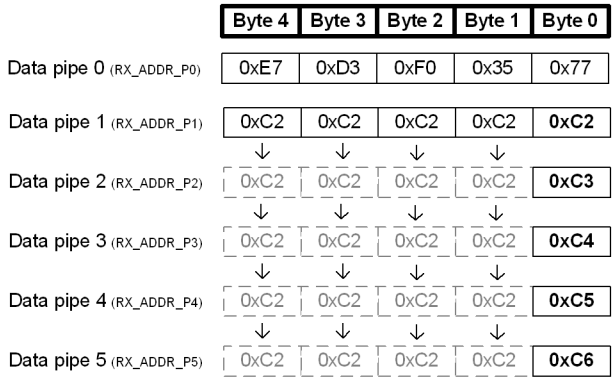



*Figure 15. Addressing data pipes 0-5*

The PRX, using MultiCeiver™ and Enhanced ShockBurst™, receives packets from more than one PTX. To ensure that the ACK packet from the PRX is transmitted to the correct PTX, the PRX takes the data pipe address where it received the packet and uses it as the TX address when transmitting the ACK packet. Figure 16. on page 36 is an example of an address configuration for the PRX and PTX. On the

PRX the RX_ADDR_Pn, defined as the pipe address, must be unique. On the PTX the TX_ADDR must be the same as the RX_ADDR_P0 and as the pipe address for the designated pipe.

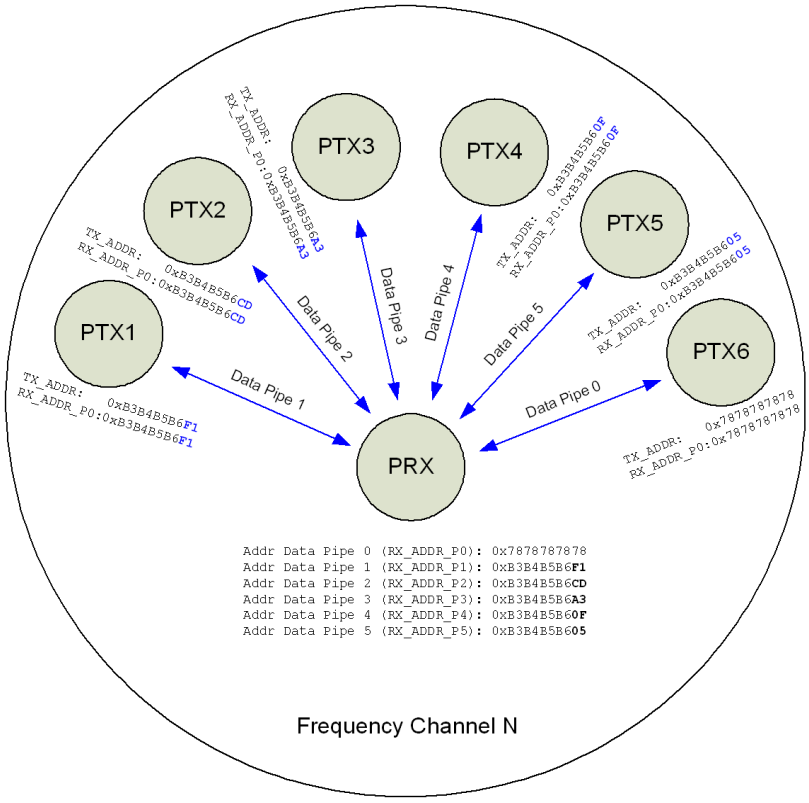


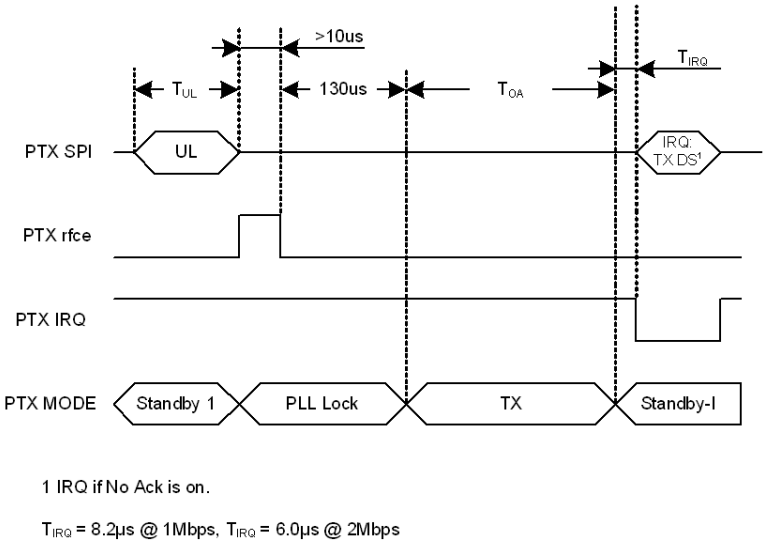

*Figure 16. Example of data pipe addressing in MultiCeiver™*

Only when a data pipe receives a complete packet can other data pipes begin to receive data. When multiple PTXs are transmitting to a PRX, the ARD can be used to skew the auto retransmission so that they only block each other once.

### 3.4.9 Enhanced ShockBurst™ timing

This section describes the timing sequence of Enhanced ShockBurst™ and how all modes are initiated and operated. The Enhanced ShockBurst™ timing is controlled through the Data and Control interface. The RF transceiver can be set to static modes or autonomous modes where the internal state machine

controls the events. Each autonomous mode/sequence ends with a `RFIRQ` interrupt. All the interrupts are indicated as IRQ events in the timing diagrams.




1 IRQ if No Ack is on.

$T_{IRQ}$ = 8.2µs @ 1Mbps, $T_{IRQ}$ = 6.0µs @ 2Mbps

*Figure 17. Transmitting one packet with* **NO_ACK** *on*

The following equations calculate various timing measurements:

| Symbol | Description | Equation |
|---|---|---|
| $T_{OA}$ | Time on-air | $T_{OA} = \frac{packet\ length}{air\ data\ rate} = \frac{8\left[\frac{bit}{byte}\right] \cdot \left(1[byte]_{preamble} + 3,4\ or\ 5[bytes]_{address} + N[bytes]_{payload} + 1\ or\ 2[bytes]_{CRC}\right) + 9[bit]_{packet\ control\ field}}{air\ data\ rate\left[\frac{bit}{s}\right]}$ |
| $T_{ACK}$ | Time on-air Ack | $T_{ACK} = \frac{packet\ length}{air\ data\ rate} = \frac{8\left[\frac{bit}{byte}\right] \cdot \left(1[byte]_{preamble} + 3,4\ or\ 5[bytes]_{address} + N[bytes]_{payload} + 1\ or\ 2[bytes]_{CRC}\right) + 9[bit]_{packet\ control\ field}}{air\ data\ rate\left[\frac{bit}{s}\right]}$ |
| $T_{UL}$ | Time Upload | $T_{UL} = \frac{payload\ length}{SPI\ data\ rate} = \frac{8\left[\frac{bit}{byte}\right] \cdot N[bytes]_{payload}}{SPI\ data\ rate\left[\frac{bit}{s}\right]}$ |
| $T_{ESB}$ | Time Enhanced Shock-Burst™ cycle | $T_{ESB} = T_{UL} + 2\ .\ T_{stby2a} + T_{OA} + T_{ACK} + T_{IRQ}$ |

*Table 7. Timing equations*

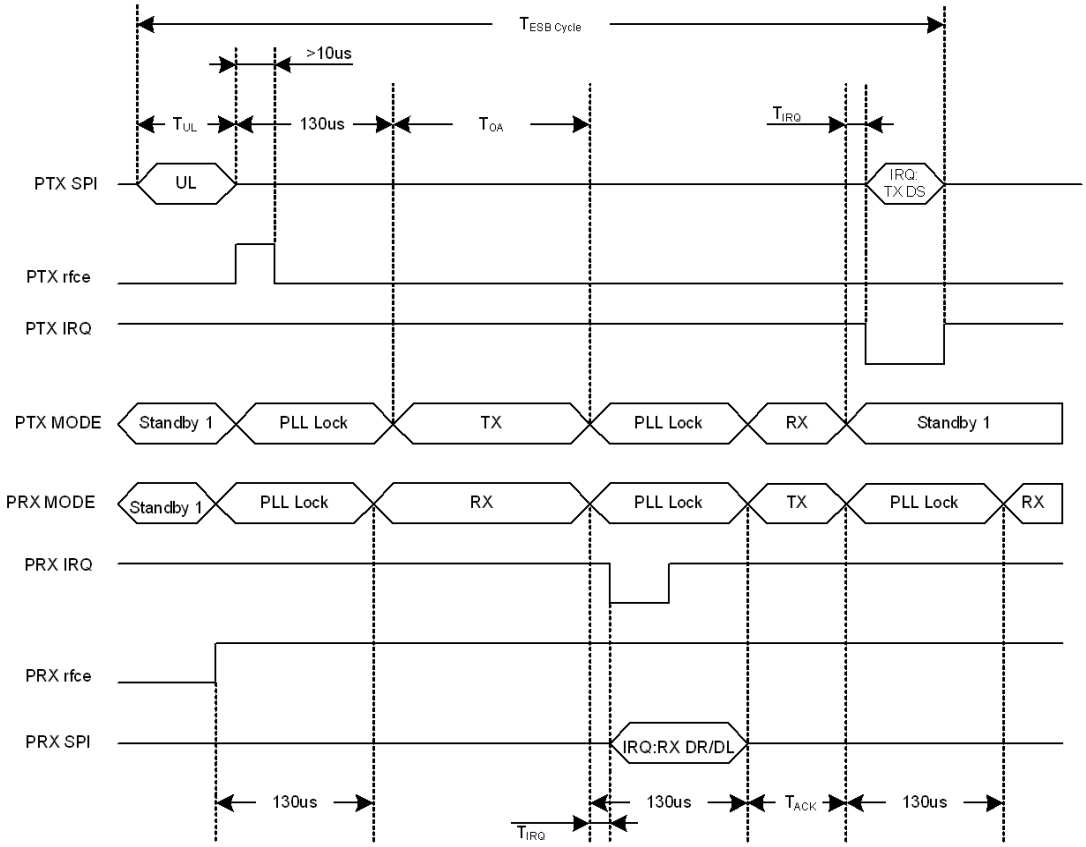



*Figure 18. Timing of Enhanced ShockBurst™ for one packet upload (2 Mbps)*

In Figure 18. the transmission and acknowledgement of a packet is shown. The PRX operation activates RX mode (`rfce`=1), and the PTX operation is activated in TX mode (`rfce`=1 for minimum 10 µs). After 130 µs the transmission starts and finishes after the elapse of $T_{OA}$.

When the transmission ends the PTX operation automatically switches to RX mode to wait for the ACK packet from the PRX operation. When the PRX operation receives the packet it sets the interrupt for the host MCU and switches to TX mode to send an ACK. After the PTX operation receives the ACK packet it sets the interrupt to the MCU and clears the packet from the TX FIFO.

In Figure 19. the PTX timing of a packet transmission is shown when the first ACK packet is lost. To see the complete transmission when the ACK packet fails see Figure 22. on page 41.

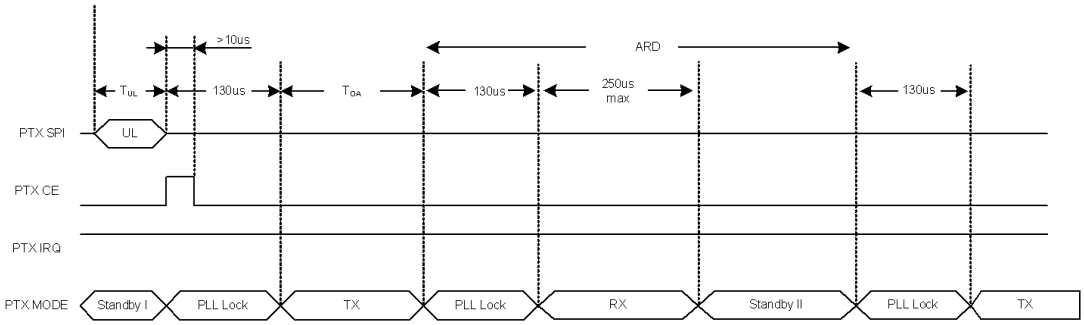



*Figure 19. Timing of Enhanced ShockBurst™ when the first ACK packet is lost (2 Mbps)*

### 3.4.10 Enhanced ShockBurst™ transaction diagram

This section describes several scenarios for the Enhanced ShockBurst™ automatic transaction handling. The call outs in this section's figures indicate the IRQs and other events. For MCU activity the event may be placed at a different timeframe.

**Note:** The figures in this section indicate the earliest possible download (DL) of the packet to the MCU and the latest possible upload (UL) of payload to the transmitter.

### 3.4.10.1 Single transaction with ACK packet and interrupts

In Figure 20. the basic auto acknowledgement is shown. After the packet is transmitted by the PTX and received by the PRX the ACK packet is transmitted from the PRX to the PTX. The `RX_DR IRQ` is asserted after the packet is received by the PRX, whereas the `TX_DS IRQ` is asserted when the packet is acknowledged and the ACK packet is received by the PTX.

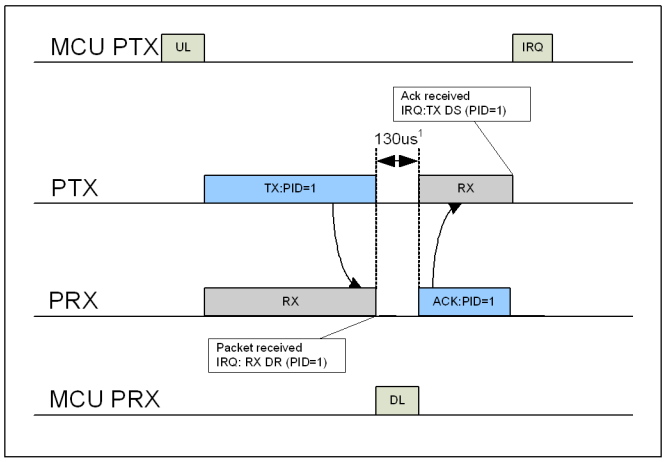



*Figure 20. TX/RX cycles with ACK and the according interrupts*

### 3.4.10.2 Single transaction with a lost packet

Figure 21. is a scenario where a retransmission is needed due to loss of the first packet transmit. After the packet is transmitted, the PTX enters RX mode to receive the ACK packet. After the first transmission, the PTX waits a specified time for the ACK packet, if it is not in the specific time slot the PTX retransmits the packet as shown in Figure 21.

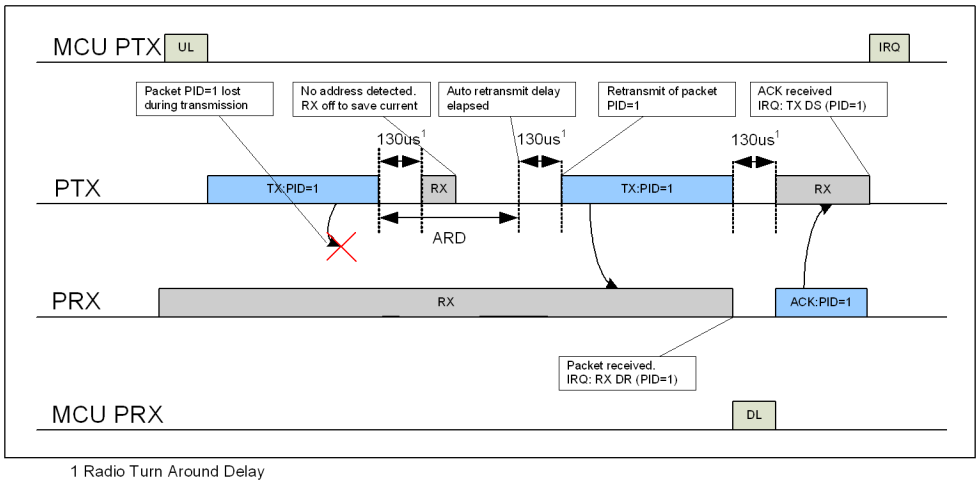



*Figure 21. TX/RX cycles with ACK and the according interrupts when the first packet transmit fails*

When an address is detected the PTX stays in RX mode until the packet is received. When the retransmitted packet is received by the PRX (see Figure 21.), the RX_DR IRQ is asserted and an ACK is transmitted back to the PTX. When the ACK is received by the PTX, the TX_DS IRQ is asserted.

### 3.4.10.3 Single transaction with a lost ACK packet

Figure 22. is a scenario where a retransmission is needed after a loss of the ACK packet. The corresponding interrupts are also indicated.

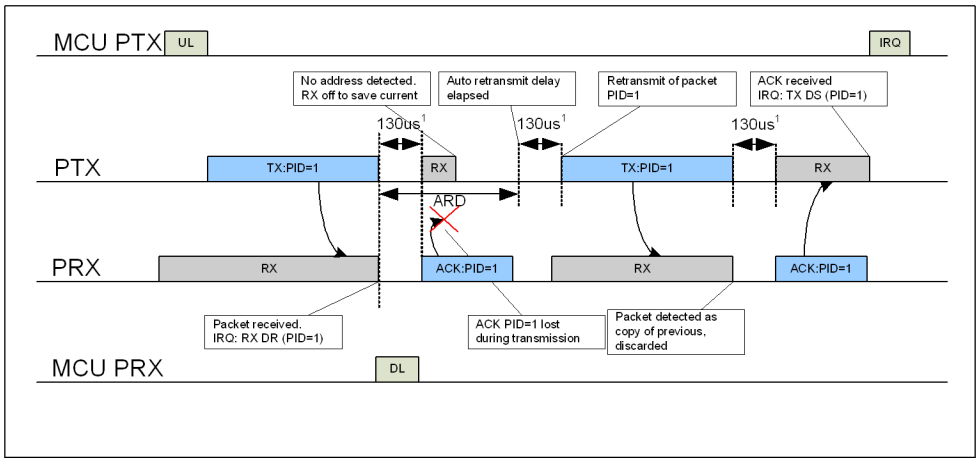



*Figure 22. TX/RX cycles with ACK and the according interrupts when the ACK packet fails*

### 3.4.10.4 Single transaction with ACK payload packet

Figure 23. is a scenario of the basic auto acknowledgement with payload. After the packet is transmitted by the PTX and received by the PRX the ACK packet with payload is transmitted from the PRX to the PTX. The RX_DR IRQ is asserted after the packet is received by the PRX, whereas on the PTX side the TX_DS IRQ is asserted when the ACK packet is received by the PTX. On the PRX side, the TX_DS IRQ for the ACK packet payload is asserted after a new packet from PTX is received. The position of the IRQ in Figure 23. shows where the MCU can respond to the interrupt.

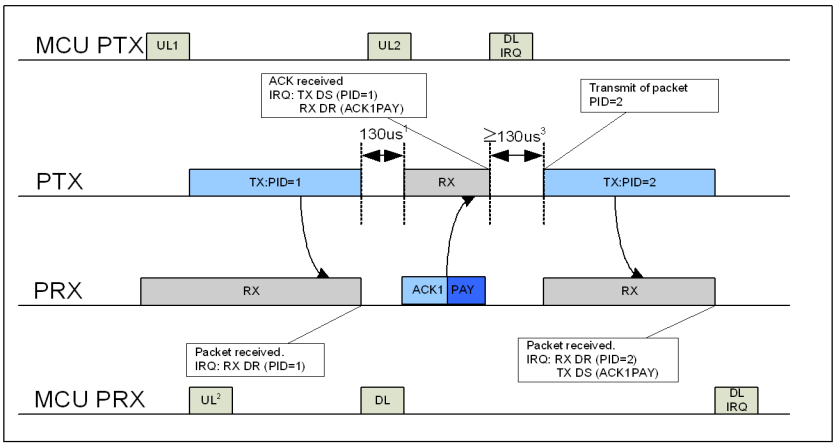
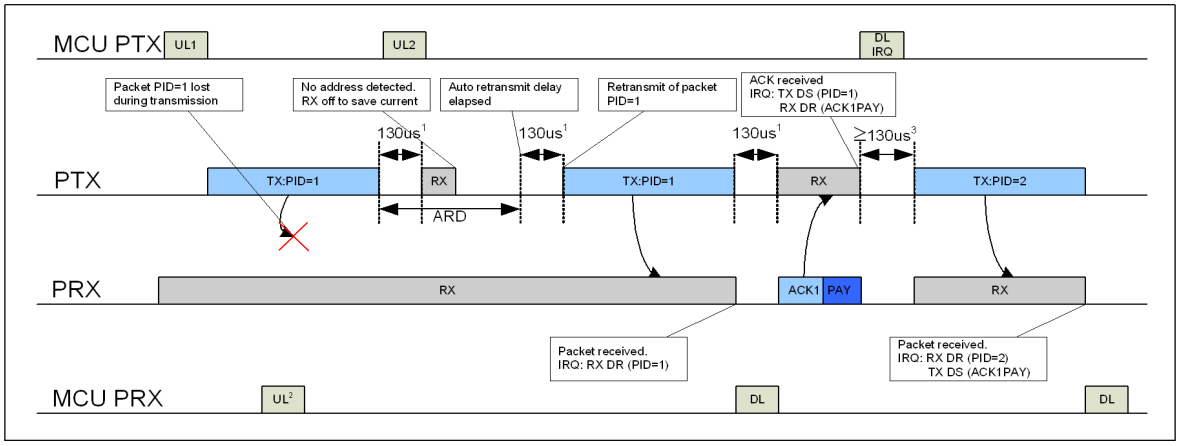



*Figure 23. TX/RX cycles with ACK Payload and the according interrupts*

### 3.4.10.5 Single transaction with ACK payload packet and lost packet

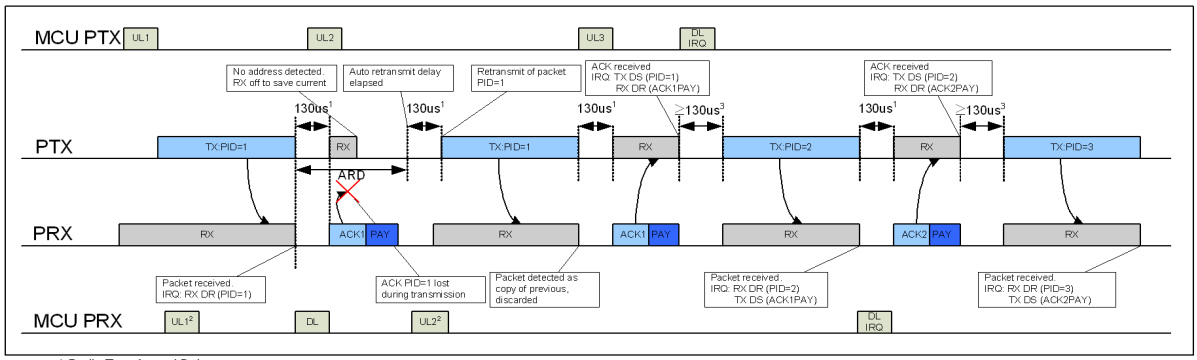
Figure 24. is a scenario where the first packet is lost and a retransmission is needed before the RX_DR IRQ on the PRX side is asserted. For the PTX both the TX_DS and RX_DR IRQ are asserted after the ACK packet is received. After the second packet (PID=2) is received on the PRX side both the RX_DR (PID=2) and TX_DS (ACK packet payload) IRQ are asserted.




1 Radio Turn Around Delay
2 Uploading Paylod for Ack Packet
3 Delay defined by MCU on PTX side, ≥ 130us

*Figure 24. TX/RX cycles and the according interrupts when the packet transmission fails*

### 3.4.10.6 Two transactions with ACK payload packet and the first ACK packet lost

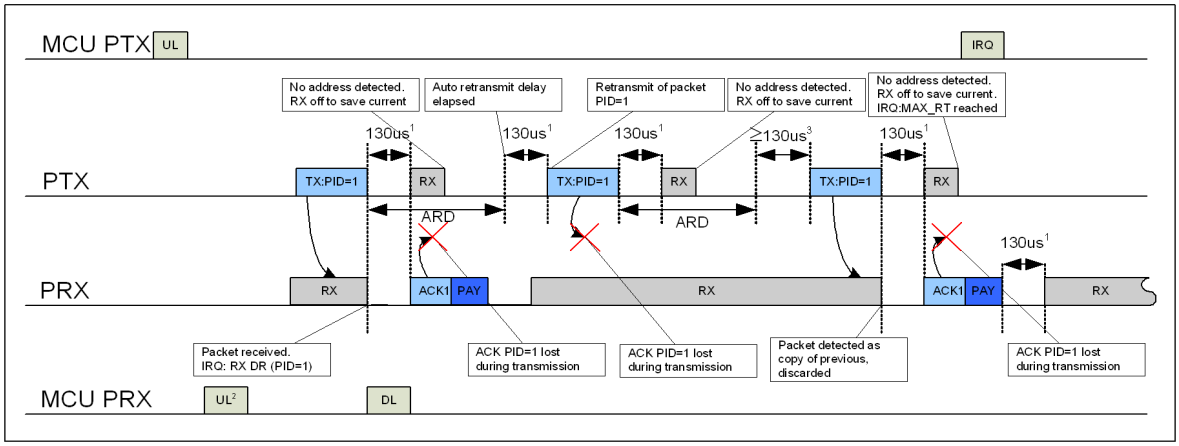



1 Radio Turn Around Delay
2 Uploading Payload for Ack Packet
3 Delay defined by MCU on PTX side, ≥ 130us

*Figure 25. TX/RX cycles with ACK Payload and the according interrupts when the ACK packet fails*

In Figure 25. the ACK packet is lost and a retransmission is needed before the TX_DS IRQ is asserted, but the RX_DR IRQ is asserted immediately. The retransmission of the packet (PID=1) results in a discarded packet. For the PTX both the TX_DS and RX_DR IRQ are asserted after the second transmission of ACK, which is received. After the second packet (PID=2) is received on the PRX both the RX_DR (PID=2) and TX_DS (ACK1PAY) IRQ is asserted. The callouts explains the different events and interrupts.

### 3.4.10.7 Two transactions where max retransmissions is reached




1 Radio Turn Around Delay
2 Uploading Paylod for Ack Packet
3 Delay defined by MCU on PTX side, ≥ 130us

*Figure 26. TX/RX cycles with ACK Payload and the according interrupts when the transmission fails. ARC is set to 2.*

`MAX_RT` IRQ is asserted if the auto retransmit counter (`ARC_CNT`) exceeds the programmed maximum limit (ARC). In Figure 26. the packet transmission ends with a `MAX_RT` IRQ. The payload in TX FIFO is NOT removed and the MCU decides the next step in the protocol. A toggle of the `rfce` bit in the `RFCON` register starts a new transmitting sequence of the same packet. The payload can be removed from the TX FIFO using the `FLUSH_TX` command.

## 3.4.11 Compatibility with ShockBurst™

You must disable Enhanced ShockBurst™ for backward compatibility with the nRF2401A, nRF2402, nRF24E1 and, nRF24E2. Set the register `EN_AA` = 0x00 and `ARC` = 0 to disable Enhanced ShockBurst™. In addition, the RF transceiver air data rate must be set to 1 Mbps or 250 kbps.

### 3.4.11.1 ShockBurst™ packet format

The ShockBurst™ packet format is described in this chapter. Figure 27. shows the packet format with MSB to the left.

| Preamble 1 byte | Address 3-5 byte | Payload 1 - 32 byte | CRC 1-2 byte |
|---|---|---|---|

*Figure 27. A ShockBurst™ packet compatible with nRF2401/nRF2402/nRF24E1/nRF24E2 devices.*

The ShockBurst™ packet format has a preamble, address, payload and CRC field that are the same as the Enhanced ShockBurst™ packet format described in section 3.4.3 on page 24.

The differences between the ShockBurst™ packet and the Enhanced ShockBurst™ packet are:

- The 9 bit Packet Control Field is not present in the ShockBurst™ packet format.

- The CRC is optional in the ShockBurst™ packet format and is controlled by the EN_CRC bit in the CONFIG register.

## 3.5 Data and control interface

The data and control interface gives you access to all the features in the RF transceiver. Compared to the standalone component SFR registers are used instead of port pins. Otherwise the interface is identical to the standalone nRF24L01+ chip.

### 3.5.1 SFR registers

| Address (Hex) | Name/Mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| 0xE4 | SPIRCON0 | 6:0 | 0x01 | R/W | SPI Master configuration register 0.<br>Reserved. Do not alter. |
| 0xE5 | SPIRCON1 | 3:0 | 0x0F | R/W | SPI Master configuration register 1. |
| | *maskIrqRxFifoFull* | 3 | 1 | R/W | 1: Disable interrupt when RX FIFO is full.<br>0: Enable interrupt when RX FIFO is full. |
| | *maskIrqRxDataReady* | 2 | 1 | R/W | 1: Disable interrupt when data is available in RX FIFO.<br>0: Enable interrupt when data is available in RX FIFO. |
| | *maskIrqTxFifoEmpty* | 1 | 1 | R/W | 1: Disable interrupt when TX FIFO is empty.<br>0: Enable interrupt when TX FIFO is empty. |
| | *maskIrqTxFifoReady* | 0 | 1 | R/W | 1: Disable interrupt when a location is available in TX FIFO.<br>0: Enable interrupt when a location is available in TX FIFO. |
| 0xE6 | spiMasterStatus SPIRSTAT | 3:0 | 0x03 | R | SPI Master status register. |
| | *rxFifoFull* | 3 | 0 | R | Interrupt source.<br>1: RX FIFO full.<br>0: RX FIFO can accept more data from SPI.<br>Cleared when the cause is removed. |
| | *rxDataReady* | 2 | 0 | R | Interrupt source.<br>1: Data available in RX FIFO.<br>0: No data in RX FIFO.<br>Cleared when the cause is removed. |
| | *txFifoEmpty* | 1 | 1 | R | Interrupt source.<br>1: TX FIFO empty.<br>0: Data in TX FIFO.<br>Cleared when the cause is removed. |
| | *txFifoReady* | 0 | 1 | R | Interrupt source.<br>1: Location available in TX FIFO.<br>0: TX FIFO full.<br>Cleared when the cause is removed. |
| 0xE7 | SPIRDAT | 7:0 | 0x00 | R/W | SPI Master data register.<br>Accesses TX (write) and RX (read) FIFO buffers, both two bytes deep. |

*Table 8. RF transceiver SPI master registers*

The RF transceiver SPI Master is configured through SPIRCON1. Four different sources can generate interrupt, unless they are masked by their respective bits in SPIRCON1. SPIRSTAT reveals which sources that are active.

SPIRDAT accesses both the TX (write) and the RX (read) FIFOs, which are two bytes deep. The FIFOs are dynamic and can be refilled according to the state of the status flags: “FIFO ready” means that the FIFO can accept data. “Data ready” means that the FIFO can provide data, minimum one byte.

| Addr | Bit | Name | R/W | Function |
|---|---|---|---|---|
| 0xE8 | 7:3 | - | | Reserved |
| | 2 | rfcken | RW | RF Clock Enable (16 MHz) |
| | 1 | rfcsn | RW | Enable RF command. 0: enabled |
| | 0 | rfce | RW | Enable RF transceiver. 1: enabled |

*Table 9. RFCON register*

`RFCON` controls the RF transceiver SPI Slave chip select signal (CSN), the RF transceiver chip enable signal (CE) and the RF transceiver clock enable signal (CKEN).

### 3.5.2 SPI operation

This section describes the SPI commands and timing.

#### 3.5.2.1 SPI commands

The SPI commands are shown in Table 10. on page 46 Every new command must be started by writing 0 to rfcsn in the `RFCON` register.

The SPI command is transferred to RF transceiver by writing the command to the `SPIRDAT` register. After the first transfer the RF transceiver's `STATUS` register can be read from `SPIRDAT` when the transfer is completed.

The serial shifting SPI commands is in the following format:

<**Command word**: MSBit to LSBit (one byte)>

<**Data bytes**: LSByte to MSByte, MSBit in each byte first>

.

| Command name | Command word (binary) | # Data bytes | Operation |
|---|---|---|---|
| R_REGISTER | 000A AAAA | 1 to 5 LSByte first | Read command and status registers. AAAAA = 5 bit Register Map Address |
| W_REGISTER | 001A AAAA | 1 to 5 LSByte first | Write command and status registers. AAAAA = 5 bit Register Map Address<br>Executable in power down or standby modes only. |
| R_RX_PAYLOAD | 0110 0001 | 1 to 32 LSByte first | Read RX-payload: 1 – 32 bytes. A read operation always starts at byte 0. Payload is deleted from FIFO after it is read. Used in RX mode. |
| W_TX_PAYLOAD | 1010 0000 | 1 to 32 LSByte first | Write TX-payload: 1 – 32 bytes. A write operation always starts at byte 0 used in TX payload. |
| FLUSH_TX | 1110 0001 | 0 | Flush TX FIFO, used in TX mode |
| FLUSH_RX | 1110 0010 | 0 | Flush RX FIFO, used in RX mode<br>Should not be executed during transmission of acknowledge, that is, acknowledge package will not be completed. |
| REUSE_TX_PL | 1110 0011 | 0 | Used for a PTX operation<br>Reuse last transmitted payload.<br>TX payload reuse is active until W_TX_PAYLOAD or FLUSH TX is executed. TX payload reuse must not be activated or deactivated during package transmission. |
| R_RX_PL_WID[a] | 0110 0000 | 1 | Read RX payload width for the top R_RX_PAYLOAD in the RX FIFO.<br><br>**Note:** Flush RX FIFO if the read value is larger than 32 bytes. |
| **W_ACK_PAYLOAD**[a] | 1010 1PPP | 1 to 32 LSByte first | Used in RX mode.<br>Write Payload to be transmitted together with ACK packet on PIPE PPP. (PPP valid in the range from 000 to 101). Maximum three ACK packet payloads can be pending. Payloads with same PPP are handled using first in - first out principle. Write payload: 1– 32 bytes. A write operation always starts at byte 0. |
| W_TX_PAYLOAD_NOACK[a] | 1011 0000 | 1 to 32 LSByte first | Used in TX mode. Disables AUTOACK on this specific packet. |
| NOP | 1111 1111 | 0 | No Operation. Might be used to read the STATUS register |

a. The bits in the FEATURE register shown in Table 11. on page 54 have to be set.

*Table 10. Command set for the RF transceiver SPI*

The W_REGISTER and R_REGISTER commands operate on single or multi-byte registers. When accessing multi-byte registers read or write to the MSBit of LSByte first. You can terminate the writing before all bytes in a multi-byte register are written, leaving the unwritten MSByte(s) unchanged. For example, the LSByte of RX_ADDR_P0 can be modified by writing only one byte to the RX_ADDR_P0 register. The content of the status register is always read to **MISO** after a high to low transition on **CSN**.

**Note:** The 3 bit pipe information in the STATUS register is updated during the RFIRQ high to low transition. The pipe information is unreliable if the STATUS register is read during an RFIRQ high to low transition.

### 3.5.3 Data FIFO

The data FIFOs store transmitted payloads (TX FIFO) or received payloads that are ready to be clocked out (RX FIFO). The FIFOs are accessible in both PTX mode and PRX mode.

The following FIFOs are present in the RF transceiver:

- TX three level, 32 byte FIFO
- RX three level, 32 byte FIFO

Both FIFOs have a controller and are accessible through the SPI by using dedicated SPI commands. A TX FIFO in PRX can store payloads for ACK packets to three different PTX operations. If the TX FIFO contains more than one payload to a pipe, payloads are handled using the first in - first out principle. The TX FIFO in a PRX is blocked if all pending payloads are addressed to pipes where the link to the PTX is lost. In this case, the MCU can flush the TX FIFO using the FLUSH_TX command.

The RX FIFO in PRX can contain payloads from up to three different PTX operations and a TX FIFO in PTX can have up to three payloads stored.

You can write to the TX FIFO using these three commands; W_TX_PAYLOAD and **W_TX_PAYLOAD_NO_ACK** in PTX mode and W_ACK_PAYLOAD in PRX mode. All three commands provide access to the TX_PLD register.

The RX FIFO can be read by the command R_RX_PAYLOAD in PTX and PRX mode. This command provides access to the RX_PLD register.

The payload in TX FIFO in a PTX is not removed if the MAX_RT IRQ is asserted.

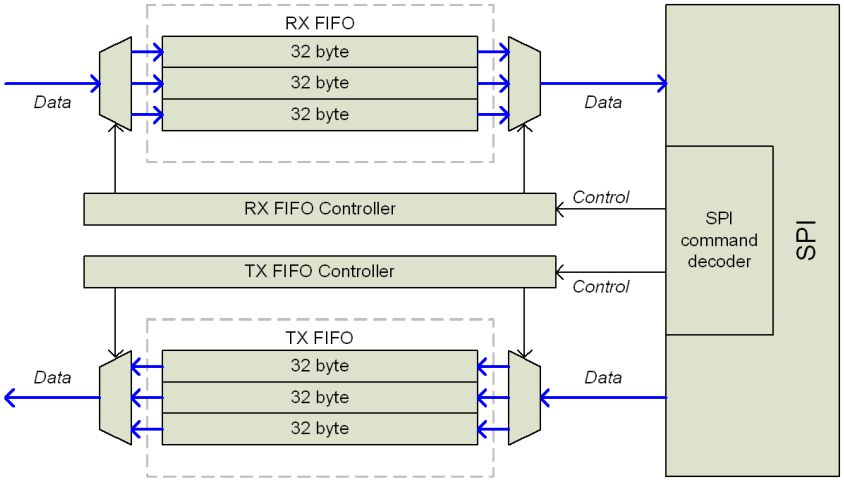



*Figure 28. FIFO (RX and TX) block diagram*

You can read if the TX and RX FIFO are full or empty in the FIFO_STATUS register. TX_REUSE (also available in the FIFO_STATUS register) is set by the SPI command REUSE_TX_PL, and is reset by the SPI commands W_TX_PAYLOAD or FLUSH TX.

### 3.5.4 Interrupt

The RF transceiver can send interrupts to the MCU. The interrupt (RFIRQ) is activated when TX_DS, RX_DR or MAX_RT are set high by the state machine in the STATUS register. RFIRQ is deactivated when the MCU writes '1' to the interrupt source bit in the STATUS register. The interrupt mask in the CONFIG register is used to select the IRQ sources that are allowed to activate RFIRQ. By setting one of the mask bits high, the corresponding interrupt source is disabled. By default all interrupt sources are enabled.

**Note:** The 3 bit pipe information in the STATUS register is updated during the RFIRQ high to low transition. The pipe information is unreliable if the STATUS register is read during a RFIRQ high to low transition.

## 3.6 Register map

You can configure and control the radio (using read and write commands) by accessing the register map through the SPI.

### 3.6.1 Register map table

All undefined bits in the table below are redundant. They are read out as '0'.

**Note:** Addresses 18 to 1B are reserved for test purposes, altering them makes the chip malfunction.

| Address (Hex) | Mnemonic | Bit | Reset Value | Type | Description |
|---|---|---|---|---|---|
| | | | | | |
| 00 | CONFIG | | | | Configuration Register |
| | Reserved | 7 | 0 | R/W | Only '0' allowed |
| | *MASK_RX_DR* | 6 | 0 | R/W | Mask interrupt caused by RX_DR<br>1: Interrupt not reflected on the RFIRQ<br>0: Reflect RX_DR as active low on RFIRQ |
| | *MASK_TX_DS* | 5 | 0 | R/W | Mask interrupt caused by TX_DS<br>1: Interrupt not reflected on the RFIRQ<br>0: Reflect TX_DS as active low interrupt on RFIRQ |
| | *MASK_MAX_RT* | 4 | 0 | R/W | Mask interrupt caused by MAX_RT<br>1: Interrupt not reflected on RFIRQ<br>0: Reflect MAX_RT as active low on RFIRQ |
| | *EN_CRC* | 3 | 1 | R/W | Enable CRC. Forced high if one of the bits in the EN_AA is high |
| | *CRCO* | 2 | 0 | R/W | CRC encoding scheme<br>'0' - 1 byte<br>'1' – 2 bytes |
| | *PWR_UP* | 1 | 0 | R/W | 1: POWER UP, 0:POWER DOWN |
| | *PRIM_RX* | 0 | 0 | R/W | RX/TX control<br>1: PRX, 0: PTX |
| | | | | | |
| 01 | EN_AA<br>Enhanced ShockBurst™ | | | | Enable 'Auto Acknowledgment' Function Disable this functionality to be compatible with nRF2401. |
| | Reserved | 7:6 | 00 | R/W | Only '00' allowed |
| | *ENAA_P5* | 5 | 1 | R/W | Enable auto acknowledgement data pipe 5 |
| | *ENAA_P4* | 4 | 1 | R/W | Enable auto acknowledgement data pipe 4 |
| | *ENAA_P3* | 3 | 1 | R/W | Enable auto acknowledgement data pipe 3 |
| | *ENAA_P2* | 2 | 1 | R/W | Enable auto acknowledgement data pipe 2 |
| | *ENAA_P1* | 1 | 1 | R/W | Enable auto acknowledgement data pipe 1 |
| | *ENAA_P0* | 0 | 1 | R/W | Enable auto acknowledgement data pipe 0 |
| | | | | | |
| 02 | EN_RXADDR | | | | Enabled RX Addresses |
| | Reserved | 7:6 | 00 | R/W | Only '00' allowed |
| | *ERX_P5* | 5 | 0 | R/W | Enable data pipe 5. |
| | *ERX_P4* | 4 | 0 | R/W | Enable data pipe 4. |
| | *ERX_P3* | 3 | 0 | R/W | Enable data pipe 3. |
| | *ERX_P2* | 2 | 0 | R/W | Enable data pipe 2. |
| | *ERX_P1* | 1 | 1 | R/W | Enable data pipe 1. |
| | *ERX_P0* | 0 | 1 | R/W | Enable data pipe 0. |

| Address (Hex) | Mnemonic | Bit | Reset Value | Type | Description |
|---|---|---|---|---|---|
| | | | | | |
| 03 | SETUP_AW | | | | Setup of Address Widths<br>(common for all data pipes) |
| | Reserved | 7:2 | 000000 | R/W | Only '000000' allowed |
| | *AW* | 1:0 | 11 | R/W | RX/TX Address field width<br>'00' - Illegal<br>'01' - 3 bytes<br>'10' - 4 bytes<br>'11' – 5 bytes<br>LSByte is used if address width is below 5 bytes |
| | | | | | |
| 04 | SETUP_RETR | | | | Setup of Automatic Retransmission |
| | *ARD*[a] | 7:4 | 0000 | R/W | Auto Retransmit Delay<br>‘0000’ – Wait 250 µs<br>‘0001’ – Wait 500 µs<br>‘0010’ – Wait 750 µs<br>……..<br>‘1111’ – Wait 4000 µs<br>(Delay defined from end of transmission to start of next transmission)[b] |
| | *ARC* | 3:0 | 0011 | R/W | Auto Retransmit Count<br>‘0000’ –Re-Transmit disabled<br>‘0001’ – Up to 1 Re-Transmit on fail of AA<br>……<br>‘1111’ – Up to 15 Re-Transmit on fail of AA |
| | | | | | |
| 05 | RF_CH | | | | RF Channel |
| | Reserved | 7 | 0 | R/W | Only '0' allowed |
| | *RF_CH* | 6:0 | 0000010 | R/W | Sets the frequency channel the RF transceiver operates on |
| | | | | | |
| 06 | RF_SETUP | | | | RF Setup Register |
| | *CONT_WAVE* | 7 | 0 | R/W | Enables continuous carrier transmit when high. |
| | Reserved | 6 | 0 | R/W | Only '0' allowed |
| | *RF_DR_LOW* | 5 | 0 | R/W | Set RF Data Rate to 250 kbps. See RF_DR_HIGH for encoding. |
| | *PLL_LOCK* | 4 | 0 | R/W | Force PLL lock signal. Only used in test |
| | *RF_DR_HIGH* | 3 | 1 | R/W | Select between the high speed data rates. This bit is don’t care if RF_DR_LOW is set.<br>Encoding:<br>RF_DR_LOW, RF_DR_HIGH:<br>‘00’ – 1 Mbps<br>‘01’ – 2 Mbps<br>‘10’ – 250 kbps<br>‘11’ – Reserved |
| | *RF_PWR* | 2:1 | 11 | R/W | Set RF output power in TX mode<br>'00' – -18 dBm<br>'01' – -12 dBm<br>'10' – -6 dBm<br>'11' – 0 dBm |

| Address (Hex) | Mnemonic | Bit | Reset Value | Type | Description |
|---|---|---|---|---|---|
| | Obsolete | 0 | | | Don't care |
| | | | | | |
| 07 | STATUS | | | | Status Register (In parallel to the SPI command word applied on the **MOSI** pin, the STATUS register is shifted serially out on the **MISO** pin) |
| | Reserved | 7 | 0 | R/W | Only '0' allowed |
| | *RX_DR* | 6 | 0 | R/W | Data Ready RX FIFO interrupt. Asserted when new data arrives RX FIFO[c].<br>Write 1 to clear bit. |
| | *TX_DS* | 5 | 0 | R/W | Data Sent TX FIFO interrupt. Asserted when packet transmitted on TX. If AUTO_ACK is activated, this bit is set high only when ACK is received.<br>Write 1 to clear bit. |
| | *MAX_RT* | 4 | 0 | R/W | Maximum number of TX retransmits interrupt<br>Write 1 to clear bit. If MAX_RT is asserted it must be cleared to enable further communication. |
| | *RX_P_NO* | 3:1 | 111 | R | Data pipe number for the payload available for reading from RX_FIFO<br>000-101: Data Pipe Number<br>110: Not Used<br>111: RX FIFO Empty |
| | *TX_FULL* | 0 | 0 | R | TX FIFO full flag.<br>1: TX FIFO full.<br>0: Available locations in TX FIFO. |
| | | | | | |
| 08 | OBSERVE_TX | | | | Transmit observe register |
| | *PLOS_CNT* | 7:4 | 0 | R | Count lost packets. The counter is overflow protected to 15, and discontinues at max until reset. The counter is reset by writing to RF_CH. |
| | *ARC_CNT* | 3:0 | 0 | R | Count retransmitted packets. The counter is reset when transmission of a new packet starts. |
| | | | | | |
| 09 | RPD | | | | |
| | Reserved | 7:1 | 000000 | R | |
| | *RPD* | 0 | 0 | R | Received Power Detector. This register is called CD (Carrier Detect) in the nRF24L01. The name is different in the RF transceiver due to the different input power level threshold for this bit. See section 3.3.4 on page 22. |
| | | | | | |
| 0A | RX_ADDR_P0 | 39:0 | 0xE7E7E7E7E7 | R/W | Receive address data pipe 0. 5 Bytes maximum length. (LSByte is written first. Write the number of bytes defined by SETUP_AW) |
| 0B | RX_ADDR_P1 | 39:0 | 0xC2C2C2C2C2 | R/W | Receive address data pipe 1. 5 Bytes maximum length. (LSByte is written first. Write the number of bytes defined by SETUP_AW) |
| 0C | RX_ADDR_P2 | 7:0 | 0xC3 | R/W | Receive address data pipe 2. Only LSB. MSBytes are equal to RX_ADDR_P1 39:8 |
| 0D | RX_ADDR_P3 | 7:0 | 0xC4 | R/W | Receive address data pipe 3. Only LSB. MSBytes are equal to RX_ADDR_P139:8 |

| Address (Hex) | Mnemonic | Bit | Reset Value | Type | Description |
|---|---|---|---|---|---|
| 0E | RX_ADDR_P4 | 7:0 | 0xC5 | R/W | Receive address data pipe 4. Only LSB. MSBytes are equal to RX_ADDR_P[139:8] |
| 0F | RX_ADDR_P5 | 7:0 | 0xC6 | R/W | Receive address data pipe 5. Only LSB. MSBytes are equal to RX_ADDR_P[139:8] |
| | | | | | |
| 10 | TX_ADDR | 39:0 | 0xE7E7E7E7E7 | R/W | Transmit address. Used for a PTX operation only. (LSByte is written first)<br>Set RX_ADDR_P0 equal to this address to handle automatic acknowledge if this is a PTX operation with Enhanced ShockBurst™ enabled. |
| | | | | | |
| 11 | RX_PW_P0 | | | | |
| | Reserved | 7:6 | 00 | R/W | Only '00' allowed |
| | *RX_PW_P0* | 5:0 | 0 | R/W | Number of bytes in RX payload in data pipe 0 (1 to 32 bytes).<br>0 Pipe not used<br>1 = 1 byte<br>…<br>32 = 32 bytes |
| | | | | | |
| 12 | RX_PW_P1 | | | | |
| | Reserved | 7:6 | 00 | R/W | Only '00' allowed |
| | *RX_PW_P1* | 5:0 | 0 | R/W | Number of bytes in RX payload in data pipe 1 (1 to 32 bytes).<br>0 Pipe not used<br>1 = 1 byte<br>…<br>32 = 32 bytes |
| | | | | | |
| 13 | RX_PW_P2 | | | | |
| | Reserved | 7:6 | 00 | R/W | Only '00' allowed |
| | *RX_PW_P2* | 5:0 | 0 | R/W | Number of bytes in RX payload in data pipe 2 (1 to 32 bytes).<br>0 Pipe not used<br>1 = 1 byte<br>…<br>32 = 32 bytes |
| | | | | | |
| 14 | RX_PW_P3 | | | | |
| | Reserved | 7:6 | 00 | R/W | Only '00' allowed |
| | *RX_PW_P3* | 5:0 | 0 | R/W | Number of bytes in RX payload in data pipe 3 (1 to 32 bytes).<br>0 Pipe not used<br>1 = 1 byte<br>…<br>32 = 32 bytes |
| | | | | | |
| 15 | RX_PW_P4 | | | | |
| | Reserved | 7:6 | 00 | R/W | Only '00' allowed |

| Address (Hex) | Mnemonic | Bit | Reset Value | Type | Description |
|---|---|---|---|---|---|
| | *RX_PW_P4* | 5:0 | 0 | R/W | Number of bytes in RX payload in data pipe 4 (1 to 32 bytes).<br>0 Pipe not used<br>1 = 1 byte<br>…<br>32 = 32 bytes |
| | | | | | |
| 16 | RX_PW_P5 | | | | |
| | Reserved | 7:6 | 00 | R/W | Only '00' allowed |
| | *RX_PW_P5* | 5:0 | 0 | R/W | Number of bytes in RX payload in data pipe 5 (1 to 32 bytes).<br>0 Pipe not used<br>1 = 1 byte<br>…<br>32 = 32 bytes |
| | | | | | |
| 17 | FIFO_STATUS | | | | FIFO Status Register |
| | Reserved | 7 | 0 | R/W | Only '0' allowed |
| | *TX_REUSE* | 6 | 0 | R | Used for a PTX operation<br>Pulse the rfce high for at least 10 µs to Reuse last transmitted payload. TX payload reuse is active until W_TX_PAYLOAD or FLUSH TX is executed.<br>TX_REUSE is set by the SPI command REUSE_TX_PL, and is reset by the SPI commands W_TX_PAYLOAD or FLUSH TX |
| | *TX_FULL* | 5 | 0 | R | TX FIFO full flag. 1: TX FIFO full. 0: Available locations in TX FIFO. |
| | *TX_EMPTY* | 4 | 1 | R | TX FIFO empty flag.<br>1: TX FIFO empty.<br>0: Data in TX FIFO. |
| | Reserved | 3:2 | 00 | R/W | Only '00' allowed |
| | *RX_FULL* | 1 | 0 | R | RX FIFO full flag.<br>1: RX FIFO full.<br>0: Available locations in RX FIFO. |
| | *RX_EMPTY* | 0 | 1 | R | RX FIFO empty flag.<br>1: RX FIFO empty.<br>0: Data in RX FIFO. |
| N/A | ACK_PLD | 255:0 | X | W | Written by separate SPI command<br>ACK packet payload to data pipe number PPP given in SPI command.<br>Used in RX mode only.<br>Maximum three ACK packet payloads can be pending. Payloads with same PPP are handled first in first out. |
| N/A | TX_PLD | 255:0 | X | W | Written by separate SPI command TX data payload register 1 - 32 bytes.<br>This register is implemented as a FIFO with three levels.<br>Used in TX mode only. |

| Address (Hex) | Mnemonic | Bit | Reset Value | Type | Description |
|---|---|---|---|---|---|
| N/A | RX_PLD | 255:0 | X | R | Read by separate SPI command.<br>RX data payload register. 1 - 32 bytes.<br>This register is implemented as a FIFO with three levels.<br>All RX channels share the same FIFO. |
| | | | | | |
| 1C | DYNPD | | | | Enable dynamic payload length |
| | Reserved | 7:6 | 0 | R/W | Only '00' allowed |
| | *DPL_P5* | 5 | 0 | R/W | Enable dynamic payload length data pipe 5.<br>(Requires EN_DPL and ENAA_P5) |
| | *DPL_P4* | 4 | 0 | R/W | Enable dynamic payload length data pipe 4.<br>(Requires EN_DPL and ENAA_P4) |
| | *DPL_P3* | 3 | 0 | R/W | Enable dynamic payload length data pipe 3.<br>(Requires EN_DPL and ENAA_P3) |
| | *DPL_P2* | 2 | 0 | R/W | Enable dynamic payload length data pipe 2.<br>(Requires EN_DPL and ENAA_P2) |
| | *DPL_P1* | 1 | 0 | R/W | Enable dynamic payload length data pipe 1.<br>(Requires EN_DPL and ENAA_P1) |
| | *DPL_P0* | 0 | 0 | R/W | Enable dynamic payload length data pipe 0.<br>(Requires EN_DPL and ENAA_P0) |
| | | | | | |
| 1D | FEATURE | | | R/W | Feature Register |
| | Reserved | 7:3 | 0 | R/W | Only '00000' allowed |
| | *EN_DPL* | 2 | 0 | R/W | Enables Dynamic Payload Length |
| | *EN_ACK_PAY*[d] | 1 | 0 | R/W | Enables Payload with ACK |
| | *EN_DYN_ACK* | 0 | 0 | R/W | Enables the W_TX_PAYLOAD_NOACK command |
| | | | | | |

a. Please take care when setting this parameter. If the ACK payload is more than 15 byte in 2 Mbps mode the ARD must be 500 µs or more, if the ACK payload is more than 5byte in 1 Mbps mode the ARD must be 500 µs or more. In 250 kbps mode (even when the payload is not in ACK) the ARD must be 500µs or more.

b. This is the time the PTX is waiting for an ACK packet before a retransmit is made. The PTX is in RX mode for a minimum of 250 µs, but it stays in RX mode to the end of the packet if that is longer than 250 µs. Then it goes to standby-I mode for the rest of the specified ARD. After the ARD it goes to TX mode and then retransmits the packet.

c. The RX_DR IRQ is asserted by a new packet arrival event. The procedure for handling this interrupt should be: 1) read payload through SPI, 2) clear RX_DR IRQ, 3) read FIFO_STATUS to check if there are more payloads available in RX FIFO, 4) if there are more data in RX FIFO, repeat from step 1).

d. If ACK packet payload is activated, ACK packets have dynamic payload lengths and the Dynamic Payload Length feature should be enabled for pipe 0 on the PTX and PRX. This is to ensure that they receive the ACK packets with payloads. If the ACK payload is more than 15 byte in 2 Mbps mode the ARD must be 500 µs or more, and if the ACK payload is more than 5 byte in 1 Mbps mode the ARD must be 500 µs or more. In 250 kbps mode (even when the payload is not in ACK) the ARD must be 500 µs or more.

*Table 11. Register map of the RF transceiver*

# 4 MCU

The nRF24LE1 OTP contains a fast 8-bit MCU, which executes the normal 8051 instruction set.

The architecture eliminates redundant bus states and implements parallel execution of fetch and execution phases. Most of the one-byte instructions are performed in a single cycle. The MCU uses one clock per cycle. This leads to a performance improvement rate of 8.0 (in terms of MIPS) with respect to legacy 8051 devices.

The original 8051 had a 12 clock architecture. A machine cycle needed 12 clocks and most instructions were either one or two machine cycles. Except for MUL and DIV instructions, the 8051 used either 12 or 24 clocks for each instruction. Each cycle in the 8051 also used two memory fetches. In many cases, the second fetch was a dummy, and extra clocks were wasted.

Table 12. shows the speed advantage compared to a legacy 8051. A speed advantage of 12 implies that the instruction is executed twelve times faster. The average speed advantage is 8.0. However, the real speed improvement seen in any system depends on the instruction mix.

| Speed advantage | Number of instructions | Number of opcodes |
|---|---|---|
| 24 | 1 | 1 |
| 12 | 27 | 83 |
| 9.6 | 2 | 2 |
| 8 | 16 | 38 |
| 6 | 44 | 89 |
| 4.8 | 1 | 2 |
| 4 | 18 | 31 |
| 3 | 2 | 9 |
| Average: 8.0 | Sum: 111 | Sum: 255 |

*Table 12. Speed advantage summary*

## 4.1 Block diagram

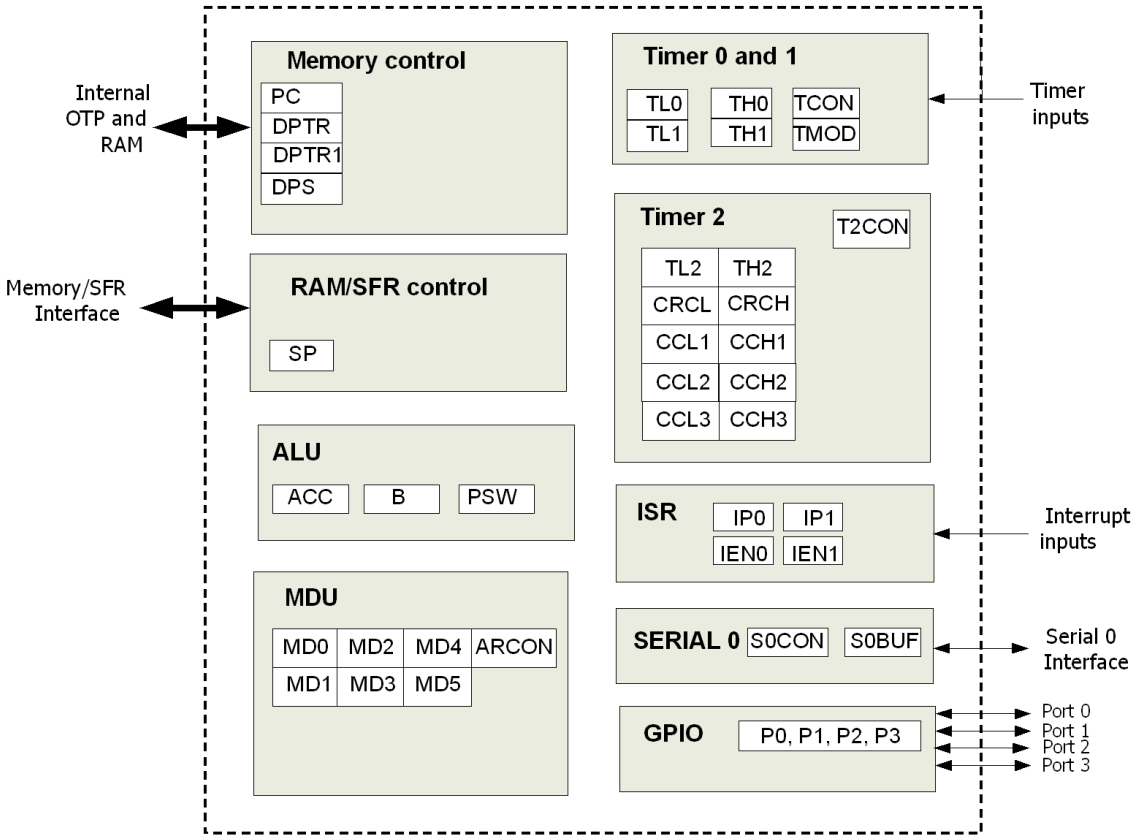



*Figure 29. MCU block diagram*

## 4.2 Features

- Control Unit
  - 8-bit instruction decoder
  - Reduced instruction cycle time (up to 12 times in respect to standard 80C51)
- Arithmetic-Logic Unit
  - 8-bit arithmetic and logical operations
  - Boolean manipulations
  - 8×8 bit multiplication and 8 / 8 bit division
- Multiplication-Division Unit
  - 16×16 bit multiplication
  - 32 / 16 bit and 16 / 16 bit division
  - 32-bit normalization
  - 32-bit L/R shifting
- Three 16-bit Timers/Counters
  - 80C51-like Timer 0 & 1
  - 80515-like Timer 2
- Compare/Capture Unit, dedicated to Timer 2
  - Software control capture
- Full Duplex Serial Interfaces
  - Serial 0 (80C51-like)
  - Synchronous mode, fixed baud rate

- 8-bit UART mode, variable baud rate
- 9-bit UART mode, fixed baud rate
- 9-bit UART mode, variable baud rate
- Baud Rate Generator

- Interrupt Controller
  - Four Priority Levels with 13 interrupt sources
- Memory interface
  - 16-bit address bus
  - Dual Data Pointer for fast data block transfer

## 4.3 Functional description

### 4.3.1 Arithmetic Logic Unit (ALU)

The Arithmetic Logic Unit (ALU) provides 8-bit division, 8-bit multiplication, and 8-bit addition with or without carry. The ALU also provides 8-bit subtraction with borrow and some bitwise logic operations, that is, logical AND, OR, Exclusive OR or NOT.

All operations are unsigned integer operations. Additionally, the ALU can increment or decrement 8-bit registers. For accumulator only, it can rotate left or right through carry or not, swap nibbles, clear or complement bits and perform a decimal adjustment.

The ALU is handled by three registers, which are memory mapped as special function registers. Operands for operations may come from accumulator `ACC`, register `B` or from outside of the unit. The result may be stored in accumulator `ACC` or may be driven outside of the unit. The control register, that contains flags such as carry, overflow or parity, is the `PSW` (Program Status Word) register.

The nRF24LE1 OTP also contains an on-chip co-processor MDU (Multiplication Division Unit). This unit enables 32-bit division, 16-bit multiplication, shift and normalize operations, see chapter 14 on page 119 for details.

### 4.3.2 Instruction set summary

All instructions are binary code compatible and perform the same functions as they do within the legacy 8051 processor. The following tables give a summary of the instruction set with the required corresponding clock cycles.

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| ADD A,Rn | Add register to accumulator | 0x28-0x2F | 1 | 1 |
| ADD A,direct | Add directly addressed data to accumulator | 0x25 | 2 | 2 |
| ADD A,@Ri | Add indirectly addressed data to accumulator | 0x26-0x27 | 1 | 2 |
| ADD A,#data | Add immediate data to accumulator | 0x24 | 2 | 2 |
| ADDC A,Rn | Add register to accumulator with carry | 0x38-0x3F | 1 | 1 |
| ADDC A, direct | Add directly addressed data to accumulator with carry | 0x35 | 2 | 2 |
| ADDC A,@Ri | Add indirectly addressed data to accumulator with carry | 0x36-0x37 | 1 | 2 |
| ADDC A,#data | Add immediate data to accumulator with carry | 0x34 | 2 | 2 |
| SUBB A,Rn | Subtract register from accumulator with borrow | 0x98-0x9F | 1 | 1 |
| SUBB A, direct | Subtract directly addressed data from accumulator with borrow | 0x95 | 2 | 2 |
| SUBB A, @Ri | Subtract indirectly addressed data from accumulator with borrow | 0x96-0x97 | 1 | 2 |
| SUBB A, #data | Subtract immediate data from accumulator with borrow | 0x94 | 2 | 2 |
| INC A | Increment accumulator | 0x04 | 1 | 1 |

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| INC Rn | Increment register | 0x08-0x0F | 1 | 2 |
| INC direct | Increment directly addressed location | 0x05 | 2 | 3 |
| INC @Ri | Increment indirectly addressed location | 0x06-0x07 | 1 | 3 |
| INC DPTR | Increment data pointer | 0xA3 | 1 | 1 |
| DEC A | Decrement accumulator | 0x14 | 1 | 1 |
| DEC Rn | Decrement register | 0x18-0x1F | 1 | 2 |
| DEC direct | Decrement directly addressed location | 0x15 | 2 | 3 |
| DEC @Ri | Decrement indirectly addressed location | 0x16-0x17 | 1 | 3 |
| MUL AB | Multiply A and B | 0xA4 | 1 | 5 |
| DIV | Divide A by B | 0x84 | 1 | 5 |
| DA A | Decimal adjust accumulator | 0xD4 | 1 | 1 |

*Table 13. Arithmetic operations*

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| ANL A, Rn | AND register to accumulator | 0x58-0x5F | 1 | 1 |
| ANL A,direct | AND directly addressed data to accumulator | 0x55 | 2 | 2 |
| ANL A,@Ri | AND indirectly addressed data to accumulator | 0x56-0x57 | 1 | 2 |
| ANL A,#data | AND immediate data to accumulator | 0x54 | 2 | 2 |
| ANL direct,A | AND accumulator to directly addressed location | 0x52 | 2 | 3 |
| ANL direct,#data | AND immediate data to directly addressed location | 0x53 | 3 | 4 |
| ORL A,Rn | OR register to accumulator | 0x48-0x4F | 1 | 1 |
| ORL A,direct | OR directly addressed data to accumulator | 0x45 | 2 | 2 |
| ORL A,@Ri | OR indirectly addressed data to accumulator | 0x46-0x47 | 1 | 2 |
| ORL A,#data | OR immediate data to accumulator | 0x44 | 2 | 2 |
| ORL direct,A | OR accumulator to directly addressed location | 0x42 | 2 | 3 |
| ORL direct,#data | OR immediate data to directly addressed location | 0x43 | 3 | 4 |
| XRL A,Rn | Exclusive OR register to accumulator | 0x68-0x6F | 1 | 1 |
| XRL A, direct | Exclusive OR directly addressed data to accumulator | 0x66-0x67 | 1 | 2 |
| XRL A,@Ri | Exclusive OR indirectly addressed data to accumulator | 0x66-0x67 | 1 | 2 |
| XRL A,#data | Exclusive OR immediate data to accumulator | 0x64 | 2 | 2 |
| XRL direct,A | Exclusive OR accumulator to directly addressed location | 0x62 | 2 | 3 |
| XRL direct,#data | Exclusive OR immediate data to directly addressed location | 0x63 | 3 | 4 |
| CLR A | Clear accumulator | 0xE4 | 1 | 1 |
| CPL A | Complement accumulator | 0xF4 | 1 | 1 |
| RL A | Rotate accumulator left | 0x23 | 1 | 1 |
| RLC A | Rotate accumulator left through carry | 0x33 | 1 | 1 |
| RR A | Rotate accumulator right | 0x03 | 1 | 1 |
| RRC A | Rotate accumulator right through carry | 0x13 | 1 | 1 |
| SWAP A | Swap nibbles within the accumulator | 0xC4 | 1 | 1 |

*Table 14. Logic operations*

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| MOV A,Rn | Move register to accumulator | 0xE8-0xEF | 1 | 1 |
| MOV A,direct | Move directly addressed data to accumulator | 0xE5 | 2 | 2 |
| MOV A,@Ri | Move indirectly addressed data to accumulator | 0xE6-0xE7 | 1 | 2 |
| MOV A,#data | Move immediate data to accumulator | 0x74 | 2 | 2 |
| MOV Rn,A | Move accumulator to register | 0xF8-0xFF | 1 | 2 |
| MOV Rn,direct | Move directly addressed data to register | 0xA8-0xAF | 2 | 4 |
| MOV Rn,#data | Move immediate data to register | 0x78-0x7F | 2 | 2 |
| MOV direct,A | Move accumulator to direct | 0xF5 | 2 | 3 |
| MOV direct,Rn | Move register to direct | 0x88-0x8F | 2 | 3 |
| MOV directl,direct2 | Move directly addressed data to directly addressed location | 0x85 | 3 | 4 |
| MOV direct,@Ri | Move indirectly addressed data to directly addressed location | 0x86-0x87 | 2 | 4 |
| MOV direct,#data | Move immediate data to directly addressed location | 0x75 | 3 | 3 |
| MOV @Ri,A | Move accumulator to indirectly addressed location | 0xF6-0xF7 | 1 | 3 |
| MOV @Ri,direct | Move directly addressed data to indirectly addressed location | 0xA6–0xA7 | 2 | 5 |
| MOV @Ri,#data | Move immediate data to indirectly addressed location | 0x76–0x77 | 2 | 3 |
| MOV DPTR,#datal6 | Load data pointer with a 16-bit immediate | 0x90 | 3 | 3 |
| MOVC A,@A+DPTR | Load accumulator with a code byte relative to DPTR | 0x93 | 1 | 3 |
| MOVC A,@A+PC | Load accumulator with a code byte relative to PC | 0x83 | 1 | 3 |
| MOVX A,@Ri | Move[a] external RAM (8-bit addr) to accumulator | 0xE2–0xE3 | 1 | 4 |
| MOVX A,@DPTR | Move[a] external RAM (16-bit addr) to accumulator | 0xE0 | 1 | 4 |
| MOVX @Ri,A | Move[a] accumulator to external RAM (8-bit addr) | 0xF2–0xF3 | 1 | 5 |
| MOVX @DPTR,A | Move[a] accumulator to external RAM (16-bit addr) | 0xF0 | 1 | 5 |
| PUSH direct | Push directly addressed data onto stack | 0xC0 | 2 | 4 |
| POP direct | Pop directly addressed location from stack | 0xD0 | 2 | 3 |
| XCH A,Rn | Exchange register with accumulator | 0xC8–0xCF | 1 | 2 |
| XCH A,direct | Exchange directly addressed location with accumulator | 0xC5 | 2 | 3 |
| XCH A,@Ri | Exchange indirect RAM with accumulator | 0xC6–0xC7 | 1 | 3 |
| XCHD A,@Ri | Exchange low-order nibbles of indirect and accumulator | 0xD6–0xD7 | 1 | 3 |

a. The MOVX instructions perform one of two actions depending on the state of pmw bit (pcon.4).

*Table 15. Data transfer operations*

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| ACALL addr11 | Absolute subroutine call | xxx10001b | 2 | 6 |
| LCALL addr16 | Long subroutine call | 0x12 | 3 | 6 |
| RET | Return from subroutine | 0x22 | 1 | 4 |
| RETI | Return from interrupt | 0x32 | 1 | 4 |
| AJMP addr11 | Absolute jump | xxx00001b | 2 | 3 |
| LJMP addrl6 | Long jump | 0x02 | 3 | 4 |
| SJMP rel | Short jump (relative address) | 0x80 | 2 | 3 |
| JMP @A+DPTR | Jump indirect relative to the DPTR | 0x73 | 1 | 2 |
| JZ rel | Jump if accumulator is zero | 0x60 | 2 | 3 |
| JNZ rel | Jump if accumulator is not zero | 0x70 | 2 | 3 |
| JC rel | Jump if carry flag is set | 0x40 | 2 | 3 |
| JNC rel | Jump if carry flag is not set | 0x50 | 2 | 3 |
| JB bit, rel | Jump if directly addressed bit is set | 0x20 | 3 | 4 |
| JNB bit, rel | Jump if directly addressed bit is not set | 0x30 | 3 | 4 |
| JBC bit, rel | Jump if directly addressed bit is set and clear bit | 0x10 | 3 | 4 |
| CJNE A, direct, rel | Compare directly addressed data to accumulator and jump if not equal | 0xB5 | 3 | 4 |
| CJNE A,#data,rel | Compare immediate data to accumulator and jump if not equal | 0xB4 | 3 | 4 |
| CJNE Rn, #data, rel | Compare immediate data to register and jump if not equal | 0xB8–0xBF | 3 | 4 |
| CJNE @Ri, #data, rel | Compare immediate data to indirect addressed value and jump if not equal | 0xB6–B7 | 3 | 4 |
| DJNZ Rn, rel | Decrement register and jump if not zero | 0xD8–DF | 2 | 3 |
| DJNZ direct, rel | Decrement directly addressed location and jump if not zero | 0xD5 | 3 | 4 |
| NOP | No operation | 0x00 | 1 | 1 |

*Table 16. Program branches*

| Mnemonic | Description | Code | Bytes | Cycles |
|---|---|---|---|---|
| CLR C | Clear carry flag | 0xC3 | 1 | 1 |
| CLR bit | Clear directly addressed bit | 0xC2 | 2 | 3 |
| SETB C | Set carry flag | 0xD3 | 1 | 1 |
| SETB bit | Set directly addressed bit | 0xD2 | 2 | 3 |
| CPL C | Complement carry flag | 0xB3 | 1 | 1 |
| CPL bit | Complement directly addressed bit | 0xB2 | 2 | 3 |
| ANL C,bit | AND directly addressed bit to carry flag | 0x82 | 2 | 2 |
| ANL C,/bit | AND complement of directly addressed bit to carry | 0xB0 | 2 | 2 |
| ORL C,bit | OR directly addressed bit to carry flag | 0x72 | 2 | 2 |
| ORL C,/bit | OR complement of directly addressed bit to carry | 0xA0 | 2 | 2 |
| MOV C,bit | Move directly addressed bit to carry flag | 0xA2 | 2 | 2 |
| MOV bit,C | Move carry flag to directly addressed bit | 0x92 | 2 | 3 |

*Table 17. Boolean manipulation*

### 4.3.3 Opcode map

| Opcode | Mnemonic | Opcode | Mnemonic | Opcode | Mnemonic |
|---|---|---|---|---|---|
| 00H | NOP | 56H | ANL A,@R0 | ACH | MOV R4,direct |
| 01H | AJMP addr11 | 57H | ANL A,@R1 | ADH | MOV R5,direct |
| 02H | JUMP addrl6 | 58H | ANL A,R0 | AE | MOV R6,direct |
| 03H | RRA | 59H | ANL A,R1 | AFH | MOV R7,direct |
| 04H | INCA | 5AH | ANL A,R2 | B0H | ANL C,/bit |
| 05H | INC direct | 5BH | ANL A,R3 | B1H | ACALL addr11 |
| 06H | INC @R0 | 5CH | ANL A,R4 | B2H | CPL bit |
| 07H | INC @R1 | 5DH | ANL A,R5 | B3H | CPLC |
| 08H | INC R0 | 5EH | ANL A,R6 | B4H | CJNE A,#data,rel |
| 09H | INC R1 | 5FH | ANL A,R7 | B5H | CJNE A, direct, rel |
| 0AH | INC R2 | 60H | JZ rel | B6H | CJNE @R0,#data,rel |
| 0BH | INC R3 | 61H | AJMP addr11 | B7H | CJNE @R1, #data,rel |
| 0CH | INC R4 | 62H | XRL direct, A | B8H | CJNE R0, #data,rel |
| 0DH | INC R5 | 63H | XRL direct, #data | B9H | CJNE R1,#data,rel |
| 0EH | INC R6 | 64H | XRL A, #data | BAH | CJNE R2,#data,rel |
| 0FH | INC R7 | 65H | XRL A,direct | BBH | CJNE R3,#data,rel |
| 10H | JBC bit, rel | 66H | XRLA,@R0 | BCH | CJNE R4,#data,rel |
| 11H | ACALL addr11 | 67H | XRL A,@R1 | BDH | CJNE R5,#data,rel |
| 12H | LCALL add r16 | 68H | XRL A,R0 | BEH | CJNE R6,#data,rel |
| 13H | RRC A | 69H | XRL A,R1 | BFH | CJNE R7,#data,rel |
| 14H | DEC A | 6AH | XRL A,R2 | C0H | PUSH direct |
| 15H | DEC direct | 6BH | XRL A,R3 | C1H | AJMP addr11 |
| 16H | DEC @R0 | 6CH | XRL A,R4 | C2H | CLR bit |
| 17H | DEC @R1 | 6DH | XRL A,R5 | C3H | CLR C |
| 18H | DEC R0 | 6EH | XRL A,R6 | C4H | SWAP A |
| 19H | DEC R1 | 6FH | XRL A,R7 | C5H | XCH A, direct |
| 1AH | DEC R2 | 70H | JNZ rel | C6H | XCH A,@R0 |
| 1BH | DECR3 | 71H | ACALL addr11 | C7H | XCH A,@R1 |
| 1CH | DECR4 | 72H | ORL C, bit | C8H | XCH A,R0 |
| 1DH | DECR5 | 73H | JMP @A+DPTR | C9H | XCH A,R1 |
| 1EH | DECR6 | 74H | MOV A, #data | CAH | XCH A,R2 |
| 1FH | DECR7 | 75H | MOV direct, #data | CBH | XCHA,R3 |
| 20H | JB bit, rel | 76H | MOV @R0,#data | CCH | XCH A,R4 |
| 21H | AJMP addr11 | 77H | MOV @R1, #data | CDH | XCH A,R5 |
| 22H | RET | 78H | MOV R0, #data | CEH | XCH A,R6 |
| 23H | RL A | 79H | MOV R1, #data | CFH | XCHA,R7 |
| 24H | ADD A, #data | 7AH | MOV R2, #data | D0H | POP direct |
| 25H | ADD A, direct | 7BH | MOV R3, #data | D1H | ACALL addr11 |
| 26H | ADD A,@R0 | 7CH | MOV R4, #data | D2H | SETB bit |
| 27H | ADD A,@R1 | 7DH | MOV R5, #data | D3H | SETB C |
| 28H | ADD A,R0 | 7EH | MOV R6, #data | D4H | DAA |
| 29H | ADD A,R1 | 7FH | MOV R7, #data | D5H | DJNZ direct, rel |
| 2AH | ADD A,R2 | 80H | SJMP rel | D6H | XCHDA,@R0 |
| 2BH | ADD A,R3 | 81H | AJMP addr11 | D7H | XCHD A,@R1 |
| 2CH | ADD A,R4 | 82H | ANL C, bit | D8H | DJNZ R0,rel |
| 2DH | ADD A,R5 | 83H | MOVC A,@A+PC | D9H | DJNZ R1,rel |
| 2EH | ADD A,R6 | 84H | DIV AB | DAH | DJNZ R2,rel |
| 2FH | ADD A,R7 | 85H | MOV direct, direct | DBH | DJNZ R3,rel |
| 30H | JNB bit, rel | 86H | MOV direct,@R0 | DCH | DJNZ R4,rel |
| 31H | ACALL addr11 | 87H | MOV direct,@R1 | DDH | DJNZ R5,rel |

| Opcode | Mnemonic | Opcode | Mnemonic | Opcode | Mnemonic |
|---|---|---|---|---|---|
| 32H | RETI | 88H | MOV direct,R0 | DE | DJNZ R6,rel |
| 33H | RLC A | 89H | MOV direct,R1 | DFH | DJNZ R7,rel |
| 34H | ADDC A,#data | 8AH | MOV direct,R2 | E0H | MOVX A,@DPTR |
| 35H | ADDC A, direct | 8BH | MOV direct,R3 | E1H | AJMP addr11 |
| 36H | ADDC A,@R0 | 8CH | MOV direct,R4 | E2H | MOVX A,@R0 |
| 37H | ADDC A,@R1 | 8DH | MOV direct, R5 | E3H | MOVX A,@R1 |
| 38H | ADDC A,R0 | 8EH | MOV direct,R6 | E4H | CLR A |
| 39H | ADDC A,R1 | 8FH | MOV direct,R7 | E5H | MOVA, direct |
| 3AH | ADDC A,R2 | 90H | MOV DPTR, #datal6 | E6H | MOVA,@R0 |
| 3BH | ADDC A,R3 | 91H | ACALL addr11 | E7H | MOV A,@R1 |
| 3CH | ADDC A,R4 | 92H | MOV bit, C | E8H | MOV A,R0 |
| 3DH | ADDC A,R5 | 93H | MOVCA,@A+DPTR | E9H | MOV A,R1 |
| 3EH | ADDC A,R6 | 94H | SUBB A, #data | EAH | MOV A,R2 |
| 3FH | ADDC A,R7 | 95H | SUBB A, direct | EBH | MOV A,R3 |
| 40H | JC rel | 96H | SUBB A,@R0 | ECH | MOV A,R4 |
| 41H | AJMP addr11 | 97H | SUBB A,@R1 | EDH | MOV A,R5 |
| 42H | ORL direct, A | 98H | SUBB A, R0 | EEH | MOV A,R6 |
| 43H | ORL direct, #data | 99H | SUBB A,R1 | EFH | MOV A,R7 |
| 44H | ORL A, #data | 9AH | SUBB A,R2 | F0H | MOVX @DPTR,A |
| 45H | ORL A, direct | 9BH | SUBB A,R3 | F1H | ACALL addr11 |
| 46H | ORL A,@R0 | 9CH | SUBB A,R4 | F2H | MOVX @R0,A |
| 47H | ORL A,@R1 | 9DH | SUBB A,R5 | F3H | MOVX @R1,A |
| 48H | ORL A,R0 | 9EH | SUBB A,R6 | F4H | CPL A |
| 49H | ORL A,R1 | 9FH | SUBB A,R7 | F5H | MOV direct, A |
| 4AH | ORL A,R2 | A0H | ORL C,/bit | F6H | MOV @R0,A |
| 4BH | ORLA,R3 | A1H | AJMP addr11 | F7H | MOV @R1,A |
| 4CH | ORL A,R4 | A2H | MOV C, bit | F8H | MOV R0,A |
| 4DH | ORL A,R5 | A3H | INC DPTR | F9H | MOV R1,A |
| 4EH | ORL A,R6 | A4H | MUL AB | FAH | MOV R2,A |
| 4FH | ORLA,R7 | A5H | - | FBH | MOV R3,A |
| 50H | JNC rel | A6H | MOV @R0,direct | FCH | MOV R4,A |
| 51H | ACALL addr11 | A7H | MOV @R1,direct | FDH | MOV R5,A |
| 52H | ANL direct, A | A8H | MOV R0,direct | FEH | MOV R6,A |
| 53H | ANL direct, #data | A9H | MOV R1,direct | FFH | MOV R7,A |
| 54H | ANL A, #data | AAH | MOV R2,direct | | |
| 55H | ANL A, direct | ABH | MOV R3,direct | | |

*Table 18. Opcode map*

# 5 Memory and I/O organization

The MCU has 64 kB of separate address space for code and data, an area of 256 byte for internal data (IRAM) and an area of 128 byte for Special Function Registers (SFR).

The nRF24LE1 OTP memory blocks has a default setting of 16 kB program memory (OTP), 1 kB of data memory (SRAM) and 1 kB of non-volatile data memory (OTP), see default memory map in Figure 30. Read- or write access to the grey areas in this figure may behave unpredictably.

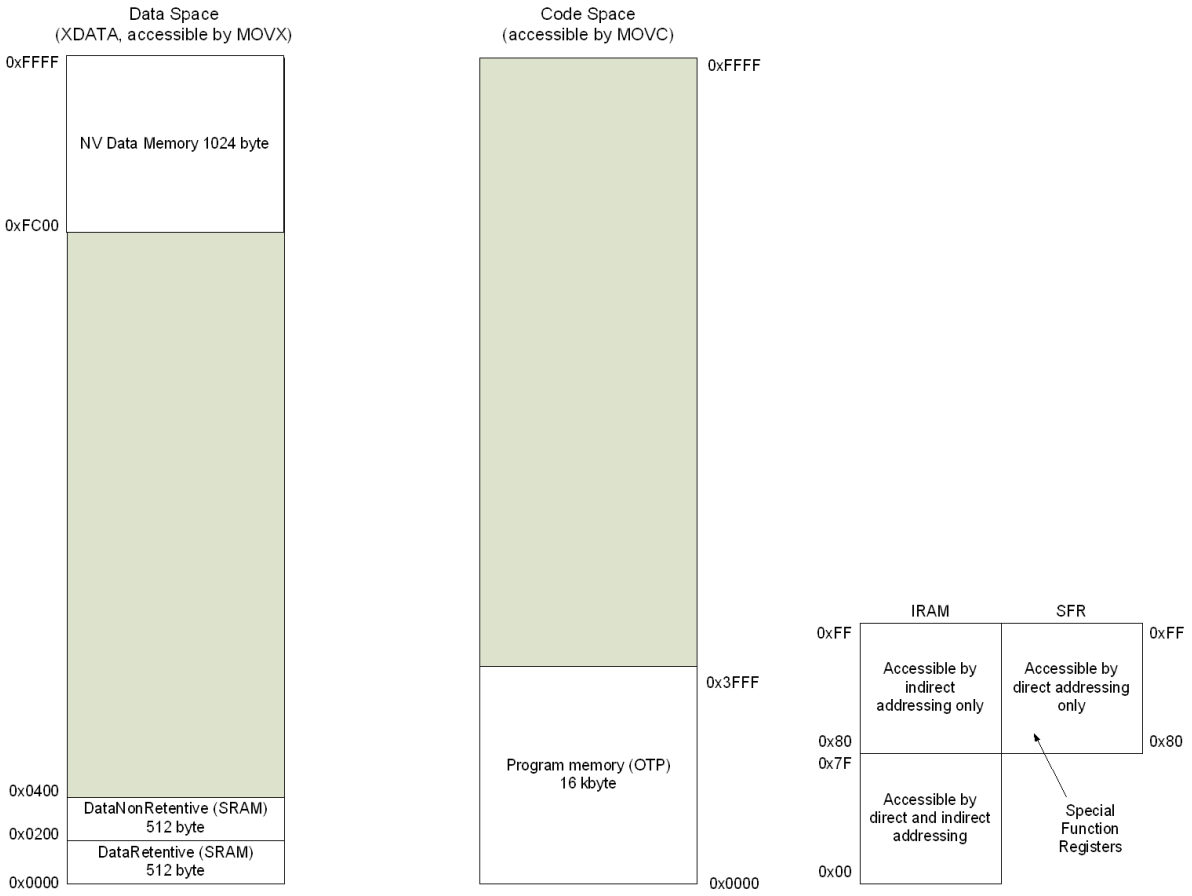



*Figure 30. Memory map*

The lower 128 bytes of the IRAM contains work registers (0x00–0x1F) and bit addressable memory (0x20–0x2F). The upper half can only be accessed by indirect addressing.

The lowest 32 bytes of the IRAM form four banks, each consisting of eight registers (R0–R7). Two bits of the program memory status word (PSW) select which bank is used. The next 16 bytes of memory form a block of bit-addressable memory, accessible through bit addresses 0x00–0x7F.

## 5.1 PDATA memory addressing

The nRF24LE1 OTP supports PDATA (Paged Data memory) addressing into data space. One page (256 bytes) can be accessed by an indirect addressing scheme through registers R0 and R1 (@R0, @R1).

The MPAGE register controls the start address of the PDATA page:

| Addr | Bit | R/W | Function | Reset value: 0x00 |
|---|---|---|---|---|
| 0xC9 | 7:0 | R/W | Start address of the PDATA page | |

*Table 19. MPAGE register*

MPAGE sets the upper half of the 16 bit address space. For example, setting MPAGE to 0x80 starts PDATA from address 0x8000.

## 5.2 MCU Special Function Registers

### 5.2.1 Accumulator - ACC

Accumulator is used by most of the MCU instructions to hold the operand and to store the result of an operation. The mnemonics for accumulator specific instructions refer to accumulator as A, not ACC.

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---|---|---|---|---|---|---|---|---|
| 0xE0 | acc.7 | acc.6 | acc.5 | acc.4 | acc.3 | acc.2 | acc.1 | acc.0 |

*Table 20. ACC register*

### 5.2.2 B Register – B

The B register is used during multiplying and division instructions. It can also be used as a scratch-pad register to hold temporary data.

| Address | bit7 | bit6 | bit5 | bit4 | bit3 | bit2 | bit1 | bit0 |
|---|---|---|---|---|---|---|---|---|
| 0xF0 | b.7 | b.6 | b.5 | b.4 | b.3 | b.2 | b.1 | b.0 |

*Table 21. B register*

### 5.2.3 Program Status Word Register - PSW

The PSW register contains status bits that reflect the current state of the MCU.

**Note:** The Parity bit can only be modified by hardware upon the state of ACC register.

| Address | Bit | Name | Description |
|---|---|---|---|
| 0xD0 | 7 | cy | Carry flag: Carry bit in arithmetic operations and accumulator for Boolean operations. |
| | 6 | ac | Auxiliary Carry flag: Set if there is a carry-out from 3rd bit of Accumulator in BCD operations |
| | 5 | f0 | General purpose flag 0 |
| | 4:3 | rs | Register bank select, bank 0..3 (0x00–0x07, 0x08–0x0f, 0x10–0x17, 0x18–0x1f) |
| | 2 | ov | Overflow flag: Set if overflow in Accumulator during arithmetic operations |
| | 1 | f1 | General purpose flag 1 |
| | 0 | p | Parity flag: Set if odd number of '1' in ACC. |

*Table 22. PSW register*

### 5.2.4 Stack Pointer – SP

This register points to the top of stack in internal data memory space. It is used to store the return address of a program before executing interrupt routine or subprograms. The SP is incremented before executing PUSH or CALL instruction and it is decremented after executing POP or RET(I) instruction (it always points to the top of stack).

| Address | Register name |
|---|---|
| 0x81 | SP |

*Table 23. SP register*

### 5.2.5 Data Pointer – DPH, DPL

| Address | Register name |
|---|---|
| 0x82 | DPL |
| 0x83 | DPH |

*Table 24. Data Pointer register (DPH:DPL)*

The Data Pointer Registers can be accessed through DPL and DPH. The actual data pointer is selected by DPS register.

These registers are intended to hold 16-bit address in the indirect addressing mode used by MOVX (move external memory), MOVC (move program memory) or JMP (computed branch) instructions. They may be manipulated as 16-bit register or as two separate 8-bit registers. DPH holds higher byte and DPL holds lower byte of indirect address.

It is generally used to access external code or data space (for example, MOVC A, @A+DPTR or MOV A, @DPTR respectively).

### 5.2.6 Data Pointer 1 – DPH1, DPL1

| Address | Register name |
|---|---|
| 0x84 | DPL1 |
| 0x85 | DPH1 |

*Table 25. Data Pointer 1 register (DPH1:DPL1)*

The Data Pointer Register 1 can be accessed through DPL1 and DPH1. The actual data pointer is selected by DPS register.

These registers are intended to hold 16-bit address in the indirect addressing mode used by MOVX (move external memory), MOVC (move program memory) or JMP (computed branch) instructions. They may be manipulated as 16-bit register or as two separate 8-bit registers. DPH1 holds higher byte and DPL1 holds lower byte of indirect address.

It is generally used to access external code or data space (for example, MOVC A,@A+DPTR or MOV A,@DPTR respectively).

The Data Pointer 1 is an extension to the standard 8051 architecture to speed up block data transfers.

### 5.2.7 Data Pointer Select Register – DPS

The MCU contains two Data Pointer registers. Both of them can be used as 16-bits address source for indirect addressing. The DPS register serves for selecting active data pointer register.

| Address | Bit | Name | Description |
|---|---|---|---|
| 0x92 | 7:1 | - | Not used |
| | 0 | dps | Data Pointer Select. 0: select DPH:DPL, 1: select DPH1:DPL1 |

*Table 26. DPS register*

### 5.2.8 PCON register

The PCON register is used to control the Program Memory Write Mode and Serial Port 0 baud rate doubler.

| Address | Bit | Name | Description |
|---|---|---|---|
| 0x87 | 7 | smod | Serial port 0 baud rate select, see Table 91. on page 151 and Table 93. on page 152. |
| | 6 | gf3 | General purpose flag 3 |
| | 5 | gf2 | General purpose flag 2 |
| | 4 | pmw | Program memory write mode:<br>1: MOVX instructions will access memory code space<br>0: MOVX instructions will access memory data space |
| | 3 | gf1 | General purpose flag 1 |
| | 2 | gfo | General purpose flag 0 |
| | 1 | - | Not used. This bit must always be cleared. Always read as 0. |
| | 0 | - | Not used. This bit must always be cleared. Always read as 0. |

*Table 27. PCON register*

### 5.2.9 Special Function Register Map

The map of Special Function Registers is shown in Table 28. Undefined locations must not be read or written.

| Address | X000 | X001 | X010 | X011 | X100 | X101 | X110 | X111 |
|---|---|---|---|---|---|---|---|---|
| 0xF8–0xFF | FSR | FPCR | FCR | Reserved | SPIMCON0 | SPIMCON1 | SPIM-STAT | SPIMDAT |
| 0xF0–0xF7 | B | | | | | | | |
| 0xE8–0xEF | RFCON | MD0 | MD1 | MD2 | MD3 | MD4 | MD5 | ARCON |
| 0xE0–0xE7 | ACC | W2CON1 | W2CON0 | Reserved | SPIRCON0 | SPIRCON1 | SPIRSTAT | SPIRDAT |
| 0xD8–0xDF | ADCON | W2SADR | W2DAT | COMP-CON | POFCON | CCPDATIA | CCP-DATIB | CCPDATO |
| 0xD0–0xD7 | PSW | ADCCON3 | ADCCON2 | ADCCON1 | ADCDATH | ADCDATL | RNGCTL | RNGDAT |
| 0xC8–0xCF | T2CON | MPAGE | CRCL | CRCH | TL2 | TH2 | WUOPC1 | WUOPC0 |
| 0xC0–0xC7 | IRCON | CCEN | CCL1 | CCH1 | CCL2 | CCH2 | CCL3 | CCH3 |
| 0xB8–0xBF | IEN1 | IP1 | S0RELH | Reserved | SPISCON0 | | SPISSTAT | SPISDAT |
| 0xB0–0Xb7 | P3 | RSTREAS | PWM-CON | RTC2CON | RTC2CMP0 | RTC2CMP1 | RTC2CPT00 | |
| 0xA8–0xAF | IEN0 | IP0 | S0RELL | RTC2CPT01 | RTC2CPT10 | CLKLFC-TRL | OPMCON | WDSV |
| 0xA0–0xA7 | P2 | PWMDC0 | PWMDC1 | CLKCTRL | PWRDWN | WUCON | INTEXP | MEMCON |
| 0x98–0x9F | S0CON | S0BUF | Reserved | Reserved | Reserved | Reserved | P0CON | P1CON |
| 0x90–0x97 | P1 | free | DPS | P0DIR | P1DIR | P2DIR | P3DIR | P2CON |
| 0x88–0x8F | TCON | TMOD | TL0 | TL1 | TH0 | TH1 | Reserved | P3CON |
| 0x80–0x87 | P0 | SP | DPL | DPH | DPL1 | DPH1 | Reserved | |

*Table 28. Special Function Registers locations*

The registers in the X000 column in `B` register are both byte and bit addressable. The other registers are only byte addressable.

### 5.2.10 Special Function Registers reset values

| Register name | Address | Reset value | Description |
|---|---|---|---|
| ACC | 0xE0 | 0x00 | Accumulator |
| ADCCON1 | 0xD3 | 0x00 | ADC Configuration Register 1 |
| ADCCON2 | 0xD2 | 0x00 | ADC Configuration Register 2 |
| ADCCON3 | 0xD1 | 0x00 | ADC Configuration Register 3 |
| ADCDATH | 0xD4 | 0x00 | ADC Data high byte |
| ADCDATL | 0xD5 | 0x00 | ADC Data low byte |
| ARCON | 0xEF | 0x00 | Arithmetic Control Register |
| B | 0xF0 | 0x00 | B Register |
| CCEN | 0xC1 | 0x00 | Compare/Capture Enable Register |
| CCH1 | 0xC3 | 0x00 | Compare/Capture Register 1, high byte |
| CCH2 | 0xC5 | 0x00 | Compare/Capture Register 2, high byte |
| CCH3 | 0xC7 | 0x00 | Compare/Capture Register 3, high byte |
| CCL1 | 0xC2 | 0x00 | Compare/Capture Register 1, low byte |
| CCL2 | 0xC4 | 0x00 | Compare/Capture Register 2, low byte |
| CCL3 | 0xC6 | 0x00 | Compare/Capture Register 3, low byte |
| CCPDATIA | 0xDD | 0x00 | Encryption/Decryption accelerator Data In Register A |
| CCPDATIB | 0xDE | 0x00 | Encryption/Decryption accelerator Data In Register B |
| CCPDATO | 0xDF | 0x00 | Encryption/Decryption accelerator Data Out Register |
| CLKLFCTRL | 0xAD | 0x07 | 32 kHz (CLKLF) control |
| CLKCTRL | 0xA3 | 0x00 | Clock control |
| COMPCON | 0xDB | 0x00 | Comparator Control Register |
| CRCH | 0xCB | 0x00 | Compare/Reload/Capture Register, high byte |
| CRCL | 0xCA | 0x00 | Compare/Reload/Capture Register, low byte |
| DPH | 0x83 | 0x00 | Data Pointer High 0 |
| DPL | 0x82 | 0x00 | Data Pointer Low 0 |
| DPH1 | 0x85 | 0x00 | Data Pointer High 1 |
| DPL1 | 0x84 | 0x00 | Data Pointer Low 1 |
| DPS | 0x92 | 0x00 | Data Pointer Select Register |
| FCR | 0xFA | | OTP Command Register |
| FPCR | 0xF9 | | OTP Protect Configuration Register |
| FSR | 0xF8 | | OTP Status Register |
| IEN0 | 0xA8 | 0x00 | Interrupt Enable Register 0 |
| IEN1 | 0xB8 | 0x00 | Interrupt Priority Register / Enable Register 1 |
| INTEXP | 0xA6 | 0x01 | Interrupt Expander Register |
| IP0 | 0xA9 | 0x00 | Interrupt Priority Register 0 |
| IP1 | 0xB9 | 0x00 | Interrupt Priority Register 1 |
| IRCON | 0xC0 | 0x00 | Interrupt Request Control Register |
| MD0 | 0xE9 | 0x00 | Multiplication/Division Register 0 |
| MD1 | 0xEA | 0x00 | Multiplication/Division Register 1 |
| MD2 | 0xEB | 0x00 | Multiplication/Division Register 2 |
| MD3 | 0xEC | 0x00 | Multiplication/Division Register 3 |
| MD4 | 0xED | 0x00 | Multiplication/Division Register 4 |
| MD5 | 0xEE | 0x00 | Multiplication/Division Register 5 |
| MEMCON | 0xA7 | 0x00 | Memory Configuration Register |
| MPAGE | 0xC9 | 0x00 | Start address of the PDATA page |
| OPMCON | 0xAE | 0x00 | Operational Mode Control |
| P0 | 0x80 | 0xFF | Port 0 value |
| P0CON | 0x9E | 0x10 | Port 0 Configuration Register |
| P0DIR | 0x93 | 0xFF | Port 0 pin direction control |

| Register name | Address | Reset value | Description |
|---|---|---|---|
| P1 | 0x90 | 0xFF | Port 1 value |
| P1CON | 0x9F | 0x10 | Port 1 Configuration Register |
| P1DIR | 0x94 | 0xFF | Port 1 pin direction control |
| P2 | 0xA0 | 0xFF | Port 2 value |
| P2CON | 0x97 | 0x10 | Port 2 Configuration Register |
| P2DIR | 0x95 | 0xFF | Port 2 pin direction control |
| P3 | 0xB0 | 0xFF | Port 3 value |
| P3CON | 0x8F | 0x10 | Port 3 Configuration Register |
| P3DIR | 0x96 | 0xFF | Port 3 pin direction control |
| POFCON | 0xDC | 0x00 | Power-fail Comparator Configuration Register |
| PSW | 0xD0 | 0x00 | Program Status Word |
| PWMCON | 0xB2 | 0x00 | PWM Configuration Register |
| PWMDC0 | 0xA1 | 0x00 | PWM Duty Cycle for channel 0 |
| PWMDC1 | 0xA2 | 0x00 | PWM Duty Cycle for channel 1 |
| PWRDWN | 0xA4 | 0x00 | Power-down control |
| RFCON | 0xE8 | 0x02 | RF transceiver Control Register |
| RNGCTL | 0xD6 | 0x40 | Random Number Generator Control Register |
| RNGDAT | 0xD7 | 0x00 | Random Number Generator Data Register |
| RSTREAS | 0xB1 | 0x00 | Reset Reason Register |
| RTC2CMP0 | 0xB4 | 0xFF | RTC2 Compare Value Register 0 |
| RTC2CMP1 | 0xB5 | 0xFF | RTC2 Compare Value Register 1 |
| RTC2CON | 0xB3 | 0x00 | RTC2 Configuration Register |
| RTC2CPT00 | 0xB6 | 0x00 | RTC2 Capture Value Register 00 |
| RTC2CPT01 | 0xAB | 0x00 | RTC2 Capture Value Register 01 |
| RTC2CPT10 | 0xAC | 0x00 | RTC2 Capture Value Register 10 |
| S0BUF | 0x99 | 0x00 | Serial Port 0, Data Buffer |
| S0CON | 0x98 | 0x00 | Serial Port 0, Control Register |
| S0RELH | 0xBA | 0x03 | Serial Port 0, Reload Register, high byte |
| S0RELL | 0xAA | 0xD9 | Serial Port 0, Reload Register, low byte |
| SP | 0x81 | 0x07 | Stack Pointer |
| SPIMCON0 | 0xFC | 0x02 | SPI Master Configuration Register 0 |
| SPIMDAT | 0xFF | 0x00 | SPI Master Data Register |
| SPIMSTAT | 0xFE | 0x03 | SPI Master Status Register |
| SPIRCON0 | 0xE4 | 0x01 | RF transceiver SPI Master Configuration Register 0 |
| SPIRCON1 | 0xE5 | 0x0F | RF transceiver SPI Master Configuration Register 1 |
| SPIRDAT | 0xE7 | 0x00 | RF transceiver SPI Master Data Register |
| SPIRSTAT | 0xE6 | 0x03 | RF transceiver SPI Master Status Register |
| SPISCON0 | 0xBC | 0xF0 | SPI Slave Configuration Register 0 |
| SPISCON1 | 0xBD | 0x0F | SPI Slave Configuration Register 1 |
| SPISDAT | 0xBF | 0x00 | SPI Slave Data Register |
| SPISSTAT | 0xBE | 0x03 | SPI Slave Status Register |
| T2CON | 0xC8 | 0x00 | Timer 2 Control Register |
| TCON | 0x88 | 0x00 | Timer/Counter Control Register |
| TH0 | 0x8C | 0x00 | Timer 0, high byte |
| TH1 | 0x8D | 0x00 | Timer 1, high byte |
| TH2 | 0xCD | 0x00 | Timer 2, high byte |
| TL0 | 0x8A | 0x00 | Timer 0, low byte |
| TL1 | 0x8B | 0x00 | Timer 1, low byte |
| TL2 | 0xCC | 0x00 | Timer 2, low byte |
| TMOD | 0x89 | 0x00 | Timer Mode Register |
| W2CON0 | 0xE2 | 0x80 | 2-Wire Configuration Register 0 |

| Register name | Address | Reset value | Description |
|---|---|---|---|
| W2CON1 | 0xE1 | 0x00 | 2-Wire Configuration Register 1/Status Register |
| W2DAT | 0xDA | 0x00 | 2-Wire Data Register |
| W2SADR | 0xD9 | 0x00 | 2-Wire Slave Address Register |
| ADCON | 0xD8 | 0x00 | Serial Port 0 Baud Rate Select register (only adcon.7 bit used) |
| WDSW | 0xAF | 0x00 | Watchdog Start Value Register |
| WUCON | 0xA5 | 0x00 | Wakeup configuration register |
| WUOPC0 | 0xCF | 0x00 | Wakeup On Pin Configuration Register 0 |
| WUOPC1 | 0xCE | 0x00 | Wakeup On Pin Configuration Register 1 |

*Table 29. Special Function Registers reset values*

# 6 OTP memory

This section describes the operation of the embedded OTP (One Time Programmable) memory. The primary use for this memory is for read only program and data storage, but the MCU may also perform write operations, for instance to store pairing information persistently.

The OTP memory is configured and programmed through an external SPI slave interface. After programming, read and write operations from the external interfaces can be disabled for code protection.

## 6.1 Features

- 16 kB code memory
- 1 kB NV data memory
- 256 byte of user programmable Info Page
- Direct SPI programmable
- Readback protection

## 6.2 Block diagram

The OTP block in nRF24LE1 OTP is split in 16 kB of generic code space memory and 1 kB of Non Volatile data memory. OTP is accessible by SPI slave during programming and MCU as selected by the PROG pin. A memory controller generate the necessary timing for accessing the OTP memory.

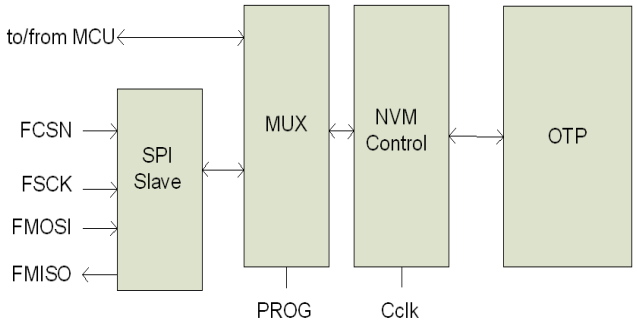



*Figure 31. nRF24LE1 OTP block diagram*

## 6.3 Functional description

The OTP block gives the MCU its code space for program storage and NVM space for storing of application data. The different parts of the memory can be accessed by the MCU through normal code and data space operations.

Configuration and setup of the memory behavior during normal mode (that is, when MCU is running application code) is defined by data stored in a separate InfoPage. During the chip reset/start-up sequence the configuration data in the InfoPage is read and stored in the memory configuration SFR's.

### 6.3.1 Using the NV data memory

The 1 kB NV memory is addressed by the MCU as XDATA memory in the range 0xFC00–0xFFFF. This memory is physically mapped (for SPI access) to OTP memory area 04400–0x47FF

The NV data memory is read/written as normal OTP as described in section 6.3.5 on page 74. When writing the NV data memory the PMW bit in the PCON register must be cleared. When writing/reading the NV data memory, XDATA memory addresses must be used.

### 6.3.2 OTP memory configuration

The on-chip OTP memory is divided into 2 blocks, the 16 kB + 1 kB NVM main block and a 512 byte Information Page (IP).

The memory configuration is stored in the InfoPage (IP) and can be used to disable read and write to the OTP from the SPI interface.

All configuration of the OTP memory must be done through the external SPI interface. The configuration information is stored in the InfoPage during programming of the device and is read out to the OTP configuration SFRs during each reset/startup sequence of the circuit.

#### 6.3.2.1 InfoPage content

The InfoPage is a separate page (512 bytes) of OTP memory that contains Nordic Semiconductor system tuning parameters and the configurable options of the OTP memory. Any changes to the OTP memory configuration must be done by updating this page. The InfoPage content is as follows:

| InfoPage data | Name | Size | Address | Comment |
|---|---|---|---|---|
| Device system | DSYS [a] | 32 bytes | 0x00 | Reserved for device use. Do not modify. |
| Reserved | - | 3 bytes | 0x20 | Reserved, must be 0xFF |
| OTP main block read back protect | RDISMB | 1 byte | 0x23 | Disable OTP main block access from SPI.<br><br>Byte value:<br>• 0xFF: OTP main block accessible from external interfaces<br>• Other value: No read/ write of OTP main block from external interfaces. Only read of info page<br>• Can only be changed once by SPI command RDISMB. |
| Reserved | | 220 bytes | 0x24 | Reserved, must be 0xFF |
| For user data | | 256 bytes | 0x100 | Free to use |

a. **NOTE:** This InfoPage area is used to store nRF24LE1 OTP system and tuning parameters. Writing to this area WILL cause changes to device behavior and performance.

*Table 30. InfoPage content*

**DSYS - Device System parameters**

This InfoPage area is used by the nRF24LE1 OTP to store core data like tuning parameters. Changing this area will cause severe changes to device behavior.

RDISMB - Read DISable Main Block

By changing this byte from 0xFF the SPI and other external interfaces no longer have any access to the OTP main block and only read access to the InfoPage.

The byte is changed by the `RDISMB` SPI command and since it cuts the SPI access to the OTP main block, must be the last command sent to a nRF24LE1 OTP during OTP programming.

#### 6.3.2.2 Memory configuration SFR

During the boot sequence the content of the OTP InfoPage (IP) is transferred to the memory configuration SFR's. The same memory configuration SFR's are used for later interfacing from both SPI and MCU.

| Address (hex) | Mnemonic | Bit | Reset value | SPI access | SFR access | Description |
|---|---|---|---|---|---|---|
| 0xF8 | `FSR` | | | | | OTP Status Register |
| | *WEN* | 5 | 0 | R/W[a] | R/W | OTP write enable latch.<br>Enables OTP write operations from SPI<br><br>WEN will be cleared after each SPI write operation, but not after a MCU operations. |
| | *RDYN* | 4 | 1 | R | R | OTP ready flag, active low.<br><br>Will be set when read out of OTP IP is completed in the MCU boot sequence |
| | *INFEN* | 3 | 0 | R/W | R/W | OTP IP Enable<br>Will re-direct general SPI read/write commands from the OTP main block to the IP. |
| | *RDISMB* | 2 | 1, until read from OTP InfoPage | R/W[a] | R | OTP main block readback protection enabled, active low.<br>RDISMB:<br>0: External interfaces have full access to the OTP<br>1: Main block read/write and InfoPage write commands from SPI disabled. |
| | - | 1 | 1 | R | R | Reserved |
| | | 0 | 0 | R | R/W | Reserved |

a. Can only be written indirectly through InfoPage, by dedicated SPI command, and is ignored by `WRSR` command.

*Table 31. Registers for MCU and SPI for OTP configuration control*

### 6.3.3 Brown-out

There is an on-chip power-fail brown-out detector, see chapter 12 on page 112, which ensures that any OTP memory program access will be ignored when the Power Fail (POF) signal, see Figure 48. on page 112, is active. Both the microcontroller and the OTP memory write operation still function according to specification, and any write operation that was started will be completed. The Power-fail comparator is disabled after startup and can be enabled by setting bit 7 in `POFCON` (refer to Figure 64. on page 152.)

If the supply voltage drops below ~1.7V, that is when the Brown-Out Reset (BOR) signal (see Figure 51. on page 114) is active, the chip will be reset. If the power supply rises again before reaching the reset threshold, there will be no reset. In order to have an indication that shows this has happened, one will need to enable the Power Failure interrupt (POFIRQ, see Table 46. on page 95).

To ensure proper programming of the OTP in the cases where power supply may be unreliable, the user should take the following precautions:

- Make sure the data read back from the OTP is identical to what is written to OTP. The mechanism above will guarantee that the data is safely stored to OTP if the value does compare. If the compare fails, the write has been ignored due to a power supply event.
- Make sure that the time from “Power fail” to “Reset” is longer than one write operation (800 µs) by a sufficient reservoir on the supply.

### 6.3.4 Temperature restriction for OTP programming

The OTP program operations may be unreliable at temperatures below 0 ºC.

### 6.3.5 OTP programming from the MCU

This section describes how you can write the OTP memory using the MCU.

#### 6.3.5.1 MCU operations in the main block

When an OTP write is initiated, the MCU is halted for 12800 clock cycles (800 µs) for each byte written.

The clock frequency of the microcontroller must be 16 MHz during OTP write operations.

To allow write OTP operations the MCU must run the following sequence:

1. Set `WEN` (bit 5) in the FSR register high to enable OTP write access. The OTP is now open for write from the MCU until `WEN` in FSR is set low again.
2. If write to code space is intended, set PMW (bit 4) in the `PCON` register high to enable program memory write mode.
3. Programming the OTP is done through normal memory write operations from the MCU. Bytes are written individually (there is no auto increment) to the OTP using the specific memory address.

When the programming code executes from the OTP, write operation is self-timed and the CPU stops until the operation is finished. If the programming code executes from the XDATA RAM the code must wait until the operation has finished. This can be done either by polling the RDYN bit in the `FSR` register to go low or by a wait loop. Do not set `WEN` low before the write operation is finished. Memory address is identical to the OTP address, see chapter 5 on page 63 for memory mapping.

### 6.3.6 OTP programming through SPI

The on-chip OTP is designed to interface a standard SPI device for programming. The interface uses an 8-bit command register and a set of commands to program and configure the OTP memory.

### 6.3.6.1 PROG pin requirements

During SPI programming of the OTP, the **PROG** pin is used as a high voltage supply.

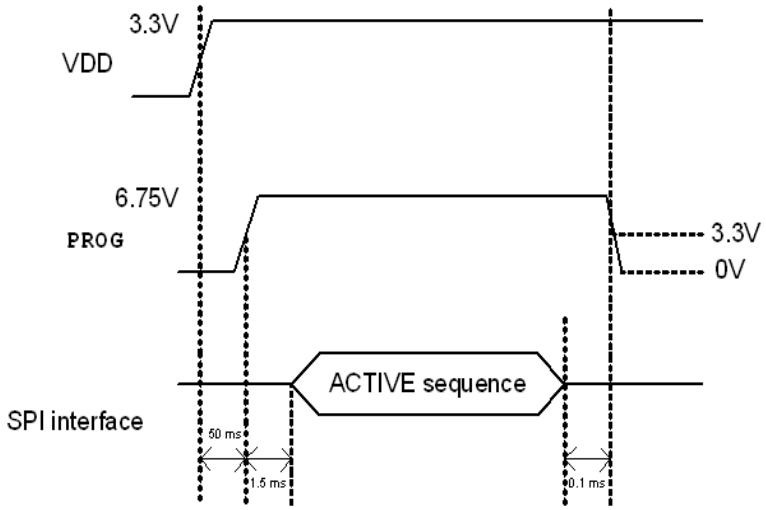



*Figure 32. Waveform for OTP programming via SPI*

The **PROG** pin is raised to VPP (6.75V) minimum of 1.5 ms before first SPI command is issued. The **PROG** pins should be held at VPP for 0.1 ms after the last SPI command is completed. For SPI read the **PROG** pin supply may be reduced to 3.3V.

**Note:** An external pull-down resistor on the **PROG** pin is required. If external programming is not needed **PROG** must be connected to ground.

### 6.3.6.2 SPI slave interface

To program the memory the SPI slave interface is used. SPI slave connection to the OTP memory is activated by setting pin **PROG** = 1 while the reset pin is kept inactive. After the **PROG** pin is set to 1, apply a pulse on the **RESET** pin (Pull **RESET** pin low for a minimum of 0.2 µs and return to high.) Selected nRF24LE1 OTP **GPIO** pins are automatically configured as a SPI slave as shown in Table 32. Further information on SPI slave timing can be found in chapter 18 on page 143.

| | 24 pin-4×4 | 32 pin-5×5 | 48 pin-7×7 |
|---|---|---|---|
| FCSN | **P0.5** | **P1.1** | **P2.0** |
| FMISO | **P0.4** | **P1.0** | **P1.6** |
| FMOSI | **P0.3** | **P0.7** | **P1.5** |
| FSCK | **P0.2** | **P0.5** | **P1.2** |

*Table 32. OTP SPI slave physical interface for each nRF24LE1 OTP package alternative*

**Note:** After activation of the **PROG** pin you must wait at least 1.5 ms before you input the first OTP command.

The program interface uses an 8 bit command register and a set of commands to program and configure the OTP memory.

| Command | Command format | Address | # Data bytes | Command operation |
|---|---|---|---|---|
| WREN | 0x06 | NA | 0 | Set OTP write enable latch.<br>Bit WEN in register FSR |
| WRDIS | 0x04 | NA | 0 | Reset OTP write enable latch.<br>Bit WEN in register FSR |
| RDSR | 0x05 | NA | 1 | Read OTP Status Register (FSR) |
| WRSR | 0x01 | NA | 1 | Write OTP Status Register (FSR) |
| READ | 0x03 | 2 bytes, First OTP address to to be read | 1–17408 | Read data from OTP |
| PROGRAM | 0x02 | 2 bytes, first OTP address to be written | 1–1024 | Write data to OTP<br><br>**Note:** WEN must be set. |
| RDFPCR | 0x89 | NA | 1 | Read OTP Protect Configuration Register FPCR |
| RDISMB | 0x85 | NA | 0 | Enable OTP readback protection<br><br>**Note:** WEN must be set. |

*Table 33. OTP operation commands*

The signaling of the SPI interface is shown in Figure 33. and Figure 34.

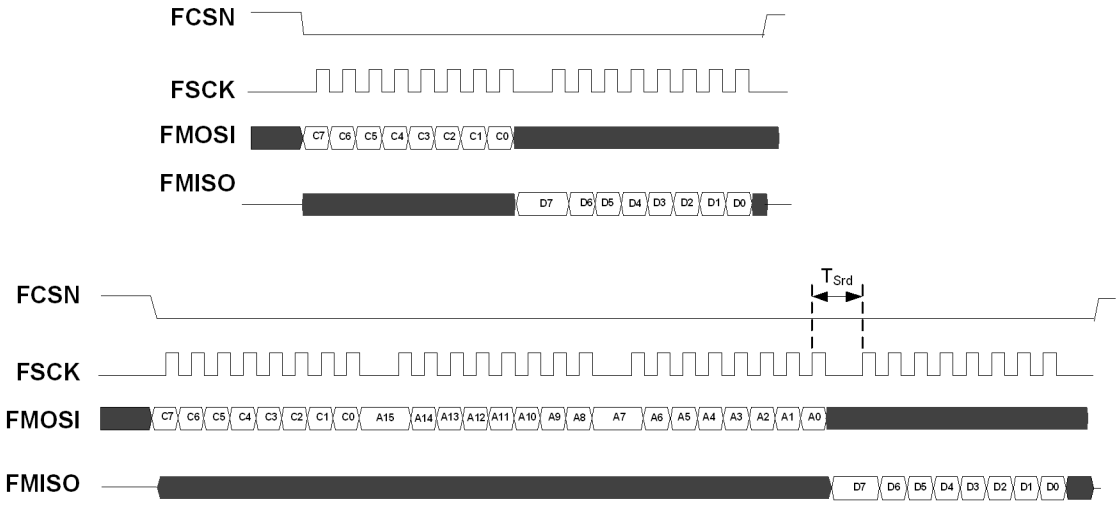



*Figure 33. SPI read operation for direct and addressed command*

**Note:** For the READ command there must be a delay, $T_{Srd}$, from the last address bit to the first data bit. Minimum value of $T_{Srd}$ is 4 clock cycles (**XC1**) + 10ns, which is 260 ns when **XC1** is 16 MHz.

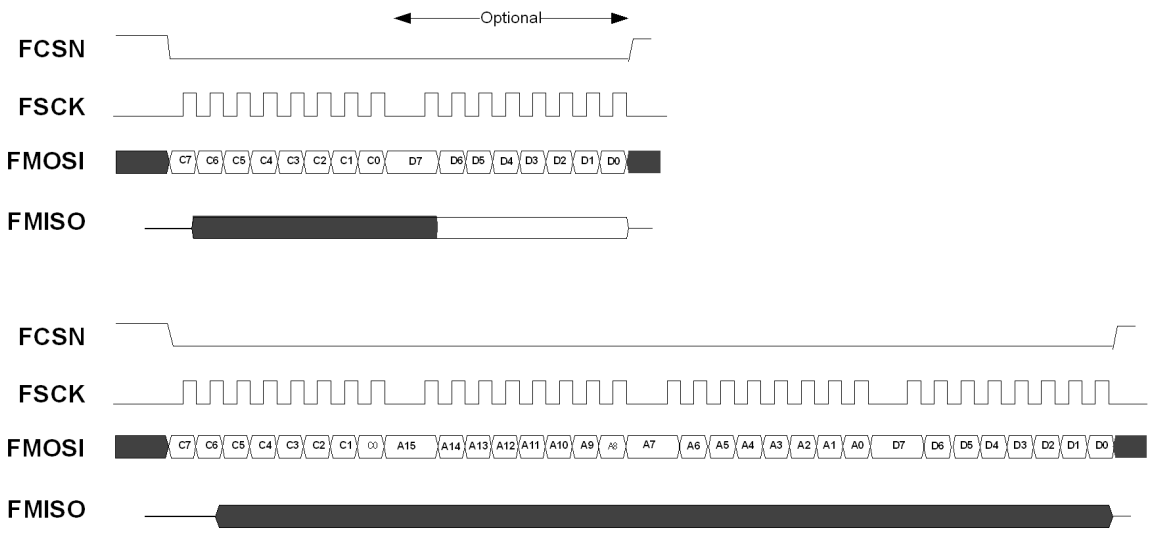



*Figure 34. SPI write operations for direct and addressed commands*

| Abbreviations | Description |
|---|---|
| Cx | SPI Command bit |
| Ax | OTP address. Sequence MS to LS byte, MS to LS bit. |
| Dx | SPI data bit, Sequence LS to MS byte, MS to LS bit. Presence depending on SPI command. |

*Table 34. OTP SPI interface signal abbreviations*

**WREN / WRDIS OTP write enable/disable:**

SPI commands `WREN` and `WRDIS` sets and resets the OTP write enable latch WEN in register `FSR`. This latch enables all write operations in the OTP blocks.

The device will power-up in write disable state, and automatically go back to write disable state after each write SPI command (FCSN set high). Each write command over the SPI interface must therefore be preceded by a `WREN` command.

Both `WREN` and `WRDIS` are 1-byte SPI commands with no data.

**RDSR / WRSR read/write OTP status register**

SPI commands `RDSR` and `WRSR` read and writes to the OTP status register FSR. Both commands are 1 and are followed by a data byte for the FSR content, see chapter 18 on page 143.

**READ**

SPI command `READ` reads out the content of an addressed position in the OTP main block. It must be followed by 2 bytes denoting the start address of the read operation, see chapter 18 on page 143. If bit INFEN in register FSR is enabled, the read operation will be conducted from the InfoPage instead.

If the FCSN line is kept active after the first data byte is read out the read command can be extended, the address is auto incremented and data continues to shift out. The internal address counter rolls over when the highest address is reached, allowing the complete memory to be read in one continuous read command.

A read back of the OTP main block content is only possible if the read disable bit **RDISMB** in the FSR register is not set.

**PROGRAM**

SPI command PROGRAM, programs the content of the addressed position in the OTP main block. It must be followed by 2 bytes denoting the start address of the write operation, see chapter 18 on page 143. If bit INFEN in register FSR is enabled, the write operation will access the InfoPage instead.

Before each write operation the write enable latch WEN must be enabled through the WREN SPI command. It is possible to write up to 1 kB in one PROGRAM command. The first byte can be at any address.

The device automatically returns to OTP write disable (WEN=0) after completion of a PROGRAM command (pin FCSN=1).

**RDISMB - Enable Read DISable of MainBlock)**

SPI command RDISMB enables the readback protection of the OTP. The command disables all read and write access to the OTP main block from the SPI interface. It also disables write operations in the InfoPage, but read InfoPage read operations are still possible. This will protect code and data in the device from being retrieved through the SPI interface.

Before the RDISMB command the write enable latch WEN must be enabled through the WREN SPI command. Once the RDISMB command is sent all SPI connection/control of the OTP from the SPI interface is lost. It is important that this command is the last one to be sent in a OTP programming sequence.

The command is a 1 byte command with no data.

# 7 Random Access memory (RAM)

The nRF24LE1 OTP contains two separate RAM blocks. These blocks are used to save temporary data or programs.

The MCU internal RAM (IRAM) is the fastest and most flexible, but with only 256 bytes it is very limited.

To accommodate more temporary storage of data or code the nRF24LE1 OTP has an additional 1024x8bit (1kB) SRAM memory block default located in the XDATA address space from address 0x0000 to 0x03FF. The location of the SRAM blocks in the MCU address space can be changed, see section 7.1.

A special feature of the nRF24LE1 OTP SRAM block is that it is composed of two physical 512 byte blocks called DataRetentive (lower 512 bytes) and DataNonRetentive. DataRetentive, in contrast to DataNonRetentive, keeps its memory content during the Memory Retention power down modes (see chapter 11 on page 100).

## 7.1 SRAM configuration

It is possible to configure the location in address space of each SRAM block as described in Figure 35.

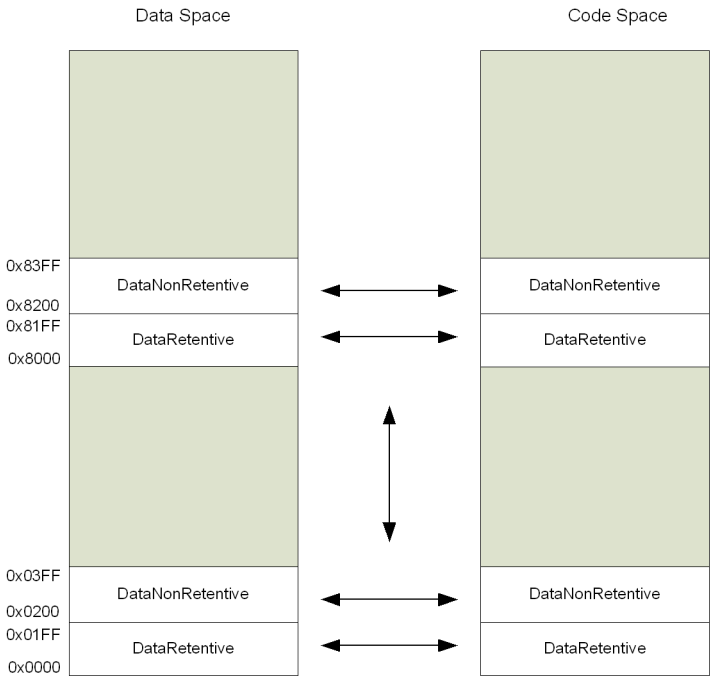



*Figure 35. Configurability of SRAM address space location*

You can address the SRAM memory blocks both as data and code. The MEMCON register controls this behavior:

| Addr | Bit | R/W | Function | Reset value: 0x00 |
|---|---|---|---|---|
| 0xA7 | 7:3 | - | Reserved | |
| | 2 | R/W | SRAM address location:<br>0: SRAM blocks start from address 0x0000<br>1: SRAM blocks start from address 0x8000 | |
| | 1 | R/W | DataNonRetentive mapping:<br>0: Mapped as data<br>1: Mapped as code | |
| | 0 | R/W | DataRetentive mapping:<br>0: Mapped as data<br>1: Mapped as code | |

*Table 35. MEMCON register*

# 8 Timers/counters

The nRF24LE1 OTP contains a set of counters used for timing up important system events. One of the timers (RTC2) is also available in power down mode where it can be used as a wakeup source.

## 8.1 Features

nRF24LE1 OTP includes the following set of timers/counters:

- Three 16-bit timers/counters (Timer 0, Timer 1 and Timer 2) which can operate as either a timer with a clock rate based on the MCU clock, or as an event counter clocked by signals from the programmable digital I/O.
- RTC2 is a configurable, linear, 16-bit real time clock with capture and compare capabilities. Input clock frequency is 32.768 KHz.

## 8.2 Block diagram

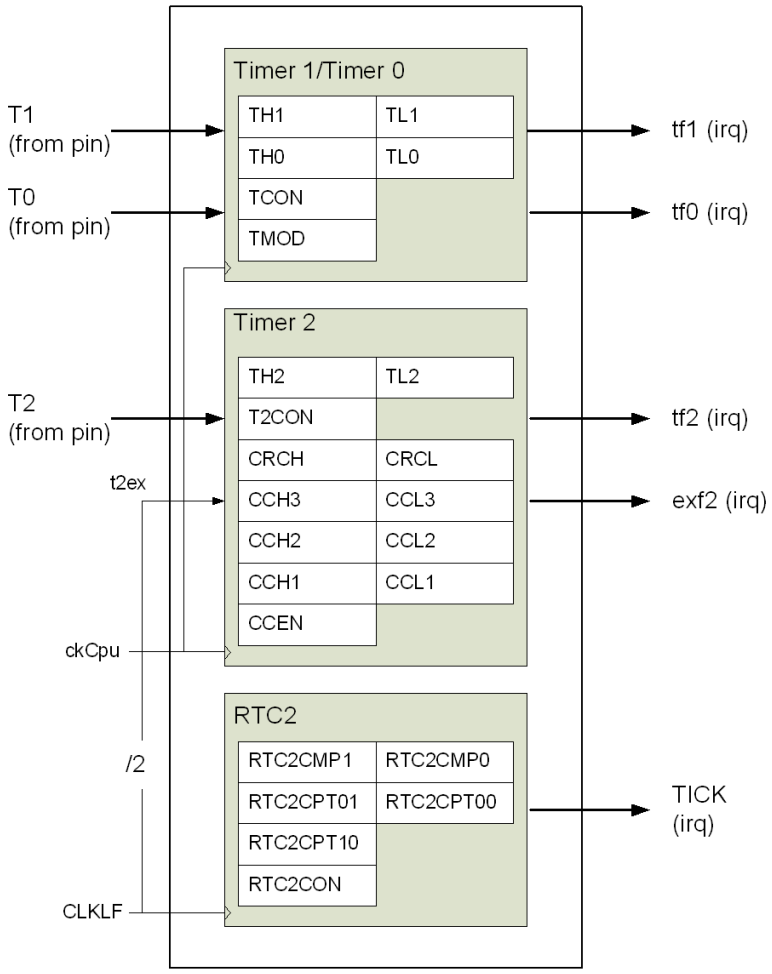



*Figure 36. Block diagram of timers/counters*

## 8.3 Functional description

### 8.3.1 Timer 0 and Timer 1

In timer mode, Timers 0 and 1 are incremented every 12 clock cycles.

In the counter mode, the Timers 0 and 1 are incremented when the falling edge is detected at the corresponding input pin T0 for Timer 0, or T1 for Timer 1.

**Note:** Timer input pins **T0**, **T1** and, **T2** must be configured as described in section 8.4 on page 86.

Since it takes two clock cycles to recognize a 1-to-0 event, the maximum input count rate is ½ of the oscillator frequency. There are no restrictions on the duty cycle, however to ensure proper recognition of 0 or 1 state, an input should be stable for at least 1 clock cycle.

Timer 0 and Timer 1 status and control are in TCON and TMOD register. The actual 16-bit Timer 0 value is in TH0 (8 msb) and TL0 (8 lsb), while Timer 1 uses TH1 and TL1.

Four operating modes can be selected for Timers 0 and 1. Two Special Function Registers, TMOD and TCON, are used to select the appropriate mode.

#### 8.3.1.1 Mode 0 and Mode 1

In mode 0, Timers 0 and 1 are each configured as a 13-bit register (TL0/TL1 = 5 bits, TH0/TH1 = 8 bits). The upper three bits of TL0 and TL1 are unchanged and should be ignored.

In mode 1 Timer 0 is configured as a 16-bit register.

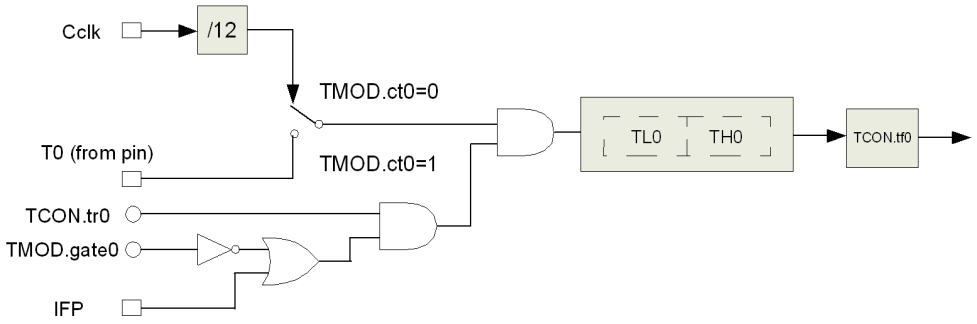



*Figure 37. Timer 0 in mode 0 and 1*

Likewise, in mode 1, Timer 1 is configured as a 16-bit register.

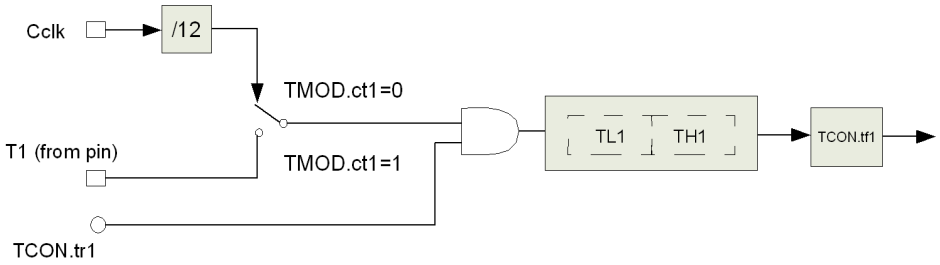



*Figure 38. Timer 1 in mode 0 and 1*

### 8.3.1.2 Mode 2

In this mode, Timers 0 and 1 are each configured as an 8-bit register with auto reload.

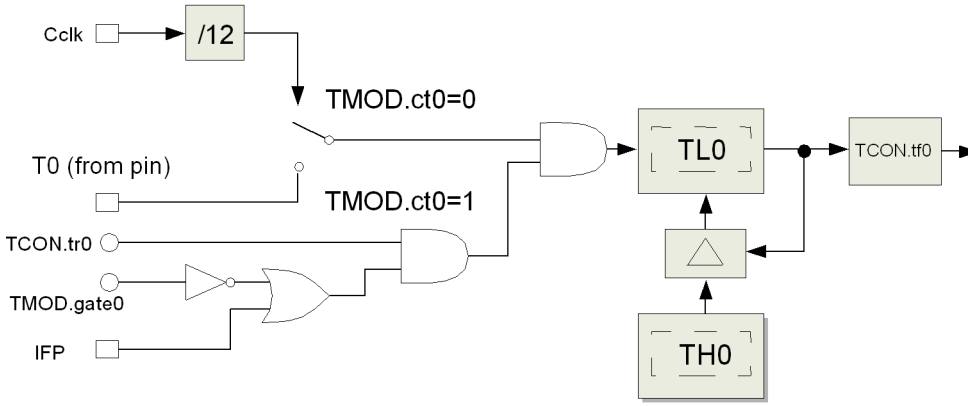



*Figure 39. Timer 0 in mode 2*

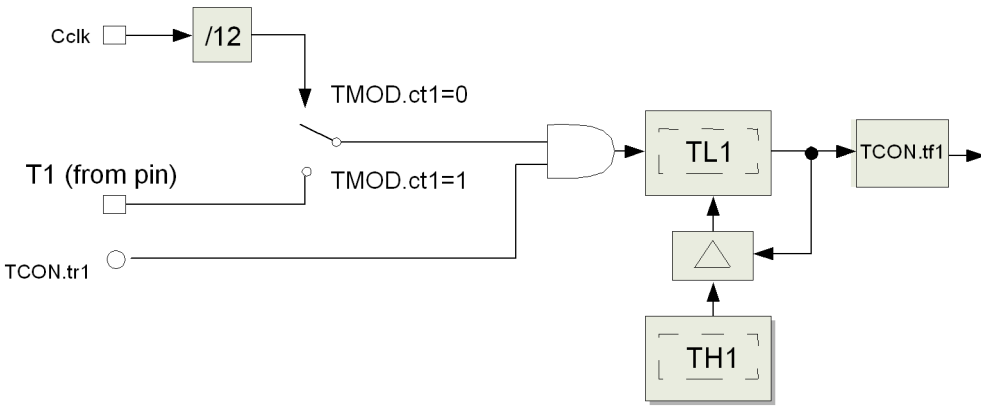



*Figure 40. Timer 1 in mode 2*

#### 8.3.1.3 Mode 3

In mode 3 Timers 0 and 1 are configured as one 8-bit timer/counter and one 8-bit timer, but timer 1 in this mode holds its count. When Timer 0 works in mode 3, Timer 1 can still be used in other modes by the serial port as a baud rate generator, or as an application not requiring an interrupt from Timer 1.

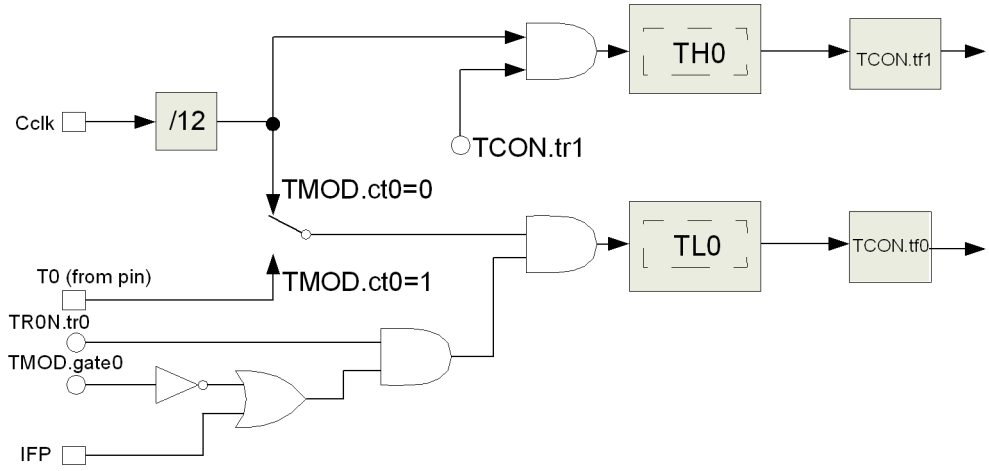



*Figure 41. Timer 0 in mode 3*

### 8.3.2 Timer 2

Timer 2 is controlled by `T2CON` while the value is in TH2 and TL2. Timer 2 also has four capture and one compare/reload registers which can read a value without pausing or reload a new 16-bit value when Timer 2 reaches zero, see chapter 8.4.7 on page 89 and chapter 8.4.8 on page 89.

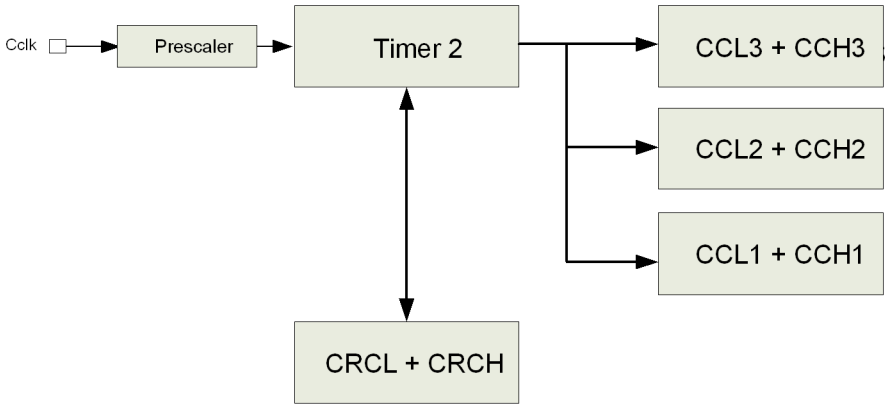



*Figure 42. Timer 2 block diagram*

#### 8.3.2.1 Timer 2 description

Timer 2 can operate as a timer, event counter, or gated timer.

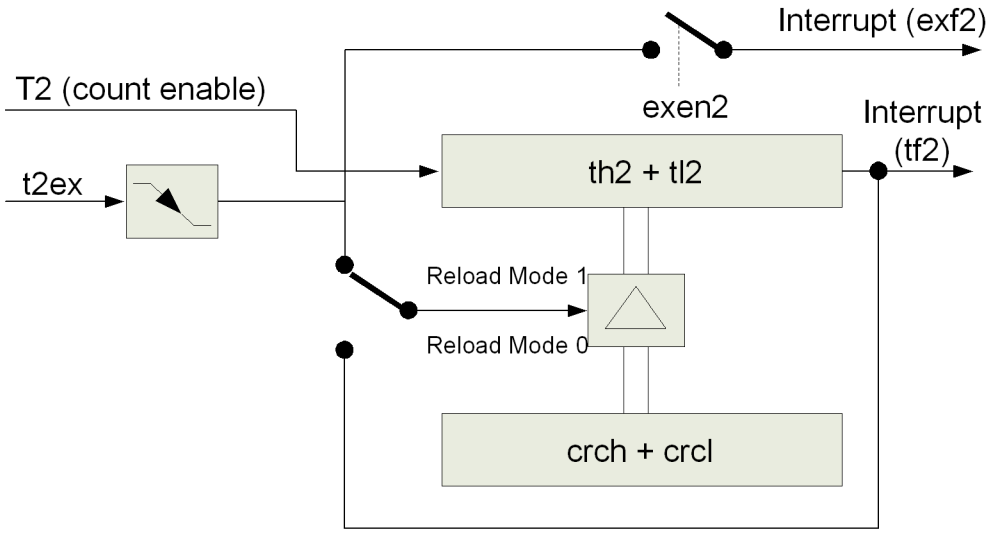



*Figure 43. Timer 2 in Reload Mode*

#### 8.3.2.2 Timer mode

Timer mode is invoked by setting the t2i0=1 and t2i1=0 in the `T2CON` register. In this mode, the count rate is derived from the clk input.

Timer 2 is incremented every 12 or 24 clock cycles depending on the 2:1 prescaler. The prescaler mode is selected by bit t2ps of `T2CON` register. When t2ps=0, the timer counts up every 12 clock cycles, otherwise every 24 cycles.

#### 8.3.2.3 Event counter mode

This mode is invoked by setting the t2i0=0 and t2i1=1 in the `T2CON` register.

In this mode, Timer 2 is incremented when external signal T2 (see section 8.4 on page 86 for more information on T2) changes its value from 1 to 0. The T2 input is sampled at every rising edge of the clock. Timer 2 is incremented in the cycle following the one in which the transition was detected. The maximum count rate is ½ of the clock frequency.

#### 8.3.2.4 Gated timer mode

This mode is invoked by setting the t2i0=1 and t2i1=1 in the `T2CON` register.

In this mode, Timer 2 is incremented every 12 or 24 clock cycles (depending on `T2CON` t2ps flag). Additionally, it is gated by the external signal T2. When T2=0, Timer 2 is stopped.

#### 8.3.2.5 Timer 2 reload

A 16-bit reload from the CRC register can be done in two modes:

- Reload Mode 0: Reload signal is generated by Timer 2 overflow (auto reload).
- Reload Mode 1: Reload signal is generated by negative transition at t2ex.

**Note:** t2ex is connected to an internal clock signal which is half frequency of CLKLF (see section 11 on page 100.)

## 8.4 SFR registers

### 8.4.1 Timer/Counter control register – TCON

TCON register reflects the current status of MCU Timer 0 and Timer 1 and it is used to control the operation of these modules.

| Address | Reset value | Bit | Name | Auto clear | Description |
|---|---|---|---|---|---|
| 0x88 | 0x00 | 7 | tf1 | Yes | Timer 1 overflow flag. Set by hardware when Timer1 over-flows. |
| | | 6 | tr1 | No | Timer 1 Run control. If cleared, Timer 1 stops. |
| | | 5 | tf0 | Yes | Timer 0 overflow flag. Set by hardware when Timer 0 over-flows. |
| | | 4 | tr0 | No | Timer 0 Run control. If cleared, Timer 0 stops. |
| | | 3 | ie1 | Yes | External interrupt 1 flag. Set by hardware. |
| | | 2 | it1 | No | External interrupt 1 type control. 1: falling edge, 0: low level |
| | | 1 | ie0 | Yes | External interrupt 0 flag. Set by hardware. |
| | | 0 | it0 | No | External interrupt 0 type control. 1: falling edge, 0: low level |

*Table 36. TCON register*

The tf0, tf1 (Timer 0 and Timer 1 overflow flags), ie0 and ie1 (external interrupt 0 and 1 flags) are automatically cleared by hardware when the corresponding service routine is called.

### 8.4.2 Timer mode register - TMOD

TMOD register is used for configuration of Timer 0 and Timer 1.

| Address | Reset value | Bit | Name | Description |
|---|---|---|---|---|
| 0x89 | 0x00 | 7 | gate1 | Timer 1 gate control |
| | | 6 | ct1 | Timer 1 counter/timer select. 1: Counter, 0: Timer |
| | | 5:4 | mode1 | Timer 1 mode<br>00 – Mode 0: 13-bit counter/timer<br>01 – Mode 1: 16-bit counter/timer<br>10 – Mode 2: 8-bit auto-reload timer<br>11 – Mode 3: Timer 1 stopped |
| | | 3 | gate0 | Timer 0 gate control |
| | | 2 | ct0 | Timer 0 counter/timer select. 1: Counter, 0: Timer |
| | | 1:0 | mode0 | Timer 0 mode<br>00 – Mode 0: 13-bit counter/timer<br>01 – Mode 1: 16-bit counter/timer<br>10 – Mode 2: 8-bit auto-reload timer<br>11 – Mode 3: two 8-bit timers/counters |

*Table 37. TMOD register*

### 8.4.3 Timer 0 – TH0, TL0

| Address | Register name |
|---|---|
| 0x8A | TL0 |
| 0x8C | TH0 |

*Table 38. Timer 0 register (TH0:TL0)*

These registers reflect the state of Timer 0. TH0 holds higher byte and TL0 holds lower byte. Timer 0 can be configured to operate as either a timer or a counter.

### 8.4.4 Timer 1 – TH1, TL1

| Address | Register name |
|---|---|
| 0x8B | TL1 |
| 0x8D | TH1 |

*Table 39. Timer 1 register (TH1:TL1)*

These registers reflect the state of Timer 1. TH1 holds higher byte and TL1 holds lower byte. Timer 1 can be configured to operate as either timer or counter.

### 8.4.5 Timer 2 control register – T2CON

T2CON register reflects the current status of Timer 2 and is used to control the Timer 2 operation.

| Address | Reset value | Bit | Name | Description |
|---|---|---|---|---|
| 0xC8 | 0x00 | 7 | t2ps | Prescaler select. 0: timer 2 is clocked with 1/12 of the ckCpu frequency. 1: timer 2 is clocked with 1/24 of the ckCpu frequency. |
| | | 6 | i3fr | Int3 edge select. 0: falling edge, 1: rising edge |
| | | 5 | i2fr | Int2 edge select: 0: falling edge, 1: rising edge |
| | | 4:3 | t2r | Timer 2 reload mode. 0X – reload disabled, 10 – Mode 0, 11 – Mode 1 |
| | | 2 | t2cm | Timer 2 compare mode. 0: Mode 0, 1: Mode 1 |
| | | 1:0 | t2i | Timer 2 input select. 00: stopped, 01: f/12 or f/24, 10: falling edge of T2, 11: f/12 or f/24 gated by T2. |

*Table 40. T2CON register*

### 8.4.6 Timer 2 – TH2, TL2

| Address | Register name |
|---|---|
| 0xCC | TL2 |
| 0xCD | TH2 |

*Table 41. Timer 2 (TH2:TL2)*

The TL2 and TH2 registers reflect the state of Timer 2. TH2 holds higher byte and TL2 holds lower byte. Timer 2 can be configured to operate in compare, capture or, reload modes.

### 8.4.7 Compare/Capture enable register – CCEN

The CCEN register serves as a configuration register for the Compare/Capture Unit associated with the Timer 2.

| Address | Reset value | Bit | Name | Description |
|---|---|---|---|---|
| 0xC1 | 0x00 | 7:6 | coca3 | compare/capture mode for CC3 register<br>00: compare/capture disabled<br>01: reserved<br>10: reserved<br>11: capture on write operation into register CCL3 |
| | | 5:4 | coca2 | compare/capture mode for CC2 register<br>00: compare/capture disabled<br>01: reserved<br>10: reserved<br>11: capture on write operation into register CCL2 |
| | | 3:2 | coca1 | compare/capture mode for CC1 register<br>00: compare/capture disabled<br>01: reserved<br>10: reserved<br>11: capture on write operation into register CCL1 |
| | | 1:0 | coca0 | compare/capture mode for CRC register<br>00: compare/capture disabled<br>01: reserved<br>10: compare enabled<br>11: capture on write operation into register CRCL |

*Table 42. CCEN register*

### 8.4.8 Capture registers – CC1, CC2, CC3

The Compare/Capture registers (CC1, CC2, CC3) are 16-bit registers used by the Compare/Capture Unit associated with the Timer 2. CCHn holds higher byte and CCLn holds lower byte of the CCn register.

| Address | Register name |
|---|---|
| 0xC2 | CCL1 |
| 0xC3 | CCH1 |
| 0xC4 | CCL2 |
| 0xC5 | CCH2 |
| 0xC6 | CCL3 |
| 0xC7 | CCH3 |

*Table 43. Capture Registers - CC1, CC2 and CC3*

### 8.4.9 Compare/Reload/Capture register – CRCH, CRCL

| Address | Reset value | Register name |
|---|---|---|
| 0xCA | 0x00 | CRCL |
| 0xCB | 0x00 | CRCH |

*Table 44. Compare/Reload/Capture register - CRCH, CRCL*

CRC (Compare/Reload/Capture) register is a 16-bit wide register used by the Compare/Capture Unit associated with Timer 2. CRCH holds higher byte and CRCL holds lower byte.

## 8.5 Real Time Clock - RTC

RTC2 contains two registers that can be used for capturing timer values; one loaded at positive edge of the 32.768 kHz clock and another register clocked by the MCU clock for better resolution. Both registers are updated as a consequence of an external event. RTC2 can also give an interrupt at predefined intervals due to value equality between the timer and a compare register. RTC2 ensures that the functions the interrupt is used for are awoken prior to the interrupt.

### 8.5.1 Features

- 32.768 kHz, sub-µA.
- 16-bit.
- Linear.
- Compare with IRQ (TICK). Resolution: 30.52 µs.
- Capture with increased resolution: 125 ns.

### 8.5.2 Functional description of SFR registers

The following registers control RTC2.

| Address (Hex) | Name/Mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| 0xB3 | RTC2CON | 4:0 | | R/W | RTC2 configuration register. |
| | *sfrCapture* | 4 | 0 | W | Trigger signal.<br>When the MCU writes a '1' to this register field, RTC2 will capture the timer value. The value is stored in RTC2CPT00 and RTC2CPT01. An additional counter clocked by the MCU clock will at this point contain the number of MCU clock cycles from the previous positive edge of the 32.768 kHz clock (edge detect @ MCU clock). The value is stored in RTC2CPT1. |

| Address (Hex) | Name/Mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| | *enableExternalCapture* | 3 | 0 | R/W | **1:** Timer value is captured if required by an IRQ from the Radio (edge detect @ MCU clock). The value is stored in RTC2CPT00 and RTC2CPT01. An additional counter clocked by the MCU clock will at this point contain the number of MCU clock cycles from the previous positive edge of the 32.768 kHz clock (edge detect @ MCU clock). The value is stored in RTC2CPT1.<br>**0:** Capture by Radio disabled. |
| | *compareMode* | 2:1 | 00 | R/W | Compare mode.<br>**11:** The Rtc2 IRQ is assigned when the timer value is equal to the concatenation of RTC2CMP1 and RTC2CMP0. RTC2 ensures that the functions for which the IRQ is intended, are all awoken prior to the Rtc2 IRQ. When the Rtc2 IRQ is assigned, the timer is reset.<br>**10:** Same as above, except that the Rtc2 IRQ will *not* reset the timer. The timer will always wrap around at overflow.<br>**0x:** Compare disabled. OK |
| | *rtc2Enable* | 0 | 0 | R/W | **1:** RTC2 is enabled. The clock to the RTC2 core functionality is running.<br>**0:** RTC2 is disabled. The clock to the RTC2 core functionality stands still and the timer is reset. |
| 0xB4 | RTC2CMP0 | 7:0 | 0xFF | R/W | RTC2 compare value register 0.<br>Contains LSByte of the value to be compared to the timer value to generate Rtc2 IRQ. Resolution: 30.52 µs. |
| 0xB5 | RTC2CMP1 | 7:0 | 0xFF | R/W | RTC2 compare value register 1.<br>Contains MSByte of the value to be compared to the timer value to generate Rtc2 IRQ. |
| 0xB6 | RTC2CPT00 | 7:0 | 0x00 | R | RTC2 capture value register 00.<br>Contains LSByte of the timer value at the time of the capture event. Resolution: 30.52 µs. |
| 0xAB | RTC2CPT01 | 7:0 | 0x00 | R | RTC2 capture value register 01.<br>Contains MSByte of the timer value at the time of the capture event. |
| 0xAC | RTC2CPT10 | 7:0 | 0x00 | R | RTC2 capture value register 1.<br>Contains the value of the counter that counts the number of MCU clock cycles from the previous positive edge of the 32.768 kHz clock until the capture event. The counter value is truncated by one bit (LSBit). Resolution: 125 ns. |

*Table 45. RTC2 register map*

The Rtc2 timer is a 16 bit timer counting from zero and upwards at the rate of the 32.768 kHz clock. When the Rtc2 timer is equal to the concatenation of RTC2CMP1 and RTC2CMP0, an Rtc2 IRQ, also referred to as TICK, is generated. There is an uncertainty of one CLKLF period, 30.52 µs, from when the Rtc2 is started or a new value is given to the RTC2 compare value registers and until the IRQ is given.

The time for the IRQ is given by the range:

$$\left[\frac{[\text{RTC2CMP1}:\text{RTC2CMP0}]-\text{timer}}{32768},\frac{[\text{RTC2CMP1}:\text{RTC2CMP0}]-\text{timer}+1}{32768}\right][\text{s}]$$

where [RTC2CMP1:RTC2CMP0] is the concatenation of RTC2CMP1 and RTC2CMP0 into a 16 bits word and **timer** is the current value of the Rtc2 timer when the RTC2 compare value register was updated or the Rtc2 enabled.

If compare mode 11 is used, the Rtc2 IRQ will be given every

$$\frac{[\text{RTC2CMP1}:\text{RTC2CMP0}]+1}{32768}[s]$$

second.

The RTC2 compare value is updated every time RTC2CMP1 or RTC2CMP0 is written. This might give unwanted behavior if precaution is not taken when updating any of the variables. When new values are written to RTC2CMP1 and RTC2CMP0, the Rtc2 IRQ should be disabled to prevent unwanted Rtc2 IRQ.

To make sure everything is up and running when the Rtc2 IRQ is given in Register retention or Memory retention timers on, the MCU is pre-started before the IRQ is given. If XOSC16M is enabled, the pre-start time is long enough to make sure that this clock is up and running before the IRQ is given[1]. If RCOSC16M is enabled by CLKCTRL[5:4], this will be the clock source in the pre-start period. To save power, the user could choose to go to Standby while waiting for the IRQ. If only RCOSC16M is enabled, the pre-start time is shorter, making sure that the RC-oscillator is up and running before the Rtc2 IRQ is given. This same, short pre-startup time is used from Register Retention to Active if XOSC16M is running while in Register retention[2]CLKCTRL[7] = 1.

This implies that the time from going to Register retention or Memory retention and until the Rtc2 IRQ is given, always must be longer then the pre-start time: 49 CLKLF periods for the long pre-start and 2 CLKLF for the short pre-start.

The Rtc2 counter uses the 32.768 kHz low frequency clock for the Rtc2 timer, and one of the 32.768 kHz sources must be enabled when using the Rtc2. See section 13.3 on page 116 for the 32.768 kHz clock.

Reading RTC2CMP0 and RTC2CMP1:

1. The crystal start-up time must be <1.5 ms to ensure that XOSC16M is clock source on arrival of the Rtc2 IRQ. Refer to section 13.3.1 on page 116 for further details.
2. To get the short pre-startup time when going to Register retention with XOSC16M running in the power down mode, make sure XOSC16M is running before going to Register retention. If it is not, the long pre-start time is used, and the minimum value for the long pre-startup for the RTC2 compare value register should be used. This apples only the first time going to Register retention after enabling XOSC16M in Register retention.

- Disable the Rtc2 IRQ, until both registers have been written.

Reading `RTC2CPT00`, `RTC2CPT01` and `RTC2CPT10`:

- Disable The Radio IRQ until all three registers have been read.

Uncertainty in capture values:

- 250 ns.

# 9 Interrupts

nRF24LE1 OTP has an advanced interrupt controller with 18 sources, as shown in Figure 44. The unit manages dynamic program sequencing based upon important real-time events as signalled from timers, the RF transceiver, pin activity, and so on.

## 9.1 Features

- Interrupt controller with 18 sources and 4 priority levels
- Interrupt request flags available
- Interrupt from pin (configurable)

## 9.2 Block diagram

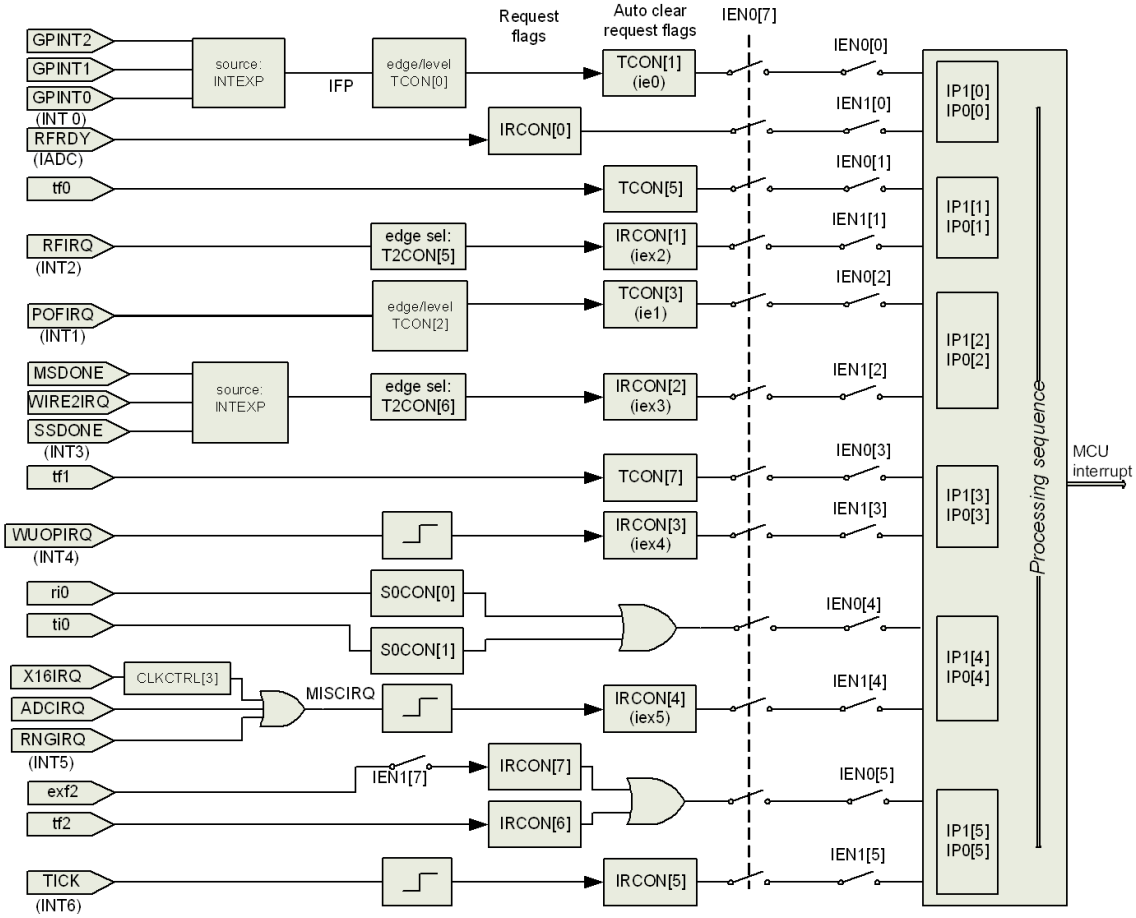



*Figure 44. Block diagram of interrupt structure*

## 9.3 Functional description

When an enabled interrupt occurs, the MCU vectors to the address of the interrupt service routine (ISR) associated with that interrupt, as listed in Table 46. The MCU executes the ISR to completion unless another interrupt of higher priority occurs.

| Source | vector | Polarity | Description |
|---|---|---|---|
| IFP | 0x0003 | low/fall | Interrupt from pin **GP INT0**, **GP INT1** or **GP INT2** as selected by bits 3,4 or 5 in SFR INTEXP. Only one of the bits may be set at a time. |
| tf0 | 0x000B | high | Timer 0 overflow interrupt |
| POFIRQ | 0x0013 | low/fall | Power Failure interrupt |
| tf1 | 0x001B | high | Timer 1 overflow interrupt |
| ri0 | 0x0023 | high | Serial channel receive interrupt |
| ti0 | 0x0023 | high | Serial channel transmit interrupt |
| tf2 | 0x002B | high | Timer 2 overflow interrupt |
| exf2 | 0x002B | high | Timer 2 external reload |
| RFRDY | 0x0043 | high | RF SPI ready |
| RFIRQ | 0x004B | fall/rise | RF interrupt |
| MSDONE | 0x0053 | fall/rise | Master SPI transaction completed |
| WIRE2IRQ | 0x0053 | fall/rise | 2-Wire transaction completed |
| SSDONE | 0x0053 | fall/rise | Slave SPI transaction completed |
| WUOPIRQ | 0x005B | rise[a] | Wakeup on pin interrupt |
| MISCIRQ | 0x0063 | rise | Miscellaneous interrupt is the sum of:<br>• XOSC16M started (X16IRQ)<br>• ADC Ready (ADCIRQ) interrupt<br>• RNG ready (RNGIRQ) interrupt |
| TICK | 0x006B | rise | Internal Wakeup (from RTC2) interrupt |

a. Polarity is always rise on interrupt, but pin polarity can be selected by bit 2 in the OPMCON register as described in Table 60. on page 109.

*Table 46. nRF24LE1 OTP interrupt sources*

**Note:** RFIRQ, WUOPIRQ, MISCIRQ and TICK are not activated unless wakeup is enabled by WUCON (see section 11.3.5 on page 110).

## 9.4 SFR registers

Various SFR registers are used to control and prioritize between different interrupts.

The TCON, IRCON, SCON, IP0, IP1, IEN0, IEN1 and INTEXP are described in this section. In addition the TCON and T2CON are used, the description for these registers can be found in chapter 8 on page 81.

### 9.4.1 Interrupt Enable 0 Register – IEN0

The IEN0 register is responsible for global interrupt system enabling/disabling and also Timer 0, 1 and 2, Port 0 and Serial Port individual interrupts enabling/disabling.

| Address | Bit | Description |
|---|---|---|
| 0xA8 | 7 | 1: Enable interrupts. 0: **all** interrupts are disabled |
| | 6 | Not used |
| | 5 | 1: Enable Timer2 (tf2/exf2) interrupt. |
| | 4 | 1: Enable Serial Port (ri0/ti0) interrupt. |
| | 3 | 1: Enable Timer1 overflow (tf1) interrupt |
| | 2 | 1: Enable Power failure (POFIRQ) interrupt |
| | 1 | 1: Enable Timer0 overflow (tf0) interrupt. |
| | 0 | 1: Enable Interrupt From Pin (IFP) interrupt. |

*Table 47. IEN0 register*

### 9.4.2 Interrupt Enable 1 Register – IEN1

The IEN1 register is responsible for RF, SPI and Timer 2 interrupts.

| Address | Bit | Description |
|---|---|---|
| 0xB8 | 7 | 1: Enable Timer2 external reload (exf2) interrupt |
| | 6 | Not used |
| | 5 | 1: Internal wakeup (TICK) interrupt enable |
| | 4 | 1: Miscellaneous (MISCIRQ) interrupt enable |
| | 3 | 1: Wakeup on pin (WUOPIRQ) interrupt enable |
| | 2 | 1: 2-Wire completed (WIRE2IRQ) interrupt, SPI master/slave completed (MSDONE/SSDONE) interrupt enable |
| | 1 | 1: RF (RFIRQ) interrupt enable |
| | 0 | 1: RF SPI ready (RFRDY) interrupt enable |

*Table 48. IEN1 register*

2-Wire Master SPI and Slave SPI share the same interrupt line.

| Address | Bit | Description | Reset value 0x01 |
|---|---|---|---|
| 0xA6 | 7:6 | not used | |
| | 5 | 1: Enable GP INT2 (from pin) to IFP | |
| | 4 | 1: Enable GP INT1 (from pin)1 to IFP | |
| | 3 | 1: Enable GP INT0 (from pin) 0 to IFP | |
| | 2 | 1: Enable 2-Wire completed (WIRE2IRQ) interrupt | |
| | 1 | 1: Enable Master SPI completed (MSDONE)interrupt | |
| | 0 | 1: Enable Slave SPI completed (SSDONE) interrupt | |

*Table 49. INTEXP register*

### 9.4.3 Interrupt Priority Registers – IP0, IP1

The 14 interrupt sources are grouped into six priority groups. For each of the groups, one of four priority levels can be selected. They can be selected by setting appropriate values in IP0 and IP1 registers.

The contents of the Interrupt Priority registers define the priority levels for each interrupt source according to the tables below.

| Address | Bit | Description |
|---|---|---|
| 0xA9 | 7:6 | Not used |
| | 5:0 | Interrupt priority. Each bit together with corresponding bit from IP1 register specifies the priority level of the respective interrupt priority group. |

*Table 50. IP0 register*

| Address | Bit | Description |
|---|---|---|
| 0xB9 | 7:6 | Not used |
| | 5:0 | Interrupt priority. Each bit together with corresponding bit from IP0 register specifies the priority level of the respective interrupt priority group. |

*Table 51. IP1 register*

| Group | Interrupt bits | Priority groups | | |
|---|---|---|---|---|
| 0 | IP1[0], IP0[0] | IFP | RFRDY | |
| 1 | IP1[1], IP0[1] | tf0 | RFIRQ | |
| 2 | IP1[2], IP0[2] | POFIRQ | MSDONE | SSDONE |
| 3 | IP1[3], IP0[3] | tf1 | WUOPIRQ | |
| 4 | IP1[4], IP0[4] | ri0 | ti0 | MISCIRQ |
| 5 | IP1[5], IP0[5] | tf2/exf2 | TICK | |

*Table 52. Priority groups*

| IP1.x | IP0.x | Priority level |
|---|---|---|
| 0 | 0 | Level 0 (lowest) |
| 0 | 1 | Level 1 |
| 1 | 0 | Level 2 |
| 1 | 1 | Level 3 (highest) |

*Table 53. Priority levels (x is the number of priority group)*

### 9.4.4 Interrupt Request Control Registers – IRCON

The IRCON register contains interrupt request flags.

| Address | Bit | Auto clear | Description |
|---|---|---|---|
| 0xC0 | 7 | No | Timer 2 external reload (exf2) interrupt flag |
| | 6 | No | Timer 2 overflow (tf2) interrupt flag |
| | 5 | Yes | Internal wakeup (TICK) interrupt flag |
| | 4 | Yes | Miscellaneous (MISCIRQ) interrupt flag |
| | 3 | Yes | Wakeup on pin (WUOPIRQ) interrupt flag |
| | 2 | Yes | 2-Wire completed (WIRE2IRQ), Master/Slave SPI (MSDONE/SSDONE) interrupt flag |
| | 1 | Yes | RF (RFIRQ) interrupt flag |
| | 0 | No | RF SPI ready (RFRDY) interrupt flag |

*Table 54. IRCON register*

# 10 Watchdog

The on-chip watchdog forces a system reset if the running software for some reason encounters a hang situation.

## 10.1 Features

- 32.768 kHz, sub-µA.
- 16-bit with an offset of 8 bits.
- Minimum Watchdog timeout interval: 7.8125 ms.
- Maximum Watchdog timeout interval: 512 s.
- Disable (reset) only by a system reset, or possibly when the chip enters the following power saving modes: Register retention and Memory retention. See section 17.3.2 on page 132 for details.

## 10.2 Block diagram

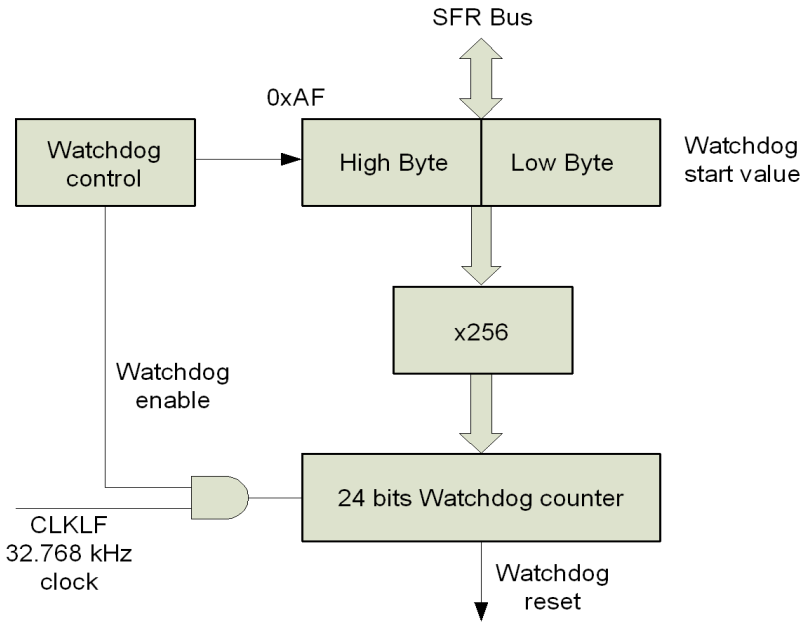



*Figure 45. Watchdog block diagram*

## 10.3 Functional description

The following register controls the Watchdog.

| Address (Hex) | Name/Mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| 0xAF | WDSV | 15:0 | 0x0000 | R/W | Watchdog start value register. MSByte and LSByte of the word are written and read as separate bytes. |

*Table 55. Watchdog register*

watchdogStartValue (WDSV) contains the upper 16 bits of the Watchdog counter's initial value. This 16 bits word is read and written as two separate bytes, LSByte and MSByte. LSByte is read and written first. After a write to WDSV, the next read of WDSV will always give the LSByte, and after a read, the next byte written will always be to the LSByte. In other words, to write to WDSV, two bytes must be written without a read between the writes, and vice-versa for read operations. Readout of WDSV will not give the current value of the Watchdog counter, but the start value for the counter.

After a reset, the default state of the Watchdog is disabled. The Watchdog is activated when both bytes of WDSV have been written. The Watchdog counter then counts down from WDSV*256 towards 0. When 0 is reached, the complete microcontroller, as well as the peripherals, are reset. A reset from the Watchdog willl have the same effect as a power-on reset or a reset from pin. To avoid the reset, the software must reload WDSV sufficiently often. The Watchdog counter is updated with a new start value and restarted every time WDSV is written.

The Watchdog counter uses the 32.768 kHz low frequency clock, and one of the 32.768 kHz sources must be enabled when using the Watchdog. See section 13.3 on page 116 for the 32.768 kHz clock.

The Watchdog timeout is given by:

$$\frac{WDSV \times 256}{32768}[s]$$

If WDSV is loaded with 0x0000, the maximum Watchdog timeout interval of 512 seconds is used, i.e. the Watchdog is not disabled.

If the Watchdog has been started, it can only be disabled (reset) by a system reset, or possibly when the chip enters the Register retention or Memory retention power-saving mode. Please refer to `OPMCON` bit 0 in Table 60. on page 107.

# 11 Power and clock management

The nRF24LE1 OTP Power Management function controls the power dissipation through administration of modes of operation and by controlling clock frequencies.

## 11.1 Block diagram

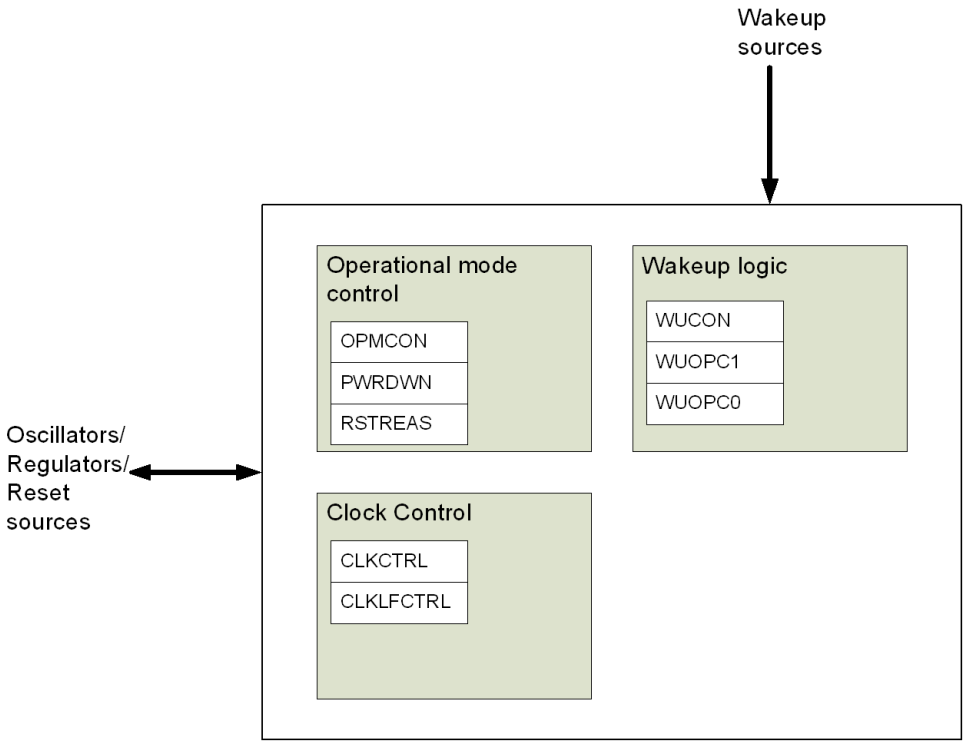



*Figure 46. Block diagram of power and clock management*

## 11.2 Modes of operation

After nRF24LE1 OTP is reset or powered on it enters active mode and the functional behavior is controlled by software. To enter one of the power saving modes, the `PWRDWN` register must be written with selected mode (as data).

To re-enter the active mode a wakeup source (valid for given power down mode) has to be activated.

The nRF24LE1 OTP modes of operation are summarized in the following table:

| Mode | Brief description |
|---|---|
| Deep Sleep[a] | Current:<br>See Table 114. on page 177<br><br>Powered functions:<br>• pins inclusive wakeup filter<br>Wakeup source(s):<br>From pin<br><br>Start-up time:<br>• < 100 µs when starting on RCOSC16M<br><br>Comment:<br>Wakeup from pin will in this mode lead to a system reset (after wakeup, program execution will start from the reset vector). |
| Memory retention, timers off | Current:<br>See Table 114. on page 177<br><br>Powered functions:<br>In addition to Deep Sleep:<br>• Power Manager<br>• IRAM and 512 bytes of data memory (DataRetentive SRAM)<br><br>Wakeup source(s):<br>From pin<br><br>Start-up time:<br>As for Deep Sleep<br><br>Comment:<br>Wakeup from pin will in this mode lead to a system reset. |

| Mode | Brief description |
|---|---|
| Memory retention, timers on | Current:<br>See Table 114. on page 177<br><br>Powered functions:<br>In addition to Memory retention, timers off:<br>• XOSC32K or RCOSC32K<br>• RTC2 and watchdog clocked on 32 KHz clock<br><br>Wakeup source(s):<br>From pin, wakeup TICK from timer or voltage level on pin (analog comparator wakeup)<br><br>Start-up time:<br>Wakeup from pin:<br>• < 100 µs when starting on RCOSC16M<br>Wakeup TICK:<br>• Pre-start voltage regulators and XOSC16M, system ready on RTC2 TICK. To save power, the user may choose to enter Standby power-down mode when the MCU system is awoken (<100 µs) and wait for TICK interrupt. A short pre-start time ( a few clock cycles) is used when XOSC16M is not enabled as controlled by CLKCTRL bit 5 and 4 (please refer to Table 57. on page 106.<br><br>Comment:<br>Wakeup will lead to system reset |
| Register retention, timers off[b] | Current:<br>See Table 114. on page 177<br><br>Powered functions:<br>In addition to Memory retention, timers on:<br>• All registers<br>• Rest of data memory (SRAM)<br>• Optional: XOSC16M<br><br>Wakeup source(s):<br>As for Memory retention, timers off. See also footnote for bit 7 regarding wakeup in Table 57. on page 106.<br><br>Start-up time:<br>As for memory retention, timers on.<br><br>Comment:<br>Wakeup does not lead to system reset (after wakeup, program execution will resume from the current instruction). |

| Mode | Brief description |
|---|---|
| Register retention, timers on[a] | Current:<br>See Table 114. on page 177<br><br>Powered functions:<br>In addition to Register retention, timers off:<br><br>Wakeup source(s):<br>As for Memory retention, timers on. See also footnote for bit 7 regarding wakeup in Table 57. on page 106.<br><br>Start-up time:<br>As for Register retention, timers off.<br>If awoken from TICK, a short pre-start time is used when XOSC16M is not enabled as controlled by CLKCTRL bit 5 and 4 (refer to Table 57. on page 106) or when XOSC16M is on in the Register retention mode (CLKCTRL bit 7). The short pre-start time will not be used if entering power-down before XOSC16M is running (this can be observed by polling bit 3 in CLKLFCTRL).<br><br>Comment:<br>Wakeup does not lead to system reset (after wakeup, program execution will resume from the current instruction). |
| Standby | Current:<br>See Table 114. on page 177<br><br>Powered functions:<br>In addition to Register retention:<br>• Program memory and Data memory<br>• VREG<br>• XOSC16M<br><br>Wakeup source(s):<br>In addition to Register retention:<br>• The interrupt sources RFIRQ and MISCIRQ (see section 9.3 on page 95 and 11.3.5 on page 110. Analog wakeup comparator is not supported in this mode.<br><br>Start-up time:<br>~ 100 ns<br><br>Comment:<br>Processor in standby, that is, clock stopped. I/O functions may be active. |

| Mode | Brief description |
|---|---|
| Active | Current:<br>See Table 114. on page 177<br><br>Powered functions:<br>Everything powered<br><br>Wakeup source(s):<br>-<br><br>Start-up time:<br>-<br><br>Comment:<br>Processor active and running |

a. The 16 MHz RC oscillator must be turned on before nRF24LE1 OTP enters Deep Sleep.
b. Please note that both Register retention power-down modes are entered by writing '100' to the PWRDWN register (refer to Table 59. on page 108). "Register retention timers on" is obtained by choosing an active CLKLF source as controlled by CLKLFCTRL[2:0] (refer to Table 58. on page 107).

*Table 56. Modes of operation*

## 11.3 Functional description

### 11.3.1 Clock control

The clock to the MCU (Cclk) is sourced from either an on-chip RC oscillator or a crystal oscillator (see chapter 13 on page 115) for details.

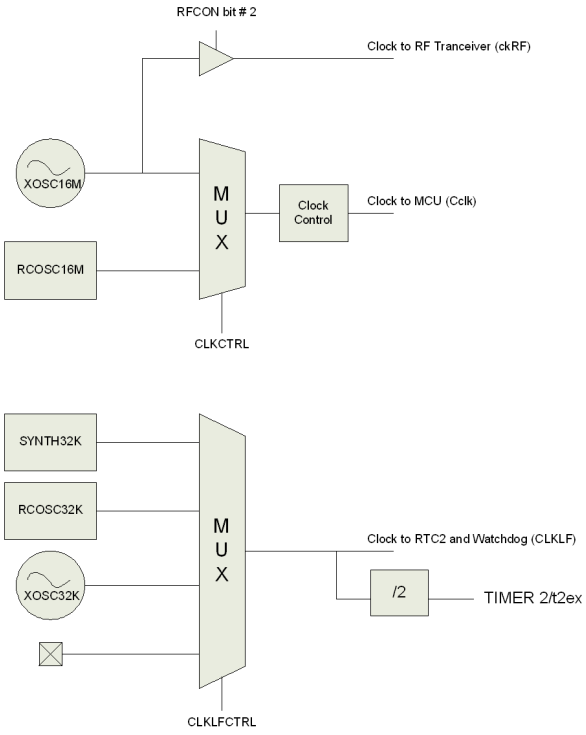



*Figure 47. nRF24LE1 OTP clock system*

The source and frequency of the clock to the microcontroller system is controlled by the CLKCTRL register:

| Addr | Bit | R/W | Function | Reset value: 0x00 |
|---|---|---|---|---|
| 0xA3 | 7[a] | R/W | 1: Keep XOSC16M on in Register retention mode | |
| | 6 | R/W | 1: Clock sourced directly from pin (XC1), bypass oscillators.[b]<br>0: Clock sourced by XOSC16M or RCOSC16M, see bit 3 | |
| | 5:4[c] | R/W | 00: Start both XOSC16M and RCOSC16M.[d]<br>01: Start RCOSC16M only.<br>10: Start XOSC16M only.[e]<br>11: Reserved | |
| | 3 | R/W | 1: Enable wakeup and interrupt (X16IRQ) from XOSC16M active<br>0: Disable wakeup and interrupt from XOSC16M active | |
| | 2:0 | R/W | Clock frequency to microcontroller system:<br>000: 16 MHz<br>001: 8 MHz<br>010: 4 MHz<br>011: 2 MHz<br>100: 1 MHz<br>101: 500 kHz<br>110: 250 kHz<br>111: 125 kHz | |

a. If this bit is set to '1', it is necessary to write '00' to bit 5:4 before entering Register retention mode. Otherwise, wakeup may not function properly.
b. For test and application development involving no radio link, only. Must be cleared in normal operation.
c. When writing to this bit, the device must be put in Register retention, Memory retention or Deep Sleep for effects to occur.
d. Default setting, both oscillators started. Clock sourced from RCOSC16M initially and automatically switched to XOSC16M.
e. The 16 MHz RC oscillator must be turned on before nRF24LE1 OTP enters Deep Sleep.

*Table 57. CLKCTRL register*

The source of the 32 kHz clock (CLKLF) is controlled by the CLKLFCTRL register:

| Addr | Bit | R/W | Function | Reset value: 0x07 |
|---|---|---|---|---|
| 0xAD | 7 | R | 1: Read CLKLF (phase). | |
| | 6 | R | 1: CLKLF ready to be used | |
| | 5 | - | Reserved | |
| | 4 | - | Reserved | |
| | 3 | R | 1: Clock sourced by XOSC16M (that is, XOSC16M active/running)<br>0: Clock sourced by RCOSC16M | |
| | 2:0 | R/W | Source for CLKLF:<br>000: XOSC32K[a]<br>001: RCOSC32K<br>010: Synthesized from XOSC16M when active, off otherwise[b]<br>011: From IO pin used as XC1 to XOSC32K (low amplitude signal)<br>100: From IO pin (digital rail-to-rail signal)<br>101: Reserved<br>110: Reserved<br>111: None selected | |

a. P0.0/P0.1 must be set to inputs and the pinMode input mode bits (refer to Table 81. on page 138) must be set to Digital input buffer off (P0CON[6:5]=11).
b. XOSC16M will be stopped in Deep Sleep and Memory Retention, and therefore, stopping CLKLF in these modes of operation.

*Table 58. CLKLFCTRL register*

**Note:** If a source for CLKLF is selected, the MCU system will not start unless CLKLF is operative. For example, when selecting CLKLF from IO pin the external clock must be active for the MCU to wake up by pin from memory retention.

### 11.3.2 Power down control – PWRDWN

The PWRDWN register is used by the MCU to set the system to a power saving mode:

| Addr | Bit | R/W | Function | Reset value: 0x00 |
|---|---|---|---|---|
| 0xA4 | 7 | R | Indicates a wakeup from pin if set<br>This bit is either cleared by a read or by entering a power down mode | |
| | 6 | R | Indicates a wakeup from TICK if set<br>This bit is either cleared by a read or by entering a power down mode | |
| | 5 | R | Indicates a wakeup from Comparator if set<br>This bit is either cleared by a read or by entering a power down mode | |
| | 4:3 | | Reserved | |
| | 2:0 | W | Set system to power down if different from 000<br>001: set system to DeepSleep<br>010: set system to Memory retention, timer off<br>011: set system to Memory retention, timer on<br>100: set system to Register retention<br>101: reserved<br>110: reserved<br>111: set system to standby (stop MCU clock) | |
| | | R | Shows previous power down mode<br>000: Power off<br>001: DeepSleep<br>010: Memory retention, timer off<br>011: Memory retention, timer on<br>100: Register retention<br>101: reserved<br>110: reserved<br>111: standby | |

**Note:** On wakeup from powerdown, the `PWRDWN` register should be reset to 0x00. The content of the `PWRDWN` register can be read out first if needed.

*Table 59. PWRDWN register*

### 11.3.3 Operational mode control - OPMCON

The OPMCON register is used to control special behavior in some of the operation modes:

| Addr | Bit | R/W | Function | Reset value: 0x00 |
|---|---|---|---|---|
| 0xAE | 7:3 | - | Reserved (always write '0' to these bits) | |
| | 2 | R/W | 1: Subset of wakeup pins have active low polarity<br>0: All wakeup pins have active high polarity.<br>Refer to section 11.3.6 on page 110. | |
| | 1 | R/W | Retention latch control<br>0: Latch open – pass through<br>1: Latch locked<br>To keep some internal chip setup, such as pin directions/setup, you need to lock a set of retention latches before entering DeepSleep and memory retention power saving modes. After a wake up you must re-establish the register settings before opening the retention latches. | |
| | 0 | R/W | Watchdog reset enable<br>0: If the on-chip watchdog functionality is enabled it will keep running as long the operational mode Deep Sleep is not entered.<br>1: The on-chip watchdog functionality will enter its reset state when the operational mode Memory Retention and Register Retention is entered. | |

*Table 60. OPMCON register*

**Note:** If the Watchdog reset enable bit is enabled, you must wait at least until the first negative edge of CLKLF after enabling CLKLF before proceeding to Register Retention or Memory Retention. Waiting for the negative edge of CLKLF is not needed when entering into any other power-down state or if the Watchdog reset enable bit is not enabled.

### 11.3.4 Reset result – RSTREAS

There are four reset sources that initiate the same reset/ start-up sequence. These are:

- Reset from the on chip reset generator
- Reset from pin
- Reset generated from the on chip watchdog function

The RSTREAS register stores the reason for the last reset, all cleared indicates that the last reset was from the on-chip reset generator. A write operation to the register will clear all bits. Unless cleared after read (by on-chip reset or by a write operation), RSTREAS will be cumulative.

| Addr | Bit | R/W | Function |
|---|---|---|---|
| 0xB1 | 7:3 | - | Not used |
| | 2:0 | R | 000: On-chip reset generator<br>001: RST pin<br>010: Watchdog |

*Table 61. RSTREAS register*

### 11.3.5 Wakeup configuration register – WUCON

The following wakeup sources is available in STANDBY power down mode.

| Addr | Bit | R/W | Function | Reset value 0x00 |
|---|---|---|---|---|
| 0xA5 | 7:6 | RW | 00: Enable wakeup on RFIRQ if interrupt is enabled (IEN1.1=1)<br>01: Reserved, not used<br>10: Enable wakeup on RFIRQ<br>11: Ignore RFIRQ | |
| | 5:4 | RW | 00: Enable wakeup on TICK (from RTC2) if interrupt is enabled (IEN1.5=1)<br>01: Reserved, not used<br>10: Enable wakeup on TICK<br>11: Ignore TICK | |
| | 3:2 | RW | 00: Enable wakeup on WUOPIRQ if interrupt is enabled (IEN1.3=1)<br>01: Reserved, not used<br>10: Enable wakeup on WUOPIRQ<br>11: Ignore WUOPIRQ | |
| | 1:0 | RW | 00: Enable wakeup on MISCIRQ if interrupt is enabled (IEN1.4=1)<br>01: Reserved, not used<br>10: Enable wakeup on MISCIRQ<br>11: Ignore MISCIRQ | |

*Table 62. WUCON register*

MISCIRQ is set if one of the following take place:

- XOSC16M has started and is ready to be used.
- ADC finished with conversion, and data ready.
- RNG finished and a new random number is ready

### 11.3.6 Pin wakeup configuration

Pin wakeup is configured by two registers, `WUOPC1` and `WUOPC2`

| Address (Hex) | Name/Mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| 0xCE | `WUOPC1` | 7:0 | 0x00 | R/W | Wake Up On Pin configuration register 1.<br>n = 1: Wake up on pin enabled.<br>n = 0: Wake up on the corresponding pin disabled. |
| 0xCF | `WUOPC0` | 7:0 | 0x00 | R/W | Wake Up On Pin configuration register 0.<br>n = 1: Wake up on pin enabled.<br>n = 0: Wake up on the corresponding pin disabled. |

*Table 63. WUOPCx registers*

The function for the WUOPCx registers depends on selected package. The following table shows which port-pin/ gpio that give wakeup if the corresponding enable bit in the WUOPCx register is asserted for each

nRF24LE1 OTP package variant. Pins marked with an asterisk have selectable polarity controlled by OPMCON[2]. All other pins have high polarity.

| WUOPC bit | nRF24LE1-Q48 wakeup pins | nRF24LE1-Q32 wakeup pins | nRF24LE1-Q24 wakeup pins |
|---|---|---|---|
| WUOPC1[7] | P1.7 | Not used | Not used |
| WUOPC1[6] | P3.6 | P1.6 | Not used |
| WUOPC1[5] | P3.5 | P1.5 | Not used |
| WUOPC1[4] | P3.4 | P1.4* | Not used |
| WUOPC1[3] | P3.3 | P1.3 | Not used |
| WUOPC1[2] | P3.2 | P1.2* | Not used |
| WUOPC1[1] | P3.1 | P1.1 | Not used |
| WUOPC1[0] | P3.0 | P1.0 | Not used |
| WUOPC0[7] | P2.7 | P0.7 | Not used |
| WUOPC0[6] | P2.6* | P0.6* | P0.6* |
| WUOPC0[5] | P2.5 | P0.5 | P0.5 |
| WUOPC0[4] | P2.4 | P0.4 | P0.4 |
| WUOPC0[3] | P2.3 | P0.3 | P0.3 |
| WUOPC0[2] | P2.2* | P0.2 | P0.2 |
| WUOPC0[1] | P2.1 | P0.1 | P0.1 |
| WUOPC0[0] | P2.0 | P0.0 | P0.0 |

*Table 64. Configuration of pin wakeup*

If the SPI Slave function is enabled, that is, bit 0 in the SPISCON0 register is set, the spiSlaveCsn signal becomes an active low pin wakeup source.

# 12 Power supply supervisor

The power supply supervisor initializes the system at power-on, provides an early warning of impending power failure, and puts the system in reset state if the supply voltage is too low for safe operation.

## 12.1 Features

- Power-on reset with timeout delay
- Brown-out reset
- Power-fail warning with programmable threshold, interrupt and hardware protection of data in program memory

## 12.2 Block diagram

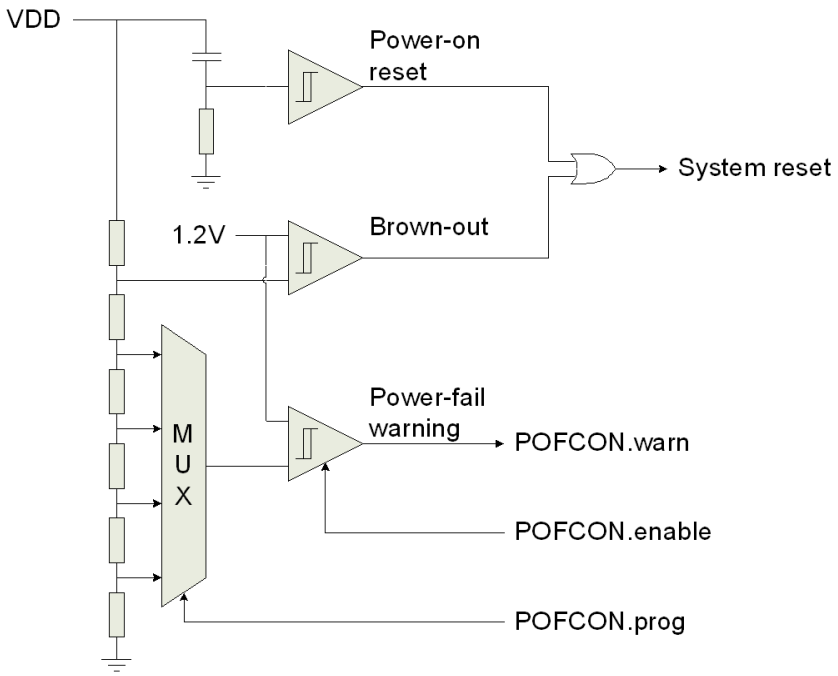



*Figure 48. Block diagram of power supply supervisor*

## 12.3 Functional description

### 12.3.1 Power-on reset

The Power-On Reset (POR) generator initializes the system at power-on. It is based on an RC network and a comparator, as illustrated in Figure 48. For proper operation the supply voltage should rise monotonically with rise time according to the specifications in Table 111. on page 172 . The system is held in reset state for at least 1ms after the supply has reached the minimum operating voltage of 1.9V.

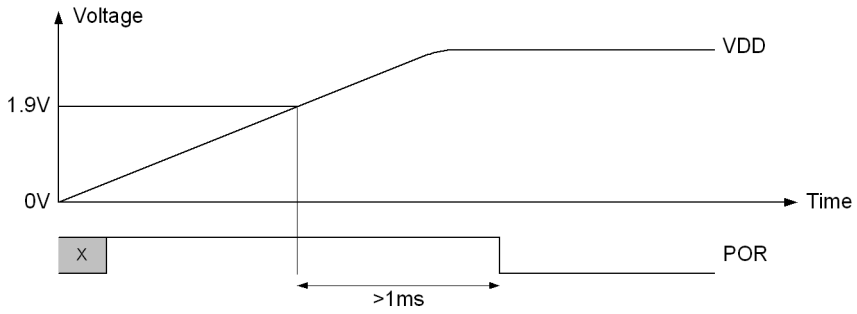



*Figure 49. Power-on reset*

### 12.3.2 Brown-out reset

The Brown-Out Reset (BOR) generator puts the system in reset state if the supply voltage drops below the BOR threshold, which is about 1.7V. There is approximately 70mV of hysteresis ($V_{HYST}$). This means that if a reset is triggered when the supply voltage drops below 1.7V, the supply must rise above 1.77V again before the nRF24LE1 OTP becomes operational. Hysteresis prevents the comparator output from oscillating when VDD is close to threshold. The BOR generator is enabled when the system is in active or standby mode.

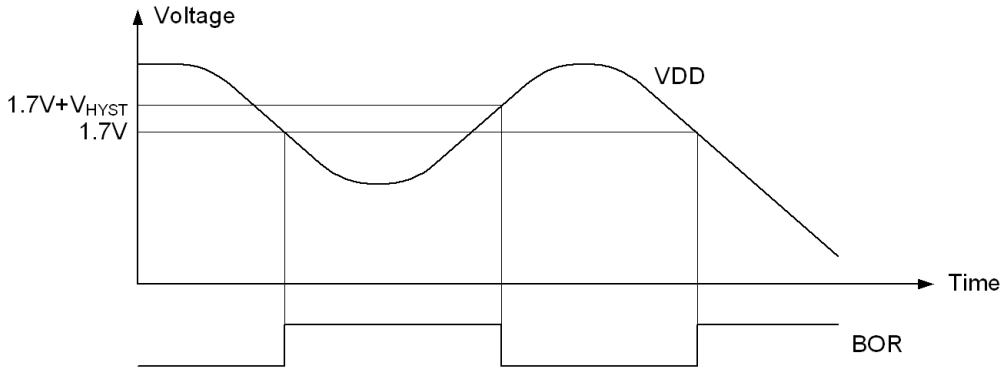



*Figure 50. Brown-Out Reset*

### 12.3.3 Power-fail comparator

The Power-Fail (POF) comparator provides the MCU with an early warning of impending power failure. It will not reset the system, but gives the MCU time to prepare for an orderly power-down. It also provides hardware protection of data stored in program memory, by preventing write instructions from being executed. Refer to section 6.3.3 on page 73 for details.

The POF comparator is enabled or disabled by writing the **enable** bit in the `POFCON` register (see Table 65. on page 114). When enabled, it will be powered up when the system is in active or standby mode. The **warn** bit is set to '1' if the supply voltage is below the programmable threshold. An interrupt (`POFIRQ`) is also produced. Write instructions to program memory will not be executed as long as **warn** is '1'.

Use the **prog** bits to configure the desired threshold voltage ($V_{POF}$). The available levels are 2.1, 2.3, 2.5 and 2.7V, defined for falling supply voltage. The comparator has approximately 0.1V of hysteresis ($V_{HYST}$).

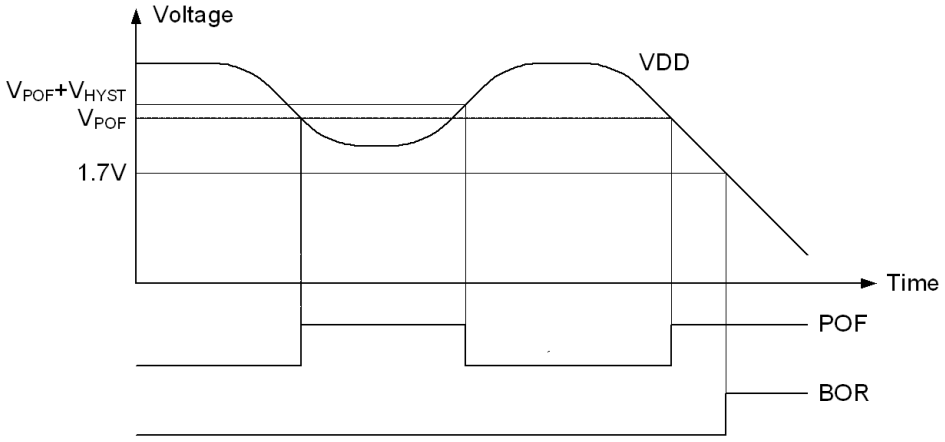



*Figure 51. Power-fail comparator*

## 12.4 SFR registers

| Addr | Bit | Name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| 0xDC | 7 | enable | RW | POF enable:<br>0: Disable POF comparator<br>1: Enable POF comparator | |
| | 6:5 | prog | RW | POF threshold:<br>00: 2.1V<br>01: 2.3V<br>10: 2.5V<br>11: 2.7V | |
| | 4 | warn | R | POF warning:<br>0: VDD above threshold<br>1: VDD below threshold | |
| | 3:0 | - | - | Not used | |

*Table 65. POFCON register*

# 13 On-chip oscillators

The nRF24LE1 OTP contains two high frequency oscillators and two low frequency oscillators. The primary high frequency clock source is a 16 MHz crystal oscillator. There is also a fast starting 16 MHz RC oscillator, which is used primarily to provide the system with a high frequency clock while it is waiting for the crystal oscillator to start up. The low frequency clock can be supplied by either a 32.768 kHz crystal oscillator or a 32.768 kHz RC oscillator. External 16 MHz and 32.768 kHz clocks may also be used instead of the on-chip oscillators. See section 11.3.1 on page 105 for control of the clock sources.

## 13.1 Features

- Low-power amplitude regulated 16 MHz crystal oscillator
- Fast starting 16 MHz RC oscillator with ±5% frequency accuracy
- Ultra low-power amplitude regulated 32.768 kHz crystal oscillator
- Ultra low-power 32.768 kHz RC oscillator with ±10% frequency accuracy

## 13.2 Block diagrams

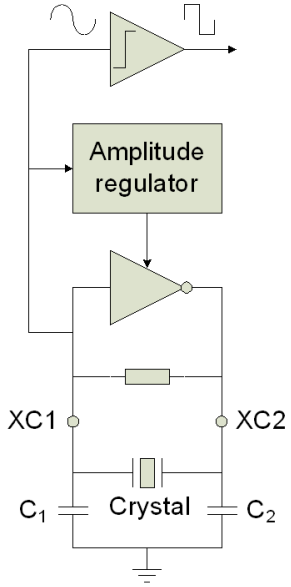



*Figure 52. Block diagram of 16 MHz crystal oscillator*

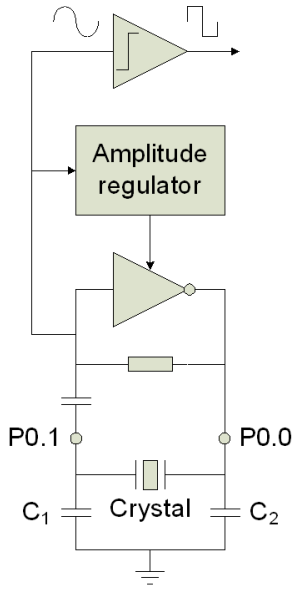



*Figure 53. Block diagram of 32.768 kHz crystal oscillator*

## 13.3 Functional description

### 13.3.1 16 MHz crystal oscillator

The 16 MHz crystal oscillator (XOSC16M) is designed to be used with an AT-cut quartz crystal in parallel resonant mode. To achieve correct oscillation frequency it is very important that the load capacitance matches the specification in the crystal datasheet. The load capacitance is the total capacitance seen by the crystal across its terminals:

$$C_{LOAD} = \frac{C_1' \cdot C_2'}{C_1' + C_2'}$$

$$C_1' = C_1 + C_{PCB1} + C_{PIN}$$

$$C_2' = C_2 + C_{PCB2} + C_{PIN}$$

$C_1$ and $C_2$ are ceramic SMD capacitors connected between each crystal terminal and VSS, $C_{PCB1}$ and $C_{PCB2}$ are stray capacitances on the PCB, while $C_{PIN}$ is the input capacitance on the **XC1** and **XC2** pins of the nRF24LE1 OTP (typically 1pF). $C_1$ and $C_2$ should be of the same value, or as close as possible.

To ensure a functional radio link the frequency accuracy must be ±60 ppm or better. The initial tolerance of the crystal, drift over temperature, aging and frequency pulling due to incorrect load capacitance must all be taken into account. For reliable operation the crystal load capacitance, shunt capacitance, equivalent series resistance (ESR) and drive level must comply with the specifications in Table 113. on page 177. It is recommended to use a crystal with lower than maximum ESR if the load capacitance and/or shunt capacitance is high. This will give faster start-up and lower current consumption.

The start-up time is typically about 1 ms for a crystal with 9 pF load capacitance and an ESR specification of 60 Ω max. This value is valid for crystals in a 3.2×2.5 mm can. If you use the smallest crystal cans (like 2.0×2.5 mm), pay particular attention to the startup time of the crystal. These crystals have a longer startup than crystals in larger cans. To make sure the startup time is <1.5 ms use a crystal for load capacitance of 6pF. A low load capacitance will reduce both startup time and current consumption. For more details regarding how to measure the startup of a specific crystal, please see the nAN24-13 application note.

The crystal oscillator is normally running only when the system is in active or standby mode. It is possible to keep it on in register retention mode as well, by writing a '1' to bit 7 in the CLKCTRL register (see Table 57. on page 106). This is recommended if the system is expected to wake up again in less than 5 ms. The reason is that the additional current drawn during start-up makes it more power-efficient to let the oscillator run for a few extra milliseconds than to restart it.

### 13.3.2 16 MHz RC oscillator

The 16 MHz RC oscillator (RCOSC16M) is used primarily to provide a high speed clock while the crystal oscillator is starting up. It starts in just a few microseconds, and has a frequency accuracy of ±5%.

By default, the 16 MHz RC and crystal oscillators are started simultaneously. The RC oscillator supplies the clock until the crystal oscillator has stabilized. The system then makes an automatic switch to the crystal oscillator clock, and turns off the RC oscillator to save power. Bit 3 in the CLKCTRL register can be polled to check which oscillator is currently supplying the high speed clock.

The system can be configured to start only one of the two 16 MHz oscillators. Write bit 4 and 5 in the CLKCTRL register to choose the desired behavior. Note that the RF transceiver cannot be used while the high frequency clock is sourced by the RC oscillator. The ADC may also have reduced performance.

### 13.3.3 External 16 MHz clock

The nRF24LE1 OTP may be used with an external 16MHz clock applied to the **XC1** pin. Write a '1' to bit 6 in the CLKCTRL register if the external clock is a rail-to-rail digital signal. (This if for test and application development involving no radio link, only. Bit 6 must be cleared in normal operation.) The input signal may also be analog, coming from e.g. the crystal oscillator of a microcontroller. In this case the crystal oscillator on the nRF24LE1 OTP must also be enabled, since it is used to convert the analog input into a digital clock signal. CLKCTRL[6] must be '0', and CLKCTRL[5:4] must be '10' to enable the oscillator. An input amplitude of 0.8V peak-to-peak or higher is recommended to achieve low current consumption and a good signal-to-noise ratio. The DC level is not important as long as the applied signal never rises above VDD or drops below VSS. The **XC1** pin will load the microcontrollers crystal with approximately 1pF in addition to PCB routing. **XC2** shall not be connected.

**Note:** A frequency accuracy of ±60 ppm or better is required to get a functional radio link.

### 13.3.4 32.768 kHz crystal oscillator

The 32.768 kHz crystal oscillator (XOSC32K) is operational in all system modes except deep sleep and memory retention, timer off. It is enabled by writing '000' to CLKCTRL[2:0].

A crystal must be connected between port pins **P0.0** and **P0.1**, which are automatically configured as crystal pins when the oscillator is enabled. To achieve correct oscillation frequency it is important that the load capacitance matches the specification in the crystal datasheet. The load capacitance is the total capacitance seen by the crystal across its terminals:

$$C_{LOAD} = \frac{C_1^{'} \cdot C_2^{'}}{C_1^{'} + C_2^{'}}$$
$$C_1^{'} = C_1 + C_{PCB1} + C_{PIN}$$
$$C_2^{'} = C_2 + C_{PCB2} + C_{PIN}$$

$C_1$ and $C_2$ are ceramic SMD capacitors connected between each crystal terminal and VSS, $C_{PCB1}$ and $C_{PCB2}$ are stray capacitances on the PCB, while $C_{PIN}$ is the input capacitance on the **P0.0** and **P0.1** pins of the nRF24LE1 OTP (typically 3pF when configured as crystal pins). $C_1$ and $C_2$ should be of the same value, or as close as possible. The oscillator uses an amplitude regulated design similar to the 16 MHz crystal oscillator. For reliable operation the crystal load capacitance, shunt capacitance, equivalent series resistance (ESR) and drive level must comply with the specifications in Table 113. on page 177. It is recommended to use a crystal with lower than maximum ESR if the load capacitance and/or shunt capacitance is high. This will give faster start-up and lower current consumption.

The start-up time is typically less than 0.5s for a crystal with 9pF load capacitance, 1pF shunt capacitance and an ESR of 50kΩ. Bit 6 in the CLKLFCTRL register can be polled to check if the oscillator is ready for use.

### 13.3.5 32.768 kHz RC oscillator

The low frequency clock may be generated by a 32.768 kHz RC oscillator (RCOSC32K) instead of the crystal oscillator, if a frequency accuracy of ±10% is sufficient. This saves the cost of a crystal, and also frees up **P0.0** and **P0.1** for other applications. The 32.768 kHz RC oscillator is enabled by writing '001' to CLKLFCTRL[2:0]. It typically starts in less than 0.5 ms. Bit 6 in the CLKLFCTRL register can be polled to check if the oscillator is ready for use.

### 13.3.6 Synthesized 32.768 kHz clock

The low frequency clock can also be synthesized from the 16 MHz crystal oscillator clock. Write '010' to CLKCTRL[2:0] to select this option. The synthesized clock will only be available in system modes where the 16 MHz crystal oscillator is active. (This will be possible in the operational modes "Register retention," "Standby," and "Active.")

### 13.3.7 External 32.768 kHz clock

The nRF24LE1 OTP may be used with an external 32.768 kHz clock applied to the **P0.1** port pin. Write '100' to CLKCTRL[2:0] if the external clock is a rail-to-rail digital signal, or '011' if it is an analog signal coming from e.g. the crystal oscillator of a microcontroller. An analog input signal must have an amplitude of 0.2V peak-to-peak or higher. The DC level is not important as long as the applied signal never rises above VDD or drops below VSS. The **P0.1** port pin will load the microcontrollers crystal with approximately 3pF in addition to PCB routing.

# 14 MDU – Multiply Divide Unit

The MDU – Multiplication Division Unit, is an on-chip arithmetic co-processor which enables the MCU to perform additional extended arithmetic operations like 32-bit division, 16-bit multiplication, shift and, normalize operations.

## 14.1 Features

The MDU is controlled by the SFR registers MD0.. MD5 and ARCON.

## 14.2 Block diagram

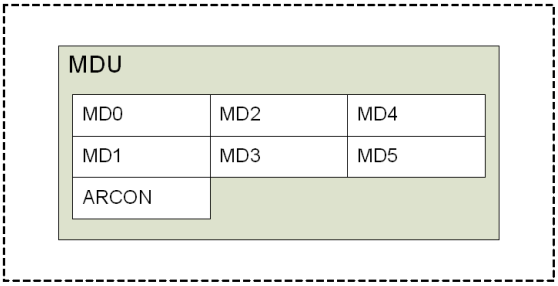



*Figure 54. Block diagram of MDU*

## 14.3 Functional description

All operations are unsigned integer operations. The MDU is handled by seven registers, which are memory mapped as Special Function Registers. The arithmetic unit allows concurrent operations to be performed independent of the MCU's activity.

Operands and results are stored in MD0.. MD5 registers. The module is controlled by the ARCON register. Any calculation of the MDU overwrites its operands.

The MDU does not allow reentrant code and cannot be used in multiple threads of the main and interrupt routines at the same time. Use the NOMDU_R515 compile directive to disable MDU operation in possible conflicting functions.

## 14.4 SFR registers

The MD0.. MD5 are registers used in the MDU operation.

| Address | Register name |
|---|---|
| 0xE9 | MD0 |
| 0xEA | MD1 |
| 0xEB | MD2 |
| 0xEC | MD3 |
| 0xED | MD4 |
| 0xEE | MD5 |

*Table 66. Multiplication/Division registers MD0..MD5*

The ARCON register controls the operation of MDU and informs you about its current state.

| Address | Reset value | Bit | Name | Description |
|---|---|---|---|---|
| 0xEF | 0x00 | 7 | mdef | MDU Error flag MDEF. Indicates an improperly performed operation (when one of the arithmetic operations has been restarted or interrupted by a new operation). |
| | | 6 | mdov | MDU Overflow flag MDOV. Overflow occurrence in the MDU operation. |
| | | 5 | slr | Shift direction, 0: shift left, 1: shift right. |
| | | 4:0 | sc | Shift counter. When set to '0's, normalize operation is selected. After normalization, the "sc.4" … "sc.0" contains the number of normalizing shifts performed.<br>Shift operation is selected when at least one of these bits is set high. The number of shifts performed is determined by the number written to "sc.4".., "sc.0", where "sc.4" is the MSB. |

*Table 67. ARCON register*

The operation of the MDU consists of the following phases:

### 14.4.1 Loading the MDx registers

The type of calculation the MDU has to perform is selected in accordance with the order in which the MDx registers are written.

| Operation | 32 bit/16 bit | | 16 bit / 16 bit | | 16 bit x 16 bit | | Shift/normalize | |
|---|---|---|---|---|---|---|---|---|
| first write | MD0 (lsb)<br>MD1<br>MD2<br>MD3 (msb) | Dividend | MD0 (lsb)<br>MD1 (msb) | Dividend | MD0 (lsb) | Num1 | MD0 (lsb)<br>MD1<br>MD2<br>MD3 (msb) | Number |
| | | | | | MD4 (lsb) | Num2 | | |
| last write | MD4 (lsb)<br>MD5 (msb) | Divisor | MD4 (lsb)<br>MD5 (msb) | Divisor | MD1 (msb) | Num1 | ARCON | |
| | | | | | MD5 (msb) | Num2 | | |

*Table 68. MDU registers write sequence*

1. Write MD0 to start any operation.
2. Write operations, as shown in Table 68. to determine appropriate MDU operation.
3. Write (to MD5 or ARCON) starts selected operation.

The SFR Control detects some of the above sequences and passes control to the MDU. When a write access occurs to MD2 or MD3 between write accesses to MD0 and finally to MD5, then a 32/16 bit division is selected.

When a write access to MD4 or MD1 occurs before writing to MD5, then a 16/16 bit division or 16×16 bit multiplication is selected. Writing to MD4 selects 16/16 bit division and writing to MD1 selects 16×16 bit multiplication, that is, Num1 × Num2.

### 14.4.2 Executing calculation

During executing operation, the MDU works on its own in parallel with the MCU.

| Operation | Number of clock cycles | |
|---|---|---|
| Division 32bit/16bit | 17 clock cycles | |
| Division 16bit/16bit | 9 clock cycles | |
| Multiplication | 11 clock cycles | |
| Shift | min. 3 clock cycles (sc = 01h) | max 18 clock cycles (sc = 1Fh) |
| Normalize | min. 4 clock cycles (sc <- 01h) | max 19 clock cycles (sc <- 1Fh) |

*Table 69. MDU operations execution times*

### 14.4.3 Reading the result from the MDx registers

| Operation | 32 bit/16 bit | | 16 bit / 16 bit | | 16 bit x 16 bit | | Shift/normalize | |
|---|---|---|---|---|---|---|---|---|
| first read | MD0 (lsb)<br>MD1<br>MD2<br>MD3 (msb) | Quotient | MD0 (lsb)<br>MD1 (msb) | Quotient | MD0 (lsb)<br>MD1<br>MD2 | Product | MD0 (lsb)<br>MD1<br>MD2 | Number |
| last read | MD4 (lsb)<br>MD5 (msb) | Remainder | MD4 (lsb)<br>MD5 (msb) | Remainder | MD3 (msb) | | MD3 (msb) | |

*Table 70. MDU registers read sequence*

The Read out sequence of the first MDx registers is not critical but the last read (from MD5 - division and MD3 - multiplication, shift or normalize) determines the end of a whole calculation (end of phase three).

### 14.4.4 Normalizing

All leading zeroes of 32-bit integer variable stored in the `MD3`.. `MD0` registers are removed by shift left operations. The whole operation is completed when the MSB (Most Significant Bit) of `MD3` register contains a '1'. After normalizing, bits `ARCON`[4] (msb) .. `ARCON`[0] (lsb) contain the number of shift left operations that were done.

### 14.4.5 Shifting

In shift operation, 32-bit integer variable stored in the `MD3`... `MD0` registers (the former contains the most significant byte) is shifted left or right by a specified number of bits. The slr bit (ARCON.5) defines the shift direction and bits `ARCON`[4] ... `ARCON`[0] specify the shift count (which must not be 0). During shift operation, zeroes come into the left end of MD3 for shifting right or they come in the right end of the MD0 for shifting left.

### 14.4.6 The mdef flag

The mdef error flag (see Table 67. on page 120) indicates an improperly performed operation (when one of the arithmetic operations is restarted or interrupted by a new operation). The error flag mechanism is automatically enabled with the first write operation to MD0 and disabled with the final read instruction from MD3 (multiplication or shift/norm) or MD5 (division) in phase three.

The error flag is set when:

- If you write to MD0.. MD5 and/or ARCON during phase two of MDU operation (restart or calculations interrupting).
- If any of the MDx registers are read during phase two of MDU operation when the error flag mechanism is enabled. In this case, the error flag is set but the calculation is not interrupted.

The error flag is reset only after read access to the ARCON register. The error flag is read only.

### 14.4.7 The mdov flag

The mdov overflow flag (see Table 67. on page 120) is set when one of the following conditions occurs:

- division by zero.
- multiplication with a result greater than 0000 FFFFh.
- start of normalizing if the most significant bit of MD3 is set (“md3.7” = ‘1’).

Any operation of the MDU that does not match the above conditions clears the overflow flag.

**Note:** The overflow flag is exclusively controlled by hardware, it cannot be written.

# 15 Encryption/decryption accelerator

You can utilize the on-chip encryption/decryption accelerator for more time and power effective firmware. The accelerator is an 8 by 8 Galois Field Multiplier with an 8 bits output. The following polynomial is used:

m(x) = x8 + x4 + x3 + x + 1

This is the polynomial used by AES (Advanced Encryption Standard).

## 15.1 Features

- Firmware available from Nordic Semiconductor.
- The result from the co-processing is available one clock period after the input data registers have changed.

## 15.2 Block diagram

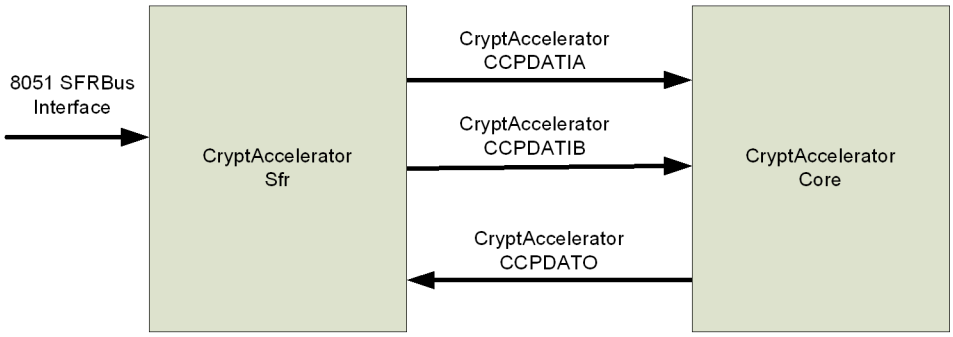



*Figure 55. Encryption/decryption accelerator*

## 15.3 Functional description

The following registers control the encryption/decryption accelerator.

| Address (Hex) | Name/Mnemonic | Bit | Reset values | Type | Description |
|---|---|---|---|---|---|
| 0xDD | CCPDATIA | 7:0 | 0x00 | R/W | Encryption/decryption accelerator data in register A. |
| 0xDE | CCPDATIB | 7:0 | 0x00 | R/W | Encryption/decryption accelerator data in register B. |
| 0xDF | CCPDATO | 7:0 | 0x00 | R | Encryption/decryption accelerator data out register. |

*Table 71. Encryption/decryption accelerator registers*

The two registers CCPDATIA and CCPDATIB contain the input data, whilst CCPDATO contains the result from the co-processing. CCPDATO is updated one clock period after one of the input data registers has changed.

# 16 Random number generator

The nRF24LE1 OTP contains a true Random Number Generator (RNG), which uses thermal noise to produce a non-deterministic bitstream. A digital corrector algorithm is employed on the bitstream to remove any bias toward '1' or '0'. The bits are then queued into an 8-bit register for parallel readout.

## 16.1 Features

- Non-deterministic architecture based on thermal noise
- No seed value required
- Non-repeating sequence
- Corrector algorithm ensures uniform statistical distribution
- Data rate up to 10 kilobytes per second
- Operational while the processor is in standby

## 16.2 Block diagram

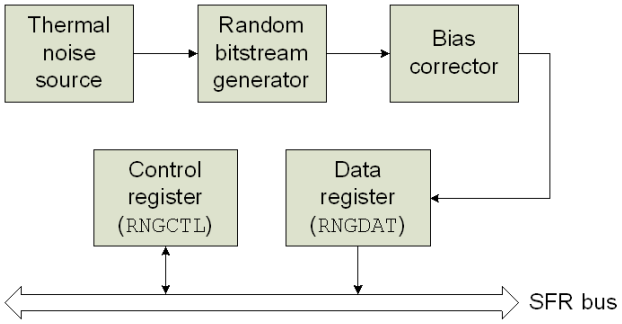



*Figure 56. Block diagram of RNG*

## 16.3 Functional description

Write a '1' to the `powerUp` control bit to start the generator. The `resultReady` status bit flags when a random byte is available for readout in the `RNGDAT` register. It will be cleared when the data has been read, and set again when a new byte is ready. An interrupt (`RNGIRQ`) is also produced each time a new byte has been generated. The behavior of the interrupt is the same as that of the `resultReady` status bit.

The random data and the `resultReady` status bit are invalid and should not be used when the RNG is powered down. When the RNG is powered up, by writing a '1' to the `powerUp` control bit, the random data and the `resultReady` status bit are cleared regardless of whether the random data has been read or not.

It is possible to disable the bias corrector by clearing the `correctorEn` bit. This offers a substantial speed advantage, but may yield a statistical distribution that is not perfectly uniform.

The time needed to generate one byte of data is unpredictable, and may vary from one byte to the next. This is especially true when the corrector is enabled. It takes about 0.1ms on average to generate one byte when the corrector is disabled, and four times as long when it is enabled. There is an additional start-up delay of about 0.25 ms for the first byte, counted from when the `powerUp` control bit is set.

## 16.4 SFR registers

The RNG is interfaced through the two registers; `RNGCTL` and `RNGDAT`. `RNGCTL` contains control bits and a status bit. `RNGDAT` contains the random data.

| Addr | Bit | name | RW | Function | Reset value: 0x40 |
|---|---|---|---|---|---|
| 0xD6 | 7 | powerUp | RW | Power up RNG | |
| | 6 | correctorEn | RW | Enable bias corrector | |
| | 5 | resultReady | R | Data ready flag. Set when a fresh random byte is available in the `RNGDAT` register. Cleared when the byte has been read and when the RNG comes out of powerdown (when the powerUp bit changes from 0 to 1). | |
| | 4:0 | - | - | Not used | |

*Table 72. RNGCTL register*

| Addr | Bit | name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| 0xD7 | 7:0 | data | R | Random data | |

*Table 73. RNGDAT register*

# 17 General purpose IO port and pin assignments

The IO pins of the nRF24LE1 OTP are default set to general purpose IO for the MCU. The numbers of available IOs are 7 for the 24 pin 4×4mm, 15 for the 32 pin 5×5mm and 31 for the 48 pin 7×7mm package. The IO pins are also shared with IO requirements from peripheral blocks like SPI and 2 wire as well as more specialized functions like a 32 kHz crystal oscillator. Connections between these other peripheral blocks and the pins are made dynamically by the PortCrossbar module.

## 17.1 Block diagram

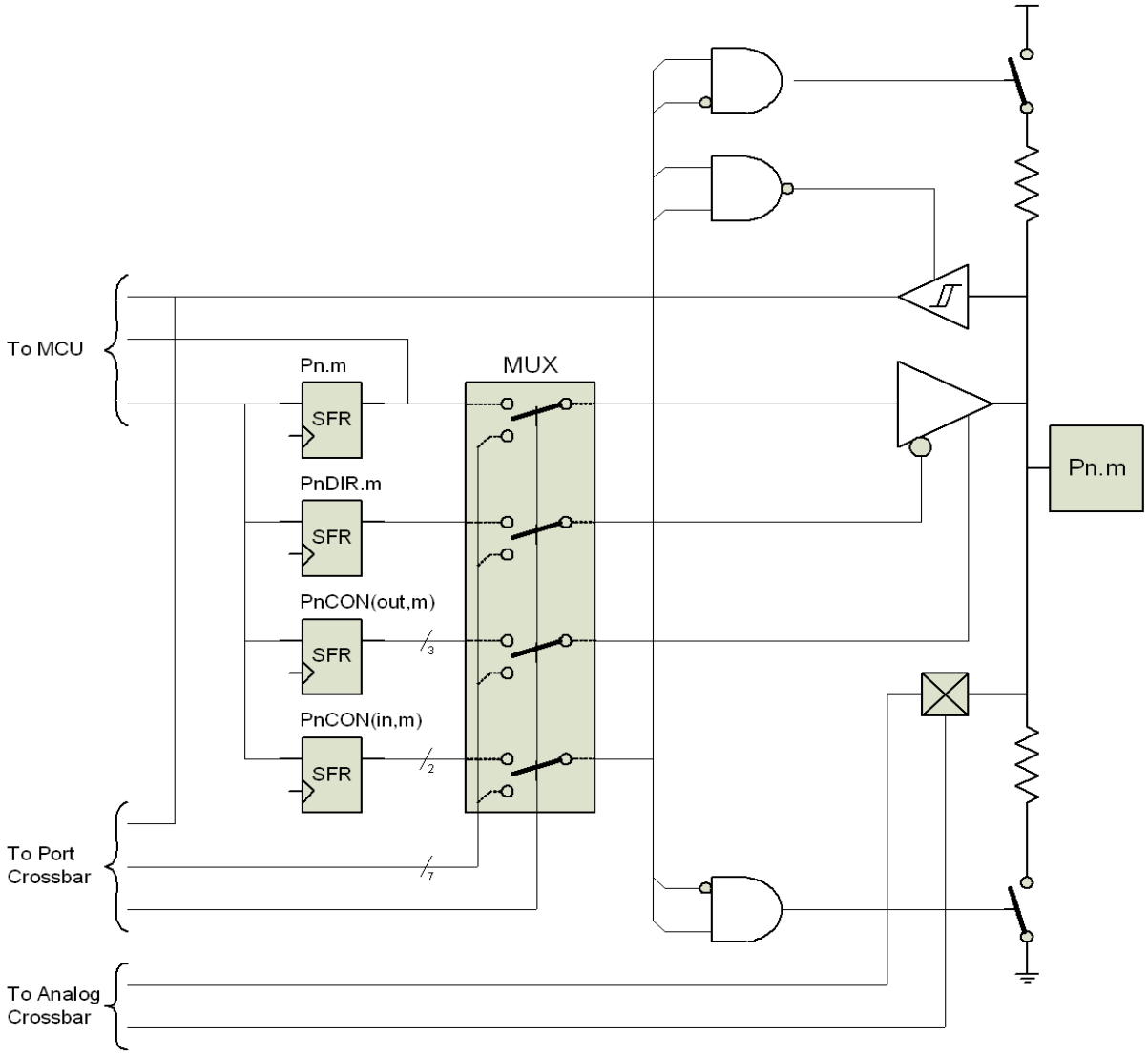



*Figure 57. IO pin circuitry block diagram*

## 17.2 Functional description

### 17.2.1 General purpose IO pin functionality

Each of the IO pins on nRF24LE1 OTP has a generic control functionality that sets pin features for the GPIO of the MCU.

The features offered by the pins include:

- Digital or Analog
- Configurable Direction
- Configurable Drive Strength
- Configurable Pull Up/Down

This functionality is multiplexed with the functionality of the PortCrossbar module which takes control and configures the pins depending on the needs of the peripheral block connected. The pin circuitry of the nRF24LE1 OTP is shown in Figure 57. on page 127

The pins on the nRF24LE1 OTP are connected by default to a pin Multiplexer (MUX) that is connected to the GPIO registers of the MCU. Register `Pn.m` (n-port number, m - bit number) contains MCU GPIO data, `PDIRn.m` register controls input/output direction and `PCONn.m` register controls pin features drive strength and pull up/down resistors for each pin.

When the MCU enables one of the peripheral blocks of the nRF24LE1 OTP the pin MUX disconnects the MCU control of the pin and hands control over to the PortCrossbar module to set direction and pin features.

However, if the pin is operated as an analog input, the MCU must set the pin control registers `PDIR` and `PCON` separately to prevent conflicts between pin configuration and the needs of the analog peripheral blocks of the nRF24LE1.

The nRF24LE1 OTP has one `Pn.m`, `PnDIRm` and `PnCONn` for each port. `Pn.m` and `PnDIRm` control only one parameter each, this means that a write/read operation to them controls/reads the status of the port directly. However, to control or read the features of a pin you use the `PnCONm` to write/read to one pin at a time. The `PnCON` register contains an address for the pin, information on whether it is an input or an output feature that is to be updated and the feature that is to be enabled.

The features available:

- Output buffer on, normal drive strength
- Output buffer on, high drive strength
- Input buffer on, no pull up/down resistor
- Input buffer on, pull up resistor
- Input buffer on, pull down resistor
- Input buffer off

Example: If four pins in port 3 are set as inputs with the pull up resistor enabled, then this is done with one write to `P3DIR` and four write operations to `P3CON` and only updating the pin address in `P3CON` for each write.

### 17.2.2 PortCrossbar functionality

The PortCrossbar sets up connections between the IO pins and the peripheral block of the device.

#### 17.2.2.1 Dynamic allocation of pins

The PortCrossbar modifies connections dynamically based on run-time variations in system needs of the peripheral blocks (SPI, 2 wire etc) of the device. This feature is necessary because the number of available pins is small compared to the combined IO needs of all the peripheral blocks. Consequently, on the smaller package options there may be conflicting pin assignments. These are resolved through a set of priorities assigned to each peripheral block. The pin out tables for each package option can be seen in Table 74. on page 131, Table 75. on page 132 and Table 76. on page 135.

#### 17.2.2.2 Dynamic pin allocation for digital blocks

Each digital peripheral block that needs an IO is represented in the pin out tables with the interface names of the block and the direction enforced on each pin. The priority of the blocks relative to potentially conflicting blocks is also shown. If the block is enabled, and no higher priority block is enabled, all the IO needs are granted. The PortCrossbar never grants partial fulfilment of a digital IO request even if a conflict exists only for some of the pins. A requesting digital device gets all or none of its IO needs granted.

#### 17.2.2.3 Dynamic pin allocation for analog blocks

A dynamic request for analog IO is similar to that of a digital IO. However, for analog blocks only the interface signals actually used as inputs to the analog blocks, configured by ADCCON1.chsel and ADCCON1.refsel, are connected to a device pin. This is different from the digital peripheral blocks where all the IO of a block are reserved once the block is enabled.

The two analog blocks, ADC and analog comparator, share a column in the pin out tables. This is done because the comparator uses the ADC configuration registers for selecting the source pins for its signal and voltage reference inputs. Please refer to chapter 21 on page 160 and chapter 22 on page 166 for more details.

**Note:** The implementation does not prevent simultaneous digital and analog use of a pin. If a pin is to be used for analog input, digital I/O buffers and digital peripheral blocks connected to the same pin should normally be disabled. Conflicts between analog blocks are resolved through priority.

The IO needs of the XOSC32K are also run-time programmable. Depending on configuration, this block may request either analog or digital IO. See section 13.3.4 on page 117 for further details.

If analog functionality is enabled for a pin, this is done without modifying or disabling the pins digital configuration. If particular digital input and/or output configuration are necessary for an analog pin to function correctly, this configuration must be enabled in registers `PxCON` and `PxDIR` separately, before enabling the analog block.

#### 17.2.2.4 Default pin allocation

If no peripheral blocks request IO, a default pinout as listed in the default column in the pin out maps are enabled. This means that all device pins are used for MCU GPIO. After reset, all IOs are configured to be digital inputs. The features, direction and IO data on the pins are in this case controlled by registers `PnCON`, `PnDIR` and `Pn`.

The default pin out also includes connections that are conditionally enabled based on the direction set for the pin. For example, if the P0DIR register in a 24 pin 4×4 mm package sets pin **P0.6** as an input, it can be used as a MCU GP input and as the UART receiver. If pin P0.5 is programmed as an output, it can be connected to the MCU as a GP output, but also have conditional output from the UART/TXD through an AND gate.

## 17.3 IO pin maps

The following conventions are used in all pin out maps:

- For dynamic connections of digital peripheral blocks, the direction of each pin is indicated by ‘in’, ‘out’ or ‘inout’ next to the interface name.
- Dynamic analog connections are indicated with ‘ana’.
- Digital peripheral blocks with potentially conflicting IO needs are highlighted with blue background in the pinout tables.
- For blocks marked with a green background, conflicts may exist with other green and blue devices, depending on the configuration. Please refer to the documentation of the configurable (green) blocks for information on how the configuration affects the IO usage.
- The relative priorities of competing digital peripheral blocks are listed in the table header.

### 17.3.1 Pin assignments in package 24 pin 4×4 mm

The connection map described in this chapter is valid for nRF24LE1 OTP in the 24 pin 4×4 mm package.Pins **P0.0**, **P0.2**, **P0.4** and **P0.6** have two system inputs listed per pin. This means that the input from the pin is driving both blocks inputs through an AND gate when the pin is configured as an input. Pin **P0.5** and **P0.6** are listed with two system outputs, such as p0Do 5 and UART/TXD. In these two cases the PortCrossbar also combines the two drivers using an AND gate and lets the AND gate drive the pin if it is configured as an output. The AND gate is chosen since both the UART/TXD and UARAT/RXD signals are high when idle.

The SMISO pin driver is only enabled when the SCSN pin is active.

| Pin | Default connections | | Dynamically enabled connections | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Inputs[a] | Outputs[a] | XOSC32K | | SPI Master | | Slave/OTP SPI | | 2-Wire | | PWM | | ADC/COMP | |
| | | | priority 1 | | priority 2 | | priority 3 | | priority 5 | | priority 6 | | priority 7 | |
| P0.6 | p0Di 6<br>UART/ RXD | p0Do 6 | | | | | | | W2SDA | inout | PWM1 | out | AIN6 | ana |
| P0.5 | p0Di 5 | p0Do 5<br>UART/ TXD | | | | | SCSN<br>FCSN[b] | in<br>in | W2SCL | inout | | | AIN5 | ana |
| P0.4 | p0Di 4<br>T0 | p0Do 4 | | | MMISO | in | SMISO<br>FMISO[a] | out<br>out | | | | | AIN4 | ana |
| P0.3 | p0Di 3 | p0Do 3 | | | MMOSI | out | SMOSI<br>FMOSI[a] | in<br>in | | | PWM0 | out | AIN3 | ana |
| P0.2 | p0Di 2<br>GPINT1 | p0Do 2 | | | MSCK | out | SSCK<br>FSCK[a] | in<br>in | | | | | AIN2 | ana |
| P0.1 | p0Di 1 | p0Do 1 | CLKLF[c] | | | | | | | | | | AIN1 | ana |
| P0.0 | p0Di 0<br>GPINT0 | p0Do 0 | CLKLF[d] | ana | | | | | | | | | AIN0 | ana |
| | Conflict exists, use priorities to determine IO allocation | | | | | | | | | | | | | |
| | Conflict may exist depending on device configuration. In the case of a conflict, use priorities to determine IO allocation | | | | | | | | | | | | | |

a. The bit prefixes p<x>Di and p<x>Do indicate normal functionality for the inputs and outputs of pin <x>.
b. OTP SPI interface only activated when **PROG** is set higher than VIH for the **PROG** pin, no conflict with runtime operations
c. Connection depends on configuration register CLKLFCTRL[2:0]
CLKLFCTRL[2:0] = 000: Crystal connected between pin **P0.0** and pin **P0.1**.
CLKLFCTRL[2:0] = 011: Low-amplitude clock source for CLKLF from analog connection pin **P0.1**.
CLKLFCTRL[2:0] = 100: Digital clock source for CLKLF.
d. Connection depends on configuration register CLKLFCTRL[2:0]
CLKLFCTRL[2:0] = 000: Crystal connected between pin **P0.0** and pin **P0.1**.

*Table 74. Pin out map for the 24 pin 4×4mm package*

### 17.3.2 Pin assignments in package 32 pin 5×5 mm

The connection map described in this chapter is valid with the 32-pin 5×5 QFN package. Pins **P0.4** to **P1.0** have two system inputs listed per pin. This means that the input from the pin is driving both block inputs if the pin is configured as an input. Pins **P0.3**–**P0.4** are listed with two system outputs, such as p0Do 3 and TXD. In these two cases the PortCrossbar combines the two drivers using an AND gate and lets the AND gate drive the pin if it is configured as an output. The AND gate is chosen since both the TXD and RXD signals are high when idle. The SMISO pin driver is enabled only when SCSN is active.

| pin | Default connections | | Dynamically enabled connections | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Inputs | Outputs | XOSC32K | | SPI Master | | Slave/OTP SPI | | PWM | | ADC/COMP | | 2-Wire | |
| | | | priority 1 | | priority 2 | | priority 3 | | priority 4 | | priority 5 | | priority 7 | |
| **P1.6** | p1Di.6 | p1Do.6 | | | MMISO | in | | | | | | | | |
| **P1.5** | p1Di.5 | p1Do.5 | | | MMOSI | out | | | | | | | | |
| **P1.4** | p1Di.4 | p1Do.4 | | | MSCK | out | | | | | | | | |
| **P1.3** | p1Di.3 | p1Do.3 | | | | | | | | | | | | |
| **P1.2** | p1Di.2 | p1Do.2 | | | | | | | | | AIN10 | ana | | |
| **P1.1** | p1Di.1 | p1Do.1 | | | | | SCSN | in | | | AIN 9 | ana | | |
| | | | | | | | FCSN[a] | in | | | | | | |
| **P1.0** | p1Di.0 | p1Do.0 | | | | | SMISO | out | | | AIN 8 | ana | | |
| | T1 | | | | | | FMISO[a] | out | | | | | | |
| **P0.7** | p0Di.7 | p0Do.7 | | | | | SMOSI | in | | | AIN 7 | ana | | |
| | T0 | | | | | | FMOSI[a] | in | | | | | | |
| **P0.6** | p0Di.6 | p0Do.6 | | | | | | | | | AIN 6 | ana | | |
| | GPINT1 | | | | | | | | | | | | | |
| **P0.5** | p0Di.5 | p0Do.5 | | | | | SSCK | in | | | AIN 5 | ana | W2SDA | inout |
| | GPINT0 | | | | | | FSCK[a] | in | | | | | | |
| **P0.4** | p0Di.4 | p0Do.4 | | | | | | | | | AIN 4 | ana | W2SCL | inout |
| | UART/ RXD | | | | | | | | | | | | | |
| **P0.3** | p0Di.3 | p0Do.3 | | | | | | | PWM1 | out | AIN 3 | ana | | |
| | | UART/ TXD | | | | | | | | | | | | |
| **P0.2** | p0Di.2 | p0Do.2 | | | | | | | PWM0 | out | AIN 2 | ana | | |
| **P0.1** | p0Di.1 | p0Do.1 | CLKLF [b] | | | | | | | | AIN1 | ana | | |
| **P0.0** | p0Di.0 | p0Do.0 | CLKLF [c] | ana | | | | | | | AIN0 | ana | | |
| | Conflict exists, use priorities to determine IO allocation | | | | | | | | | | | | | |
| | Conflict may exist depending on device configuration. In the case of a conflict, use priorities to determine IO allocation | | | | | | | | | | | | | |

a. OTP SPI interface only activated when PROG is set higher than VIH for the **PROG** pin, no conflict with runtime operations.

b. Connection depends on configuration register CLKLFCTRL[2:0]
CLKLFCTRL[2:0] = 000: Crystal connected between pin **P0.0** and pin **P0.1**.
CLKLFCTRL[2:0] = 011: Low-amplitude clock source for CLKLF from pin **P0.1**.
CLKLFCTRL[2:0] = 100: Digital clock source for CLKLF.

c. Connection depends on configuration register CLKLFCTRL[2:0]
CLKLFCTRL[2:0] = 000: Crystal connected between pin **P0.0** and pin **P0.1**.

*Table 75. Pin out map for the 32 pin 5×5mm package*

### 17.3.3 Pin assignments in package 48 pin 7×7 mm

Due to the pin count in this package no IO conflicts exists between digital peripheral blocks. Pins **P1.1**–**P1.7** have two system inputs listed per pin. This means that the input from the pin is driving both system inputs if the pin is configured as an input.

Pins **P1.0**–**P1.1** are listed with two system outputs, such as p1Do 1 and TXD. In these two cases the PortCrossbar combines the two drivers using an AND gate and lets the AND gate drive the pin if it is configured as an output. The AND gate is chosen since both the TXD and RXD signals are high when idle. The SMISO pin driver is enabled only when SCSN is active.

| Pin | Default connections | | Dynamically enabled connections | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Inputs | Outputs | XOSC32K | | ADC/COMP | | SPI Master | | Slave/OTP SPI | | PWM | | | 2-Wire | | | | |
| | | | priority 1 | | priority 4 | | priority 2 | | | | priority 6 | | | priority 7 | | | | |
| P3.6 | p3Di.6 | p3Do.6 | | | | | | | | | | | | | | | | |
| P3.5 | p3Di.5 | p3Do.5 | | | | | | | | | | | | | | | | |
| P3.4 | p3Di.4 | p3Do.4 | | | | | | | | | | | | | | | | |
| P3.3 | p3Di.3 | p3Do.3 | | | | | | | | | | | | | | | | |
| P3.2 | p3Di.2 | p3Do.2 | | | | | | | | | | | | | | | | |
| P3.1 | p3Di.1 | p3Do.1 | | | | | | | | | | | | | | | | |
| P3.0 | p3Di.0 | p3Do.0 | | | | | | | | | | | | | | | | |
| P2.7 | p2Di.7 | p2Do.7 | | | | | | | | | | | | | | | | |
| P2.6 | p2Di.6 | p2Do.6 | | | | | | | | | | | | | | | | |
| P2.5 | p2Di.5 | p2Do.5 | | | | | | | | | | | | | | | | |
| P2.4 | p2Di.4 | p2Do.4 | | | | | | | | | | | | | | | | |
| P2.3 | p2Di.3 | p2Do.3 | | | | | | | | | | | | | | | | |
| P2.2 | p2Di.2 | p2Do.2 | | | | | | | | | | | | | | | | |
| P2.1 | p2Di.1 | p2Do.1 | | | | | | | | | | | | | | | | |
| P2.0 | p2Di.0 | p2Do.0 | | | | | | | FCSN[a] | in | | | | | | | | |
| P1.7 | p1Di.7 | p1Do.7 | | | | | | | | | | | | | | | | |
| | T2 | | | | | | | | | | | | | | | | | |
| P1.6 | p1Di.6 | p1Do.6 | | | | | | | FMISO[a] | out | | | | | | | | |
| | T1 | | | | | | | | | | | | | | | | | |
| P1.5 | p1Di.5 | p1Do.5 | | | AIN13 | ana | | | FMOSI[a] | in | | | out | | | | | |
| | T0 | | | | | | | | | | | | | | | | | |
| P1.4 | p1Di.4 | p1Do.4 | | | AIN12 | ana | | | | | | | out | | | | | |
| | GPINT2 | | | | | | | | | | | | | | | | | |
| P1.3 | p1Di.3 | p1Do.3 | | | AIN11 | ana | | | | | | | in | W2SDA | | inout | | |
| | GPINT1 | | | | | | | | | | | | | | | | | |
| P1.2 | p1Di.2 | p1Do.2 | | | AIN10 | ana | | | FSCK[a] | in | | | in | W2SCL | | inout | | |
| | GPINT0 | | | | | | | | | | | | | | | | | |
| P1.1 | p1Di.1 | p1Do.1 | | | AIN9 | ana | | | | | | | in | | | | | |
| | UART/ RXD | | | | | | | | | | | | | | | | | |
| P1.0 | p1Di.0 | p1Do.0 | | | AIN8 | ana | MMISO | in | | | | | | | | | | |
| | | UART/ TXD | | | | | | | | | | | | | | | | |
| P0.7 | p0Di.7 | p0Do.7 | | | AIN7 | ana | MMOSI | out | | | PWM0 | out | | | | | | |
| P0.6 | p0Di.6 | p0Do.6 | | | AIN6 | ana | MSCK | out | | | PWM1 | out | | | | | | |
| P0.5 | p0Di.5 | p0Do.5 | | | AIN5 | ana | | | SCSN | in | | | | | | | | |
| P0.4 | p0Di.4 | p0Do.4 | | | AIN4 | ana | | | SMISO | out | | | | | | | | |
| P0.3 | p0Di.3 | p0Do.3 | | | AIN3 | ana | | | SMOSI | in | | | | | | | | |
| P0.2 | p0Di.2 | p0Do.2 | | | AIN2 | ana | | | SSCK | in | | | | | | | | |
| P0.1 | p0Di.1 | p0Do.1 | CLKLF [b] | | AIN1 | ana | | | | | | | | | | | | |
| P0.0 | p0Di.0 | p0Do.0 | CLKLF [c] | ana | AIN0 | ana | | | | | | | | | | | | |
| | Conflict may exist depending on device configuration. In the case of a conflict, use priorities to determine IO allocation. | | | | | | | | | | | | | | | | | |

a. OTP SPI interface only activated when PROG is set higher than the VIH of the **PROG** pin, no conflict with runtime operations.

b. Connection depends on configuration register CLKLFCTRL[2:0]
   CLKLFCTRL[2:0] = 000: Crystal connected between pin **P0.0** and pin **P0.1**.
   CLKLFCTRL[2:0] = 011: Low-amplitude clock source for CLKLF from pin **P0.1**.
   CLKLFCTRL[2:0] = 100: Digital clock source for CLKLF.
c. Connection depends on configuration register CLKLFCTRL[2:0]
   CLKLFCTRL[2: 0] = 000: Crystal connected between pin **P0.0** and pin **P0.1**.

*Table 76. Pin out map for the 48 pin 7×7mm package*

### 17.3.4 Programmable registers

Depending on the package size 1 to 4 ports are available on nRF24LE1. Desired pin direction and functionality is configured using the configuration registers P0DIR, P1DIR, P2DIR, P3DIR, collectively referred to as PxDIR, and P0CON, P1CON, P2CON and P3CON, referred to as PxCON. The PxDIR registers determine the direction of the pins and the PxCON registers contain the functional options for input and output pin operation.

The PortCrossbar by default (at reset) configures all pins as inputs and connects them to the MCU GPIO (pxDi).

To change pin direction, write the desired direction to the PxDIR registers.

| Register name: P0DIR | | | Address: 0x93 | Reset value: 0xFF |
|---|---|---|---|---|
| **Bit** | **Name** | **RW** | **Function** | |
| 7:0 | dir | RW | Direction bits for pins **P0.0–P0.7**. Output: dir = 0, Input: dir = 1.<br><br>P0DIR 0 - **P0.0**<br>P0DIR 1 - **P0.1**<br>P0DIR 2 - **P0.2**<br>P0DIR 3 - **P0.3**<br>P0DIR 4 - **P0.4**<br>P0DIR 5 - **P0.5**<br>P0DIR 6 - **P0.6**<br>P0DIR 7 - **P0.7**<br><br>**P0.7** only available on packages 32 pin 5×5mm and 48 pin 7×7mm | |

*Table 77. P0DIR register*

| Register name: P1DIR | | | Address: 0x94 | Reset value: 0xFF |
|---|---|---|---|---|
| **Bit** | **name** | **RW** | **Function** | |
| 7:0 | dir | RW | Direction bits for pins **P1.0–P1.7**. Output: dir = 0, Input: dir = 1.<br>P1DIR 0 - **P1.0**<br>P1DIR 1 - **P1.1**<br>P1DIR 2 - **P1.2**<br>P1DIR 3 - **P1.3**<br>P1DIR 4 - **P1.4**<br>P1DIR 5 - **P1.5**<br>P1DIR 6 - **P1.6**<br>P1DIR 7 - **P1.7**<br><br>Port1 only available on packages 32 pin 5×5 mm and 48 pin 7×7 mm<br>**P1.7** only available on package 48 pin 7×7 | |

*Table 78. P1DIR register*

| Register name: P2DIR | | | Address: 0x95 | Reset value: 0xFF |
|---|---|---|---|---|
| **Bit** | **Name** | **RW** | **Function** | |
| 7:0 | dir | RW | Direction bits for pins **P2.0** – **P2.7**. (Not used by the 5x5mm package). Output: dir = 0, Input: dir = 1.<br>P2DIR 0 - **P2.0**<br>P2DIR 1 - **P2.1**<br>P2DIR 2 - **P2.2**<br>P2DIR 3 - **P2.3**<br>P2DIR 4 - **P2.4**<br>P2DIR 5 - **P2.5**<br>P2DIR 6 - **P2.6**<br>P2DIR 7 - **P2.7**<br><br>Port2 only available on package 48 pin 7x7mm | |

*Table 79. P2DIR register*

| Register name: P3DIR | | | Address: 0x96 | Reset value: 0xFF |
|---|---|---|---|---|
| **Bit** | **Name** | **RW** | **Function** | |
| 7:0 | dir | RW | Direction bits for pins **P3.0** – **P3.6**. (Not used by the 5x5mm package). Output: dir = 0, Input: dir = 1.<br>P3DIR 0 - **P3.0**<br>P3DIR 1 - **P3.1**<br>P3DIR 2 - **P3.2**<br>P3DIR 3 - **P3.3**<br>P3DIR 4 - **P3.4**<br>P3DIR 5 - **P3.5**<br>P3DIR 6 - **P3.6**<br>P3DIR 7 - reserved<br><br>Port 3 only available on package 48 pin 7x7mm | |

*Table 80. P3DIR register*

The input and output options of each pin are configured in the PxCON registers. The PxCON registers have to be written once per pin (one write operation to the PxCON register configures the input/output options of a selected pin in the port).

To read the current input or output options for a pin, you first need to perform a write operation to retrieve the desired bit address and option type (input or output).

For instance, to read the output mode of pin **P0.5**: Write to P0CON with a bitAddr value of 101, a readAddr value of 1 and a inOut value of 0 (output). Then read from P0CON. The output mode of pin 5 is now found in bits 7:5 of the read data.

| Register name: P0CON | | | Address: 0x9E | Reset value: 0x00 |
|---|---|---|---|---|
| **Bit** | **Name** | **RW** | **Function** | |
| 7:5 | pinMode | RW | Functional input or output mode for pins **P0.0** – **P0.7**.<br><br>For a write operation: The functional mode you would like to write to the pin. The inOut field determines if the input or output mode is written, the bitAddr field determines which pin is affected.<br><br>Output modes using bits 7:5:<br>000 Digital output buffer normal drive strength<br>011 Digital output buffer high drive strength<br>(all other value combinations are illegal)<br><br>Input modes using bits 6:5:<br>00 Digital input buffer on, no pull up/down resistors<br>01 Digital input buffer on, pull down resistor connected<br>10 Digital input buffer on, pull up resistor connected<br>11 Digital input buffer off<br><br>For a read operation: The current functional mode of the pin. The inOut field determines if the input or output mode is reported, while the bitAddr field indicates which pin is selected. | |
| 4 | inOut | W | This bit indicates if the current write operation relates to the input or output configuration of the addressed pin.<br>inOut = 0 - Operate on the output configuration<br>inOut = 1 - Operate on the input configuration | |
| 3 | readAddr | W | If this bit is set, the purpose of the current write operation is to provide the bit address for later read operations. Consequently, the value of the bitAddr field is saved. The value of the inOut field is also saved, determining if the input or output mode is to be read. The pinMode field is ignored when readAddr is set.<br>If this bit is not set, the pin mode of the addressed pin is updated with the value of the pinMode field. The inOut field determines if the input or output mode is updated. | |
| 2:0 | bitAddr | W | If the readAddr bit is set, the value of the bitAddr field is stored. For subsequent read operations from P0CON, the pin for which the pinMode will be returned is given by the list below.<br><br>7x7mm 5x5mm 4x4mm<br>bitAddr = 000 - **P0.0** **P0.0** **P0.0**<br>bitAddr = 001 - **P0.1** **P0.1** **P0.1**<br>bitAddr = 010 - **P0.2** **P0.2** **P0.2**<br>bitAddr = 011 - **P0.3** **P0.3** **P0.3**<br>bitAddr = 100 - **P0.4** **P0.4** **P0.4**<br>bitAddr = 101 - **P0.5** **P0.5** **P0.5**<br>bitAddr = 110 - **P0.6** **P0.6** **P0.6**<br>bitAddr = 111 - **P0.7** **P0.7** reserved | |

*Table 81. P0CON register*

| Register name: P1CON | | | Address: 0x9F | Reset value: 0x00 |
|---|---|---|---|---|
| **Bit** | **Name** | **RW** | **Function** | |
| 7:5 | pinMode | RW | Functional input or output mode for pins **P1.0** – **P1.7**.<br><br>For a write operation: The functional mode you would like to write to the pin. The inOut field determines if the input or output mode is written, the bitAddr field determines which pin is affected.<br><br>Output modes using bits 7:5:<br>000 Digital output buffer normal drive strength<br>011 Digital output buffer high drive strength<br>(all other value combinations are illegal)<br><br>Input modes using bits 6:5:<br>00 Digital input buffer on, no pull up/down resistors<br>01 Digital input buffer on, pull down resistor connected<br>10 Digital input buffer on, pull up resistor connected<br>11 Digital input buffer off<br><br>For a read operation: The current functional mode of the pin. The inOut field determines if the input or output mode is reported, while the bitAddr field indicates which pin is selected. | |
| 4 | inOut | W | This bit indicates if the current write operation relates to the input or output configuration of the addressed pin.<br>inOut = 0 - Operate on the output configuration<br>inOut = 1 - Operate on the input configuration | |
| 3 | readAddr | W | If this bit is set, the purpose of the current write operation is to provide the bit address for later read operations. Consequently, the value of the bitAddr field is saved. The value of the inOut field is also saved, determining if the input or output mode is to be read. The pinMode field is ignored when readAddr is set.<br><br>If this bit is not set, the pin mode of the addressed pin is updated with the value of the pinMode field. The inOut field determines if the input or output mode is updated. | |
| 2:0 | bitAddr | W | If the readAddr bit is set, the value of the bitAddr field is stored. For subsequent read operations from P1CON, the pin for which the pinMode will be returned, is given by the list below.<br>7×7mm 5×5mm 4×4mm<br>bitAddr = 000 - **P1.0** **P1.0** reserved<br>bitAddr = 001 - **P1.1** **P1.1** reserved<br>bitAddr = 010 - **P1.2** **P1.2** reserved<br>bitAddr = 011 - **P1.3** **P1.3** reserved<br>bitAddr = 100 - **P1.4** **P1.4** reserved<br>bitAddr = 101 - **P1.5** **P1.5** reserved<br>bitAddr = 110 - **P1.6** **P1.6** reserved<br>bitAddr = 111 - **P1.7** reserved reserved | |

*Table 82. P1CON register*

| Register name: P2CON | | | Address: 0x97 | Reset value: 0x00 |
|---|---|---|---|---|
| **Bit** | **Name** | **RW** | **Function** | |
| 7:5 | pinMode | RW | Functional input or output mode for pins **P2.0** – **P2.7**. (Not used by the 5x5mm package).<br><br>For a write operation: The functional mode you would like to write to the pin. The inOut field determines if the input or output mode is written, the bitAddr field determines which pin is affected.<br><br>Output modes using bits 7:5:<br>000 Digital output buffer normal drive strength<br>011 Digital output buffer high drive strength<br>(all other value combinations are illegal)<br><br>Input modes using bits 6:5:<br>00 Digital input buffer on, no pull up/down resistors<br>01 Digital input buffer on, pull down resistor connected<br>10 Digital input buffer on, pull up resistor connected<br>11 Digital input buffer off<br><br>For a read operation: The current functional mode of the pin. The inOut field determines if the input or output mode is reported, while the bitAddr field indicates which pin is selected. | |
| 4 | inOut | W | This bit indicates if the current write operation relates to the input or output configuration of the addressed pin.<br>inOut = 0 - Operate on the output configuration<br>inOut = 1 - Operate on the input configuration | |
| 3 | readAddr | W | If this bit is set, the purpose of the current write operation is to provide the bit address for later read operations. Consequently, the value of the bitAddr field is saved. The value of the inOut field is also saved, determining if the input or output mode is to be read. The pinMode field is ignored when readAddr is set.<br><br>If this bit is not set, the pin mode of the addressed pin is updated with the value of the pinMode field. The inOut field determines if the input or output mode is updated. | |
| 2:0 | bitAddr | W | If the readAddr bit is set, the value of the bitAddr field is stored. For subsequent read operations from **P2CON**, the pin for which the pinMode will be returned, is given by the list below.<br>7×7mm 5×5mm 4×4mm<br>bitAddr = 000 - **P2.0** reserved reserved<br>bitAddr = 001 - **P2.1** reserved reserved<br>bitAddr = 010 - **P2.2** reserved reserved<br>bitAddr = 011 - **P2.3** reserved reserved<br>bitAddr = 100 - **P2.4** reserved reserved<br>bitAddr = 101 - **P2.5** reserved reserved<br>bitAddr = 110 - **P2.6** reserved reserved<br>bitAddr = 111 - **P2.7** reserved reserved | |

*Table 83. P2CON register*

| Register name: P3CON | | | Address: 0x8F | Reset value: 0x00 |
|---|---|---|---|---|
| **Bit** | **Name** | **RW** | **Function** | |
| 7:5 | pinMode | RW | Functional input or output mode for pins **P3.0** – **P3.6**. (Not used by the 5×5 mm package).<br><br>For a write operation: The functional mode you would like to write to the pin. The inOut field determines if the input or output mode is written, the bitAddr field determines which pin is affected.<br><br>Output modes using bits 7:5:<br>000 Digital output buffer normal drive strength<br>011 Digital output buffer high drive strength<br>(all other value combinations are illegal)<br><br>Input modes using bits 6:5:<br>00 Digital input buffer on, no pull up/down resistors<br>01 Digital input buffer on, pull down resistor connected<br>10 Digital input buffer on, pull up resistor connected<br>11 Digital input buffer off<br><br>For a read operation: The current functional mode of the pin. The inOut field determines if the input or output mode is reported, while the bitAddr field indicates which pin is selected. | |
| 4 | inOut | W | This bit indicates if the current write operation relates to the input or output configuration of the addressed pin.<br>inOut = 0 - Operate on the output configuration<br>inOut = 1 - Operate on the input configuration | |
| 3 | readAddr | W | If this bit is set, the purpose of the current write operation is to provide the bit address for later read operations. Consequently, the value of the bitAddr field is saved. The value of the inOut field is also saved, determining if the input or output mode is to be read. The pinMode field is ignored when readAddr is set.<br><br>If this bit is not set, the pin mode of the addressed pin is updated with the value of the pinMode field. The inOut field determines if the input or output mode is updated. | |
| 2:0 | bitAddr | W | If the readAddr bit is set, the value of the bitAddr field is stored. For subsequent read operations from P3CON, the pin for which the pinMode will be returned, is given by the list below.<br>7×7mm 5×5mm 4×4mm<br>bitAddr = 000 - **P3.0** reserved reserved<br>bitAddr = 001 - **P3.1** reserved reserved<br>bitAddr = 010 - **P3.2** reserved reserved<br>bitAddr = 011 - **P3.3** reserved reserved<br>bitAddr = 100 - **P3.4** reserved reserved<br>bitAddr = 101 - **P3.5** reserved reserved<br>bitAddr = 110 - **P3.6** reserved reserved<br>bitAddr = 111 - reserved reserved reserved | |

*Table 84. P3CON register*

While the IO ports are used as MCU GPIO, the pin values are read and controlled by the MCU port registers `P3` to `P0`.

| Address | Name | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| 0xB0 | **P3** | 7:0 | 0xFF | R/W | Port 3 value |
| 0xA0 | **P2** | 7:0 | 0xFF | R/W | Port 2 value |
| 0x90 | **P1** | 7:0 | 0xFF | R/W | Port 1 value |
| 0x80 | **P0** | 7:0 | 0xFF | R/W | Port 0 value |

*Table 85. P0-P3 registers*

How many ports are available depends on which of the three nRF24LE1 OTP package sizes you are using.

# 18 SPI

nRF24LE1 OTP features a double buffered Serial Peripheral Interface (SPI). You can configure it to work in all four SPI modes. The default is mode 0.

The SPI connects to the following pins of the device: **MMISO**, **MMOSI**, **MSCK**, **SCSN**, **SMISO**, **SMOSI** and **SSCK**.

The SPI Master function does not generate any chip select signal (CSN). The programmer typically uses another programmable digital I/O to act as chip selects for one or more external SPI Slave devices.

## 18.1 Features

- Double buffered FIFO
- Full-duplex operation
- Supports SPI modes 0 through 3
- Configurable data order on xMISO/xMOSI
- Four (Master) and three (Slave) interrupt sources

## 18.2 Block diagram

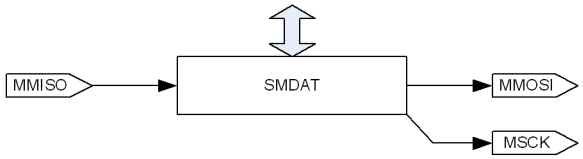



*Figure 58. SPI Master*

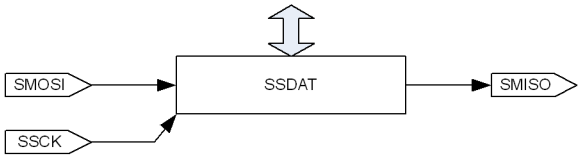



*Figure 59. SPI Slave*

## 18.3 Functional description

### 18.3.1 SPI master

The following registers control the SPI master:

| Address (Hex) | Name/mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| 0xFC | SPIMCON0 | 6:0 | 0x20 | R/W | SPI Master configuration register 0. |
| | *clockFrequency* | 6:4 | 010 | R/W | Frequency on MSCK. ckMCU is the MCU clock frequency.)<br>000: 1/2 ·ckMCU<br>001: 1/4·ckMCU<br>010: 1/8 ·ckMCU<br>011: 1/16·ckMCU<br>100: 1/32·ckMCU<br>101: 1/64·ckMCU<br>110: 1/64·ckMCU<br>111: 1/64·ckMCU |
| | *dataOrder* | 3 | 0 | R/W | Data order (bit wise per byte) on serial output and input (MMOSI and MMISO respectively).<br>1: LSBit first, MSBit last.<br>0: MSBit first, LSBit last. |
| | *clockPolarity* | 2 | 0 | R/W | Defines the SPI Master's operating mode together with SPIMCON0.1, see section 18.3.3 on page 147.<br>1: MSCK is active 'low'.<br>0: MSCK is active 'high'. |
| | *clockPhase* | 1 | 0 | R/W | Defines the SPI Master's operating mode together with SPIMCON0.2, see section 18.3.3 on page 147.<br>1: Sample on trailing edge of MSCK, shift on leading edge.<br>0: Sample on leading edge of MSCK, shift on trailing edge. |
| | *spiMasterEnable* | 0 | 0 | R/W | 1: SPI Master is enabled. The clock to the SPI Master core functionality is running. An SPI transfer can be initiated by the MCU via the 8051 SFR Bus (TX).<br>0: SPI Master is disabled. The clock to the SPI Master core functionality stands still. |
| 0xFD | SPIMCON1 | 3:0 | 0x0F | R/W | SPI Master configuration register 1. |
| | *maskIrqRxFifoFull* | 3 | 1 | R/W | 1: Disable interrupt when RX FIFO is full.<br>0: Enable interrupt when RX FIFO is full. |
| | *maskIrqRxDataReady* | 2 | 1 | R/W | 1: Disable interrupt when data is available in RX FIFO.<br>0: Enable interrupt when data is available in RX FIFO. |
| | *maskIrqTxFifoEmpty* | 1 | 1 | R/W | 1: Disable interrupt when TX FIFO is empty.<br>0: Enable interrupt when TX FIFO is empty. |

| Address (Hex) | Name/mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| | *maskIrqTxFifo-Ready* | 0 | 1 | R/W | 1: Disable interrupt when a location is available in TX FIFO.<br>0: Enable interrupt when a location is available in TX FIFO. |
| 0xFE | SPIMSTAT | 3:0 | 0x03 | R | SPI Master status register. |
| | *rxFifoFull* | 3 | 0 | R | Interrupt source.<br>1: RX FIFO full.<br>0: RX FIFO can accept more data from SPI.<br>Cleared when the cause is removed. |
| | *rxDataReady* | 2 | 0 | R | Interrupt source.<br>1: Data available in RX FIFO.<br>0: No data in RX FIFO.<br>Cleared when the cause is removed. |
| | *txFifoEmpty* | 1 | 1 | R | Interrupt source.<br>1: TX FIFO empty.<br>0: Data in TX FIFO.<br>Cleared when the cause is removed. |
| | *txFifoReady* | 0 | 1 | R | Interrupt source.<br>1: Location available in TX FIFO.<br>0: TX FIFO full.<br>Cleared when the cause is removed. |
| 0xFF | SPIMDAT | 7:0 | 0x00 | R/W | SPI Master data register.<br>Accesses TX (write) and RX (read) FIFO buffers, both two bytes deep. |

*Table 86. SPI Master registers*

The SPI Master is configured through SPIMCON0 and SPIMCON1. It is enabled by setting SPIMCON0[0] to '1'. The SPI Master supports all four SPI modes, selected by SPIMCON0[2] and SPIMCON0[1] as described in section 18.3.3. The bit wise data order per byte on MMISO/MMOSI is defined by SPIMCON0[3]. MSCK can run on one of six predefined frequencies in the range of 1/2 to 1/64 of the MCU clock frequency, as defined by SPIMCON0[6] down to SPIMCON0[4].

SPIMDAT accesses both the TX (write) and the RX (read) FIFOs, which are two bytes deep. The FIFOs are dynamic and can be refilled according to the state of the status flags: "FIFO ready" means that the FIFO can accept data. "Data ready" means that the FIFO can provide data, minimum one byte.

Four different sources can generate interrupt, unless they are masked by their respective bits in SPIMCON1. SPIMSTAT reveals which sources are active.

### 18.3.2 SPI slave

The following registers control the SPI slave:

| Address (Hex) | Name/mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| 0xBC | SPISCON0 | 6:0 | 0x70 | R/W | SPI Slave configuration register 0 |
| | *maskIrqCsnHigh* | 6 | 1 | R/W | 1: Disable interrupt when SCSN goes high.<br>0: Enable interrupt when SCSN goes high. |
| | *maskIrqCsnLow* | 5 | 1 | R/W | 1: Disable interrupt when SCSN goes low.<br>0: Enable interrupt when SCSN goes low. |
| | *maskIrqSpiSlave-Done* | 4 | 1 | R/W | 1: Disable interrupt when SPI Slave is done with the current SPI transaction.<br>0: Enable interrupt when SPI Slave is done with the current SPI transaction. |
| | *dataOrder* | 3 | 0 | R/W | Data order (bit wise per byte) on serial input and output (SMOSI and SMISO, respectively.)<br>1: LSBit first, MSBit last.<br>0: MSBit first, LSBit last. |
| | *clockPolarity* | 2 | 0 | R/W | Defines the SPI Slave's operating mode together with with SPISCON0.1, see chapter '18.3.3 Slave SPI timing'.<br>1: SSCK is active 'low'.<br>0: SSCK is active 'high'. |
| | *clockPhase* | 1 | 0 | R/W | Defines the SPI Slave's operating mode together with with SPISCON0.2, see chapter '18.3.3 Slave SPI timing'.<br>1: Sample on trailing edge of SSCK, shift on leading edge.<br>0: Sample on leading edge of SSCK, shift on trailing edge. |
| | *spiSlaveEnable* | 0 | 0 | R/W | 1: SPI Slave is enabled. The clock to the SPI Slave core functionality is running. An SPI transfer can be initiated by an SPI Master (RX).<br>0: SPI Slave is disabled. The clock to the SPI Slave core functionality stands still. |
| 0xBE | SPISSTAT | 5:0 | 0x00 | R | SPI Slave status register |
| | *csnHigh* | 5 | 0 | R | Interrupt source.<br>1: Positive edge of SCSN detected<br>0: Positive edge of SCSN not detected. Cleared when read. |
| | *csnLow* | 4 | 0 | R | Interrupt source<br>1: Negative edge of SCSN detected<br>0: Negative edge of SCSN not detected. Cleared when read. |
| | *<Reserved>* | 3:1 | - | - | - |
| | *spiSlaveDone* | 0 | 0 | R | Interrupt source.<br>1: SPI Slave done with an SPI transaction.<br>0: SPI Slave not done with an SPI transaction. Cleared when read. |
| 0xBF | SPISDAT | 7:0 | 0x00 | R/W | SPI Slave data register. Accesses the RX (read)/TX (write) buffers. |

*Table 87. SPI Slave registers*

The SPI slave is configured through SPISCON0. It is enabled by setting SPISCON0.0 to '1'. The SPI Slave supports all four SPI modes, selected by SPISCON0.2 and SPISCON0.1 as described in section 18.3.3. The bit wise data order per byte on SMISO/SMOSI is defined by SPISCON0.3. There are three possible interrupt sources in the SPI Slave. Any one of them can be masked.

When an interrupt occurs, SPISSTAT provides information on what the source was.

SPISDAT is used for data access in both directions. Prior to the first clock from the external SPI master, the MCU can write a up to two bytes to SPISDAT, but only one before SCSN goes low. The first byte will be transferred from the external SPI Master to the Slave on SMISO while data is being transferred from the external Master to the Slave on SMOSI. For maximum data throuput, after the first byte has been transferred, software must ensure that there always are two bytes in the TX chain; one that is being transferred and another in the pipe. There are two ways of doing this:

1. Preload *two* TX data bytes as described above, and then one byte for each *Spi Slave done* interrupt until the transfer is completed.
2. Preload *one* TX data byte as described above, load two bytes at the first *Spi Slave done* interrupt and then one byte for each *Spi Slave done* interrupt until the transfer is completed. This approach is, for some of the highest SSCK frequencies, likely to require a pause between the first and the second bytes on SPI, giving the MCU time to load the next TX data.

### 18.3.3 Slave SPI timing

The four different SPI modes are presented in Table 88., Figure 60. and Figure 61. on page 148

| SPI mode | clockPolarity | clockPhase | Clock shift edge | | Clock sample edge | |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | Trailing | Falling | Leading | Rising |
| 1 | 0 | 1 | Leading | Rising | Trailing | Falling |
| 2 | 1 | 0 | Trailing | Rising | Leading | Falling |
| 3 | 1 | 1 | Leading | Falling | Trailing | Rising |

*Table 88. SPI modes*

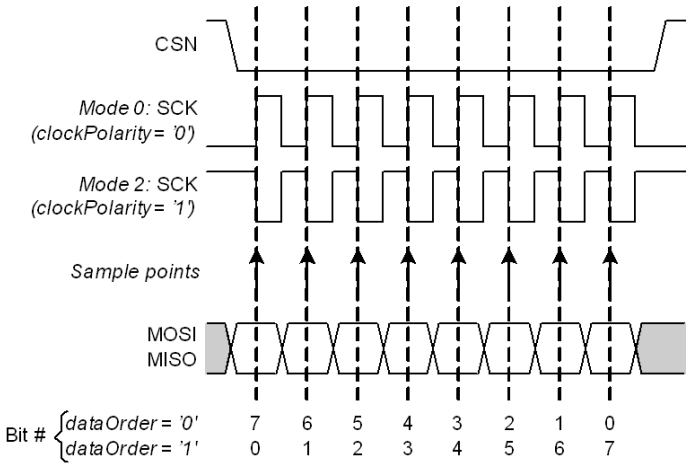



*Figure 60. SPI Modes 0 and 2: clockPhase = '0'. One byte transmission.*

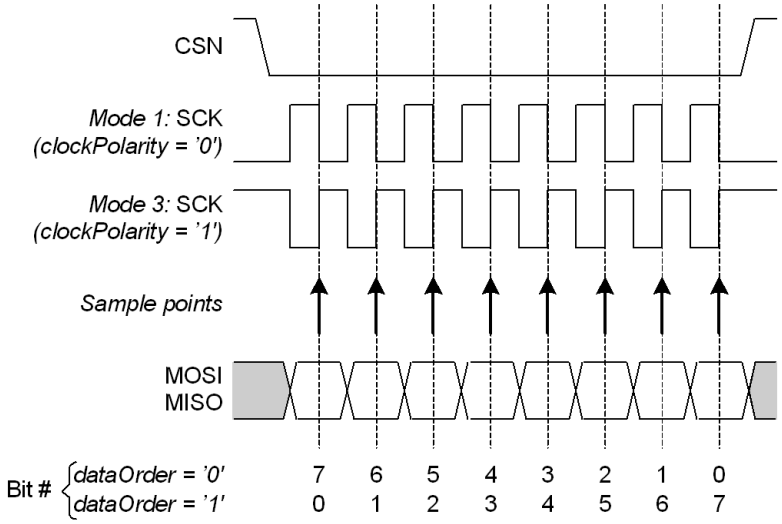



*Figure 61. SPI Modes 1 and 3: clockPhase = '1'. One byte transmission.*

SPI timing is given in Figure 62. and in Table 89. and Table 90.

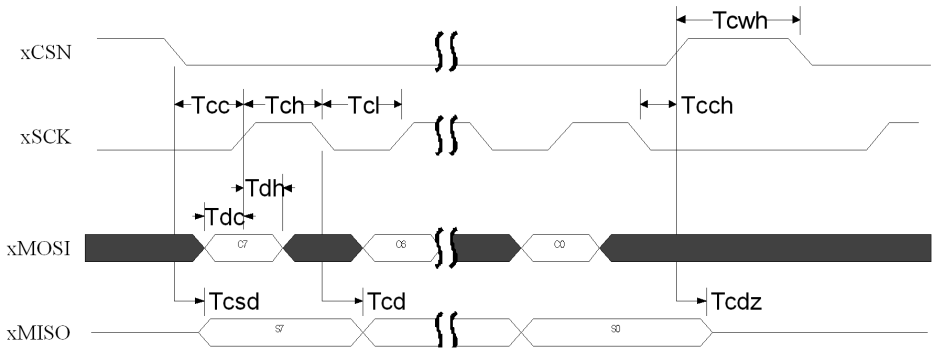



*Figure 62. SPI timing diagram. One byte transmission.*

| Parameters | Symbol | Min | Max | Units |
|---|---|---|---|---|
| Data to SCK Setup | Tdc | 2 | | ns |
| SCK to Data Hold | Tdh | 2 | | ns |
| CSN to Data Valid | Tcsd | | 38 | ns |
| SCK to Data Valid | Tcd | | 55 | ns |
| SCK Low Time | Tcl | 40 | | ns |
| SCK High Time | Tch | 40 | | ns |
| SCK Frequency | Fsck | 0 | 8 | MHz |
| SCK Rise and Fall | Tr,Tf | | 100 | ns |
| CSN to SCK Setup | Tcc | 2 | | ns |
| SCK to CSN Hold | Tcch | 2 | | ns |
| CSN Inactive time | Tcwh | 50 | | ns |
| CSN to Output High Z | Tcdz | | 38 | ns |

*Table 89. SPI timing parameters ($C_{Load}$ = 5pF)*

| Parameters | Symbol | Min | Max | Units |
|---|---|---|---|---|
| Data to SCK Setup | Tdc | 2 | | ns |
| SCK to Data Hold | Tdh | 2 | | ns |
| CSN to Data Valid | Tcsd | | 42 | ns |
| SCK to Data Valid | Tcd | | 58 | ns |
| SCK Low Time | Tcl | 40 | | ns |
| SCK High Time | Tch | 40 | | ns |
| SCK Frequency | Fsck | 0 | 8 | MHz |
| SCK Rise and Fall | Tr,Tf | | 100 | ns |
| CSN to SCK Setup | Tcc | 2 | | ns |
| SCK to CSN Hold | Tcch | 2 | | ns |
| CSN Inactive time | Tcwh | 50 | | ns |
| CSN to Output High Z | Tcdz | | 42 | ns |

*Table 90. SPI timing parameters ($C_{Load}$ = 10pF)*

# 19 Serial port (UART)

The MCU system is configured with one serial port that is identical in operation to the standard 8051 serial port (Serial interface 0). The two serial port signals RXD and TXD are available on device pins UART/RSD and UART/TXD

The serial port (UART) derives its clock from the MCU clock; ckCpu. See chapter 11.3.1 on page 105 for more information. The direction for the UART pins must be set to input for the RXD pin and output for the TXD pin in the corresponding PxDIR registers.

## 19.1 Features

- Synchronous mode, fixed baud rate
- 8-bit UART mode, variable baud rate
- 9-bit UART mode, variable baud rate
- 9-bit UART mode, fixed baud rate
- Additional baud rate generator

**Note:** It is not recommended to use Timer 1 overflow as baud generator.

## 19.2 Block diagram

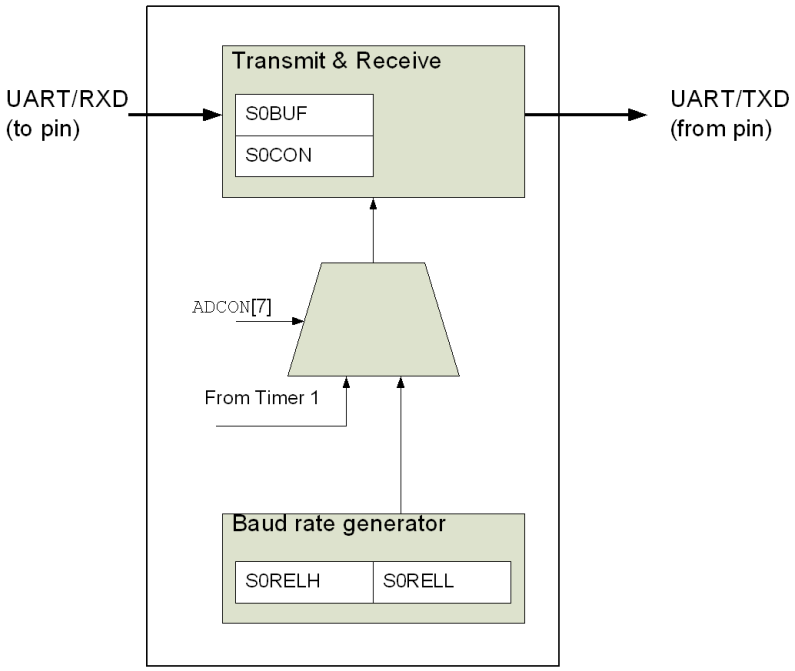



*Figure 63. Block diagram of serial port*

## 19.3 Functional description

The serial port is controlled by `S0CON`, while the actual data transferred is read or written in the `S0BUF` register. Transmission speed (“baud rate”) is selected using the `S0RELL`, `S0RELH` and `ADCON` registers.

### 19.3.1 Serial port 0 control register – S0CON

The S0CON register controls the function of Serial Port 0.

| Address | Reset value | Bit | Name | Description |
|---|---|---|---|---|
| 0x98 | 0x00 | 7:6 | sm0: sm1 | Serial Port 0 mode select<br>0 0: Mode 0 - Shift register at baud rate ckCpu / 12<br>0 1: Mode 1 - 8-bit UART.<br>1 0: Mode 2 - 9-bit UART at baud rate ckCpu /32 or ckCpu/64[a]<br>1 1: Mode 3 - 9 bit UART. |
| | | 5 | sm20 | Multiprocessor communication enable |
| | | 4 | ren0 | Serial reception enable: 1: Enable Serial Port 0. |
| | | 3 | tb80 | Transmitter bit 8. This bit is used while transmitting data through Serial Port 0 in Modes 2 and 3. The state of this bit corresponds with the state of the 9th transmitted bit (for example, parity check or multi-processor communication). It is controlled by software. |
| | | 2 | rb80 | Received bit 8. This bit is used while receiving data through Serial Port 0 in Modes 2 and 3. It reflects the state of the 9th received bit. |
| | | 1 | ti0 | Transmit interrupt flag. It indicates completion of a serial transmis-sion at Serial Port 0. It is set by hardware at the end of bit 8 in mode 0 or at the beginning of a stop bit in other modes. It must be cleared by software. |
| | | 0 | ri0 | Receive interrupt flag. It is set by hardware after completion of a serial reception at Serial Port 0. It is set by hardware at the end of bit 8 in mode 0 or in the middle of a stop bit in other modes. It must be cleared by software. |

a. If smod = 0 baud rate is ckCpu/64, if smod = 1 then baud rate is ckCpu/32.

*Table 91. S0CON register*

Table 92. on page 151 shows registers settings for some common UART baud rates.

| Baud rate | Cclk | SMOD | s0rel |
|---|---|---|---|
| 600 | 16 MHz | 1 | 0x00BF |
| 1200 | 16 MHz | 1 | 0x025F |
| 2400 | 16 MHz | 1 | 0x0330 |
| 4800 | 16 MHz | 1 | 0x0398 |
| 9600 | 16 MHz | 1 | 0x03CC |
| 19200 | 16 MHz | 1 | 0x03E6 |
| 38400 | 16 MHz | 1 | 0x03F3 |

*Table 92. Register settings for selected baud rates*

To configure other baud rates, please use the formulas in Figure 64. on page 152.

$$for\ bd(\texttt{adcon}[7]) = 0:$$

$$baudrate = \frac{2^{SMOD} * ckCpu}{32} * (Timer1\ overflowrate)$$

$$for\ bd(\texttt{adcon}[7]) = 1:$$

$$baudrate = \frac{2^{SMOD} * ckCpu}{64 * (2^{10} - s0rel)}$$

*Figure 64. Equation of baud rate settings for Serial Port 0*

Below is an explanation of some of the values used in Figure 64.:

| Value | Definition |
|---|---|
| SMOD (PCON[7]) | Serial Port 0 baud rate select flag |
| S0REL | The contents of S0REL registers (s0relh, s0rell) see section 19.3.3. |
| bd (adcon[7]) | The MSB of ADCON register see section 19.3.4 |

*Table 93.Values of S0CON equation*

### 19.3.2 Serial port 0 data buffer – S0BUF

| Address | Reset value | Register name |
|---|---|---|
| 0x99 | 0x00 | S0BUF |

*Table 94. S0BUF register*

Writing data to the SOBUF register sets data in serial output buffer and starts the transmission through Serial Port 0. Reading from the S0BUF reads data from the serial receive buffer.

### 19.3.3 Serial port 0 reload register – S0RELH, S0RELL

Serial Port 0 Reload register is used for Serial Port 0 baud rate generation. Only 10 bits are used, 8 bits from the S0RELL, and 2 bits from the S0RELH.

| Address | Reset value | Register name |
|---|---|---|
| 0xAA | 0xD9 | S0RELL |
| 0xBA | 0x03 | S0RELH |

*Table 95. S0RELL/S0RELH register*

### 19.3.4 Serial port 0 baud rate select register - ADCON

The MSB of this register is used by Serial Port 0 for baud rate generation.

| Address | Reset value | Bit | Name | Description |
|---|---|---|---|---|
| 0xD8 | 0x00 | 7 | bd | Serial Port 0 baud rate select (in modes 1 and 3)<br>When 1, additional internal baud rate generator is used, otherwise Timer 1 overflow is used. |
| | | 6:0 | | Not used |

*Table 96. ADCON register*

# 20 2-Wire

The nRF24LE1 OTP has a single buffered 2-Wire interface. It can be configured to transmit or receive data as master or slave, at two different data rates. The 2-Wire is not CBUS compatible.

The 2 wire interface connects to device pins **W2SDA** and **W2SCL**.

## 20.1 Features

- I2C compatible
- Single buffered
- Half-duplex operation
- Supports four modes: Master transmitter, Master receiver, Slave transmitter and Slave receiver
- Supports two baud rates: Standard mode (100 Kbit/s) and Fast mode (400 Kbit/s).
- Supports broadcast.
- Supports 7-bit addressing
- Supports Slave stall of serial clock (SCL).

## 20.2 Functional description

### 20.2.1 Recommended use

- The wire2Enable bit (W2CON0[0]) bit must be set to '1' in a separate write operation before any other programming of the 2-Wire is attempted.
- If the clockstop feature is used, the clockStop bit (W2CON0[6]) should be set to '1' before transmissions begin. In clockStop mode, all received data must be read from the W2DAT register, even received addresses. This is necessary to avoid stalling the 2-Wire bus.
- Updates to the maskIrq configuration bit (W2CON1[5]) should be performed before transmission begins.
- Once a '1' has been written to the xStart bit (W2CON0[4]) or the xStop bit (W2CON0[5]), the user should not attempt to cancel the request by clearing the bit at a later time.

### 20.2.2 Master transmitter/receiver

A new transfer is initiated by entering a start condition. This can be done by setting W2CON0[4] to '1', or simply by writing the first byte to W2DAT. The first byte is always transmitted from the Master.

#### 20.2.2.1 TX mode

To enter TX mode, MCU must write the address to the Slave it wants the 2-Wire to connect to, or the general call address (0x00), to W2DAT[7:1], and write '0' to the direction bit; W2DAT[0]. The byte is then transmitted to the Slave(s). If not masked, an interrupt request is asserted on the rising edge of SCL following the last bit in the byte. Simultaneously, the acknowledge from the addressed Slave is stored in W2CON1[1]. The 2-Wire is then ready to accept TX data from the MCU, and the bytewise transmissions will follow the same procedure as for the first byte.

To do a repeated start, the MCU must set W2CON0[4] before writing a new Slave address and direction bit to W2DAT. To stop the transfer, it must write '1' to W2CON0[5] after writing the last TX data byte to W2DAT. Start and stop conditions have lower priorities than pending TX data, that is, W2CON0[4] and W2CON0[5] can be set immediately after the last TX data write. If both bits are set, the stop condition is transmitted first.

#### 20.2.2.2 RX mode

To enter RX mode, MCU must write the address to the Slave it wants the 2-Wire to connect to, to W2DAT[ 7:1], and write '1' to the direction bit; W2DAT[0]. The byte is then transmitted to the Slave(s). If not masked, an interrupt request is asserted on the rising edge of SCL following the last bit in the byte. Simultaneously, the acknowledge from the addressed Slave is stored in W2CON1[1]. The 2-Wire then releases the control over the bus and is ready to accept bytewise RX data from the addressed Slave. For each byte received, if not masked, an interrupt request is asserted at the same time as the last bit is sampled, prior to sending the acknowledge to the Slave. The acknowledge is also stored in W2CON1[1].

To do a repeated start or stop the transfer, the MCU must set W2CON0[5] after receiving the second to last byte from the Slave. This makes the 2-Wire Master send a not-acknowledge after the last byte, which forces the Slave to let go of the bus control. After receiving the last byte, the Master can do a repeated start by writing a new Slave address and direction bit to W2DAT.

### 20.2.3 Slave transmitter/receiver

As the 2-Wire Slave detects a start condition it will enter RX mode and wait for the first byte from the Master. When the first byte is completed, the Slave compares W2DAT[7] down to W2DAT[1] to W2SADR (or the general call address, 0x00) to see if it is supposed to reply. If so, W2DAT[0] decides if it should stay in RX mode ('0') or enter TX mode ('1').

The 2-Wire Slave asserts interrupt requests to the MCU when 1) there is an address match after a start condition; 2) after each data byte received (RX mode) or transmitted (TX mode), or; 3) a stop condition is detected. All interrupts can be masked by configuration.

If the 2-Wire Slave's MCU has trouble processing the data fast enough, it can stall the transmission by setting W2CON0[6] to '1' between bytes. In TX mode, this forces SCL 'low' after transmission until the MCU has written new data to W2DAT. In RX mode, SCL is kept 'low' after reception, until the MCU has read the new data.

New TX data must always be written by the MCU to W2DAT before the next falling edge on SCL.

New RX data must always be read by the MCU from W2DAT before the next rising edge on SCL, after the corresponding interrupt request.

### 20.2.3.1 2-Wire timing

| Symbol | Parameter (CK = 16 MHz) | Standard | | Fast | | Unit |
|---|---|---|---|---|---|---|
| | | Min | Max | Min | Max | |
| $f_{CK}$ | System clock frequency. | 16 | | 16 | | MHz |
| $CK_{PERIOD}$ | System clock period. | 62.5 | | 62.5 | | ns |
| $SCL_{PERIOD}$ | SCL clock period. | 10000 | | 2500 | | ns |
| $t_{STA2SCL0}$ | Time from start condition to SCL goes 'low'. | | 4700 | | 940 | ns |
| $t_{SCL0F}$ | SCL 'low' time after start condition. | 5000 | | 1250 | | ns |
| $t_{DSETUP}$ | Data setup time before positive edge on SCL. | 4400 | | 800 | | ns |
| $t_{DHOLD}$ | Data hold time after negative edge on SCL. | $3 \cdot CK_P$ | 560 | $3 \cdot CK_P$ | 440 | ns |
| $t_{SCL0L}$ | SCL 'low' time after last bit before stop condition. | 5000 | | 1250 | | ns |
| $t_{SCL12STOP}$ | Time from SCL goes 'high' to stop condition. | 5000 | | 1300 | | ns |
| $t_{STOP2START}$ | Time from stop condition to start condition. | 4700 | | 1000 | | ns |
| $t_{REL}$ | Time from change on SDA until SCL is released when the module is a Slave that forces SCL 'low'. | 1400 | | 1400 | | ns |
| WIRQ | Width of IRQ signal. | $4 \cdot CK_P$ | | $4 \cdot CK_P$ | | ns |
| P2IRQ | Time from positive edge on SCL to IRQ signal. | $9 \cdot CK_P$ | | $8 \cdot CK_P$ | | ns |

*Table 97. Timing (16 MHz system clock)*

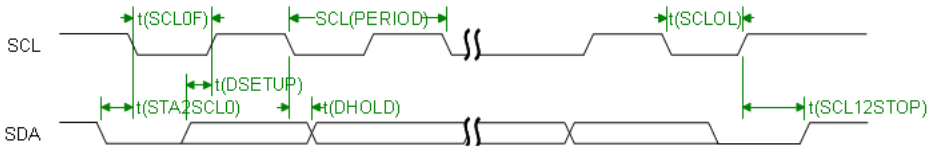



*Figure 65. Timing SCL/SDA*

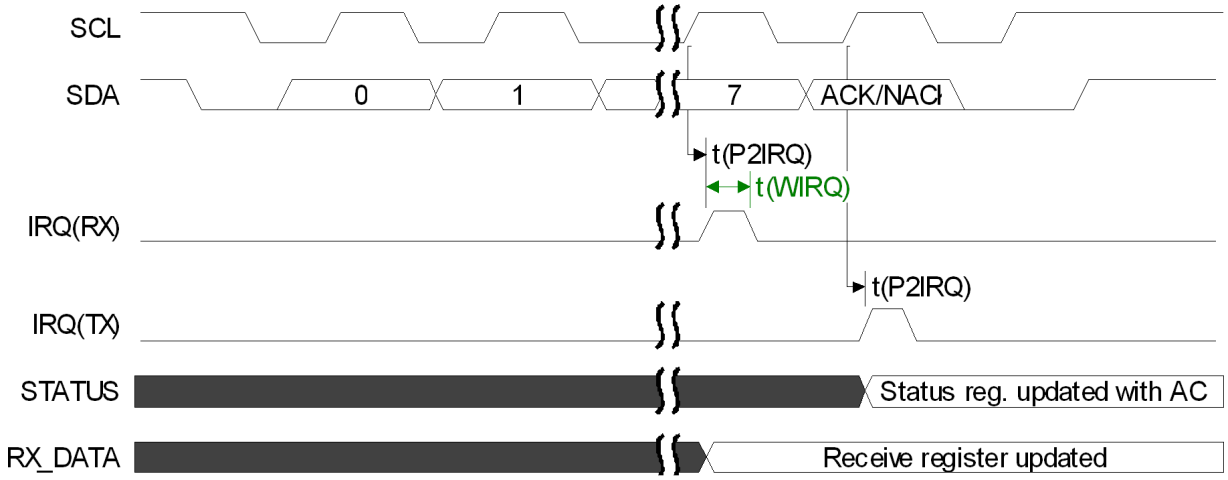



*Figure 66. Interrupt request timing towards MCU*

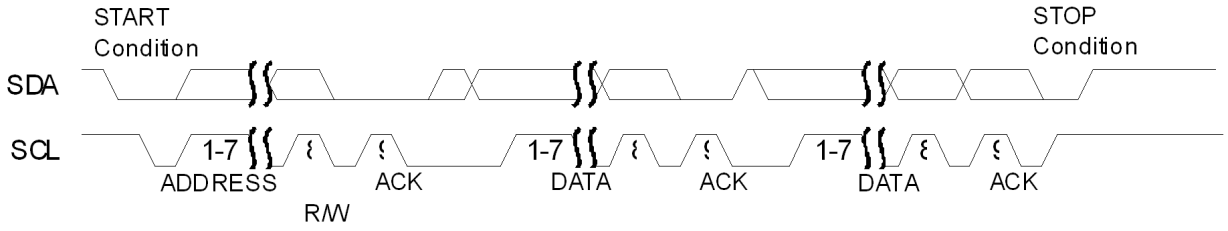



*Figure 67. Complete data transfer*

## 20.3 SFR registers

The following registers control the 2-Wire:

| Address (Hex) | Name/Mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| 0xE2 | W2CON0 | 7:0 | 0x80 | R/W | 2-Wire configuration register 0. |
| | *broadcastEnable* | 7 | 1 | R/W | **Slave only:**<br>1: Respond to the general call address (0x00), as well as the address defined in WIRE2ADR.<br>0: Respond only to the address defined in WIRE2ADR. |

| Address (Hex) | Name/Mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| | *clockStop* | 6 | 0 | R/W | **Slave only:**<br>1: SCL is kept 'low' by the slave between byte transfers. This buys the MCU time to read RX data or write TX data. In TX mode SCL is released $t_{REL}$ after TX data has been written to W2DAT. $t_{REL}$ = 1400 ns in Standard and Fast modes, while $t_{REL}$ = $5 \cdot T_{ckCPU}$ in High-speed mode. In RX mode SCL is released immediately after the RX data is read from W2DAT.<br>**Note:** Update this bit before any transmissions begin.<br>0: The 2-Wire Slave does not alter the clock. |
| | *xStop* | 5 | 0 | R/W | **Master only:**<br>1: Transmit stop condition 1) in RX mode: After the ongoing byte reception is completed; or 2) in TX mode: After any pending TX data is transmitted.<br>**Note:** Do not attempt to clear a stop bit by writing a 0 to it.<br>0: No stop condition to be sent.<br>Cleared when the stop condition is transmitted.<br><br>**Slave only:**<br>1: Disable interrupt when stop condition is detected.<br>0: Enable interrupt when stop condition is detected. |
| | *xStart* | 4 | 0 | R/W | **Master only:**<br>1: Transmit start (repeated start) condition after any pending TX data or stop condition.<br>**Note:** Do not attempt to clear a start bit by writing a 0 to it.<br>0: No start (repeated start) condition to be sent.<br>Cleared when the start (repeated start) condition is transmitted.<br><br>**Slave only:**<br>1: Disable interrupt on address match.<br>0: Enable interrupt on address match. |
| | *clockFrequency* | 3:2 | 00 | R/W | Frequency on SCL.<br>00: Idle.<br>01: 100 kHz (Standard mode). Requires a system clock frequency of at least 4 MHz.<br>10: 400 kHz (Fast mode). Requires a system clock frequency of at least 8 MHz.<br>11: reserved. |
| | *masterSelect* | 1 | 0 | R/W | 1: Master mode selected.<br>0: Slave mode selected. |

| Address (Hex) | Name/Mnemonic | Bit | Reset value | Type | Description |
|---|---|---|---|---|---|
| | *wire2Enable* | 0 | 0 | R/W | 1: 2-Wire is enabled. The clock to the 2-Wire core functionality is running. An 2-Wire transfer can be initiated by the MCU via the 8051 SFR Bus (TX).<br>**Note:** This bit must be set in a separate write operation before any other 2-Wire configuration bits are written.<br>0: 2-Wire is disabled. The clock to the 2-Wire core functionality stands still. |
| 0xE1 | W2CON1 | 5:0 | 0x00 | R/W | 2-Wire configuration register 1/status register. |
| | *maskIrq* | 5 | 0 | R/W | 1: Disable all interrupts.<br>0: Enable all interrupts (not masked otherwise).<br>**Note:** Update this bit before any transmissions begin. |
| | *broadcast* | 4 | | R | **Slave only:**<br>1: The last received address was a broadcast address (0x00).<br>0: The last received address was not a broadcast address.<br>Cleared when reading W2CON1. |
| | *stop* | 3 | | R | **Slave only:**<br>1: Interrupt caused by stop condition.<br>0: No interrupt caused by stop condition.<br>Cleared when reading W2CON1. |
| | *addressMatch* | 2 | | R | **Slave only:**<br>1: Interrupt caused by address match.<br>0: No interrupt caused by address match.<br>Cleared when reading W2CON1. |
| | *ack_n* | 1 | | R | **TX mode only:**<br>1: Not-acknowledge (NACK).<br>0: Acknowledge (ACK).<br>This bit contains the acknowledge 2-Wire has received after the last transfer.<br>Cleared when reading W2CON1. |
| | *dataReady* | 0 | | R | 1: Interrupt caused by byte transmitted/received.<br>0: No interrupt caused by byte transmitted/ received.<br>Cleared when reading W2CON1. |
| 0xD9 | W2SADR | 6:0 | 0x00 | R/W | 2-Wire Slave address register.<br>The address the 2-Wire reacts upon in slave mode. |
| 0xDA | W2DAT | 7:0 | 0x00 | R/W | 2-Wire data register.<br>Accesses TX (write) and RX (read) buffers, both one byte deep. |

*Table 98. Wire registers*

The 2-Wire is enabled by setting W2CON0[0] to '1'. W2CON0[1] decides whether it shall act as Master or Slave. The baudrate is defined by W2CON0[3:2].

**Note:** The 2-Wire needs a system clock frequency of at least 4 MHz to function correctly in Standard mode. In Fast mode, the system clock frequency must be at least 8 MHz.

# 21 ADC

nRF24LE1 OTP includes a general purpose ADC with up to 14 input channels, depending on package variant. The ADC contains an internal 1.2V reference, but can also be used with external reference or full scale range equal to VDD. It can be operated in a single step mode with sampling under software control, or a continuous conversion mode with a programmable sampling rate.

## 21.1 Features

- 6, 8, 10 or 12 bit resolution
- Up to 14 input channels
- Single ended or differential input
- Full-scale range set by internal reference, external reference or VDD
- Single step mode with conversion time down to 3 µs
- Continuous mode with 2, 4, 8 or 16 kbps sampling rate
- Low current consumption; only 0.1 mA at 2 kbps
- Mode for measuring supply voltage

## 21.2 Block diagram

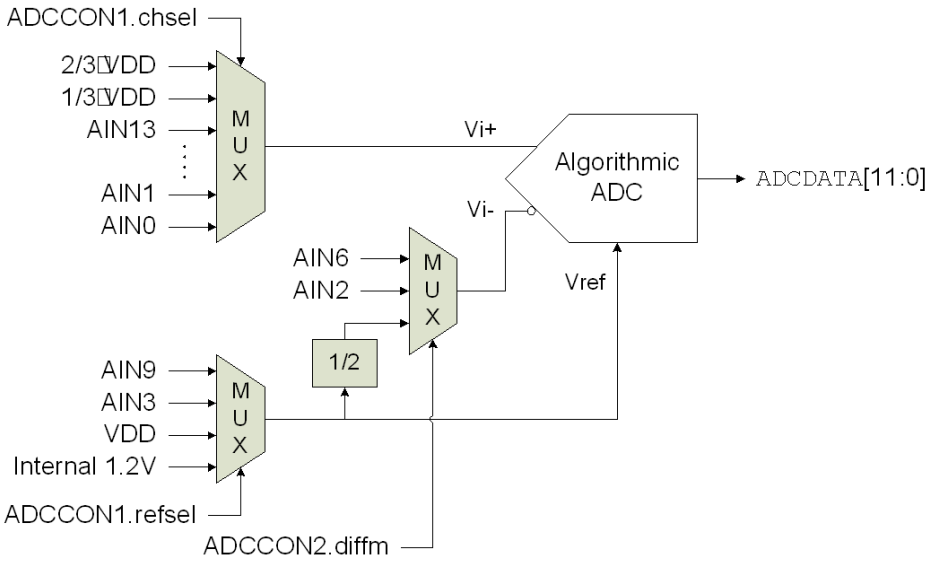



*Figure 68. Block diagram of ADC*

## 21.3 Functional description

### 21.3.1 Activation

A write operation to the `ADCCON1` register automatically starts a conversion, provided that the `pwrup` bit is set. If the ADC is busy, the unfinished conversion is aborted and a new one initiated. Write operations to `ADCCON2` and `ADCCON3` do not start a conversion. It is not advisable to change these registers while the ADC is busy.

### 21.3.2 Input selection

The ADC supports up to 14 external and 2 internal input channels, and can be configured for single ended or differential measurements. Input channel is selected with the `chsel` bits. Channel 0 to 13 (AIN0-AIN13) are external inputs applied through port pins. Channel 14 and 15 are internally generated inputs equal to 1/3×VDD and 2/3×VDD, respectively. The number of available external inputs depends on package variant. See chapter 17 on page 127 for a description of the mapping between port pins and AIN0-AIN13.

Configure `diffm` to select between single ended and differential mode. In single ended mode the input range is from 0V up to the reference voltage $V_{REF}$, in differential mode from $–V_{REF}/2$ to $+V_{REF}/2$. Either AIN2 or AIN6 can be used as inverting input in differential mode. Non-inverting input is selected with `chsel`. The common-mode voltage must be between 25% and 75% of VDD.

The internally generated 1/3 VDD and 2/3 VDD inputs may be used for supply voltage measurement or calibration of offset and gain error.

### 21.3.3 Reference selection

Full-scale range is controlled by the `refsel` bits. It can be set by an internal bandgap reference (nominally 1.2V), external reference or VDD. The external reference voltage is applied on AIN3 or AIN9, and must be between 1.15V and 1.5V. It is buffered by an on-chip CMOS buffer with very high input impedance.

### 21.3.4 Resolution

The ADC can do 6, 8, 10 or 12 bit conversions. Configure the `resol` bits to set resolution.

### 21.3.5 Conversion modes

The `cont` bit selects between single step and continuous conversion mode. In single step mode the ADC performs one conversion and then stops. In continuous mode it runs continuously with a programmable sampling rate.

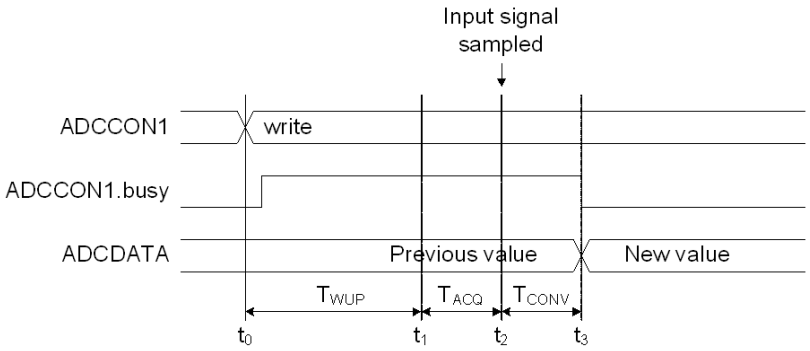



*Figure 69. Timing diagram for single step conversion*

Figure 69. illustrates the timing of a single step conversion. The conversion is started by writing to the `ADCCON1` register. The busy bit is set to '1' four 16 MHz clock cycles afterwards and cleared again when the conversion result becomes available in the `ADCDATH`/`ADCDATL` registers. An interrupt to the MCU (ADCIRQ) is also generated at the end of conversion.

By default the ADC is powered down immediately after end of conversion. It can also be configured to enter standby mode after end of conversion, and proceed to a full power-down after a programmable delay. This shortens the wakeup time if a new conversion is initiated before the power-down delay has elapsed. Configure the `rate` bits to choose behavior. Note that this automatic power-down will not clear the `pwrup` bit, and the selected port pin(s) will continue to be configured as analog input(s) until the `pwrup` bit is cleared from software.

A conversion can be divided into three phases: wakeup, signal acquisition and conversion. The wakeup time depends on whether the ADC was powered down or in standby mode before initiation. If it was powered down it needs $T_{WUP}$ = 15 µs to wake up. Otherwise, $T_{WUP}$ = 0.6 µs.

The sampling capacitor is switched to the analog input at the end of the wakeup phase (at $t = t_1$) and remains connected throughout the acquisition phase. The sample is acquired at the end of the acquisition phase (at $t = t_2$). The duration of this phase is $T_{ACQ}$ = 0.75, 3, 12 or 36 µs, selected with the `tacq` bits.

The final phase is the time used by the ADC to convert the analog sample into a N-bit digital representation. This time depends on the selected resolution: $T_{CONV}$ = 1.7, 1.9, 2.1 and 2.3 µs for 6, 8, 10 and 12-bit conversions, respectively. Table 99. shows the total conversion time for all combinations of acquisition time and resolution.

| $T_{ACQ}$ | Starting from standby mode | | | | Starting from power-down | | | | Unit |
|---|---|---|---|---|---|---|---|---|---|
| | 6-bit | 8-bit | 10-bit | 12-bit | 6-bit | 8-bit | 10-bit | 12-bit | |
| 0.75 | 3.0 | 3.2 | 3.4 | 3.6 | 17.4 | 17.6 | 17.8 | 18.0 | µs |
| 3 | 5.3 | 5.4 | 5.6 | 5.8 | 19.7 | 19.9 | 20.1 | 20.3 | µs |
| 12 | 14.3 | 14.4 | 14.6 | 14.8 | 28.7 | 28.9 | 29.1 | 29.3 | µs |
| 36 | 38.3 | 38.4 | 38.6 | 38.8 | 52.7 | 52.9 | 53.1 | 53.3 | µs |

*Table 99. Single step conversion time*

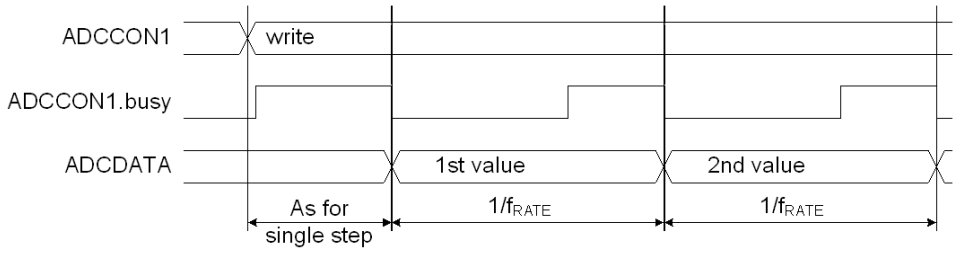



*Figure 70. Timing diagram for continuous conversion*

Continuous conversion mode operates exactly like single step, except that new conversions are started automatically at a programmable rate. The converter enters power down mode between conversions to minimize current consumption. Sampling rate is specified with the `rate` bits, and can be 2, 4, 8 or 16 ksps.

### 21.3.6 Output data coding

The ADC uses straight binary coding for single ended conversions. An input voltage ≤ 0V is represented by all zeroes (000...00), and an input voltage ≥ $V_{REF}$ by all ones (111...11). Midscale is represented by a one followed by all zeroes (100...00).

Differential conversions use offset binary coding. A differential input voltage ≤ $-V_{REF}/2$ is represented by all zeroes (000...00), and an input voltage ≥ $+V_{REF}/2$ by all ones (111...11). Zero-scale is represented by a one followed by all zeroes (100...00).

The ADCCON3 register contains 3 overflow bits; uflow is set when the ADC is under ranged, oflow is set when the ADC is over ranged, while range is the logical OR of uflow and oflow.

### 21.3.7 Driving the analog input

The analog input pin draws a small current transient each time the internal sampling capacitor is switched to the input at the beginning of the acquisition phase. It is important that the circuitry driving the input settles from this disturbance before the conversion is started. Unless the input is driven by a sufficiently fast op-amp, it may be necessary to choose a longer than minimum acquisition time to ensure proper settling. But note that this extends the conversion time accordingly, and hence the time delay before the ADC returns to power-down mode. If current consumption is important, the acquisition time should be made as short as possible.

Figure 71. gives recommendations for acquisition time as a function of source resistance and capacitance, assuming a passive signal source and 10-bit conversions. If for instance the source resistance is 100kΩ and the off-chip capacitance on the analog input pin is 10pF, it can be read out from the figure that the recommended acquisition time is 12 µs.

Alternatively, a large capacitor may be connected between the analog input pin and VSS. It will supply all the current to the sampling capacitor, so that minimum acquisition time can be used even if the source resistance is high. A capacitor value of 33nF or higher is recommended.

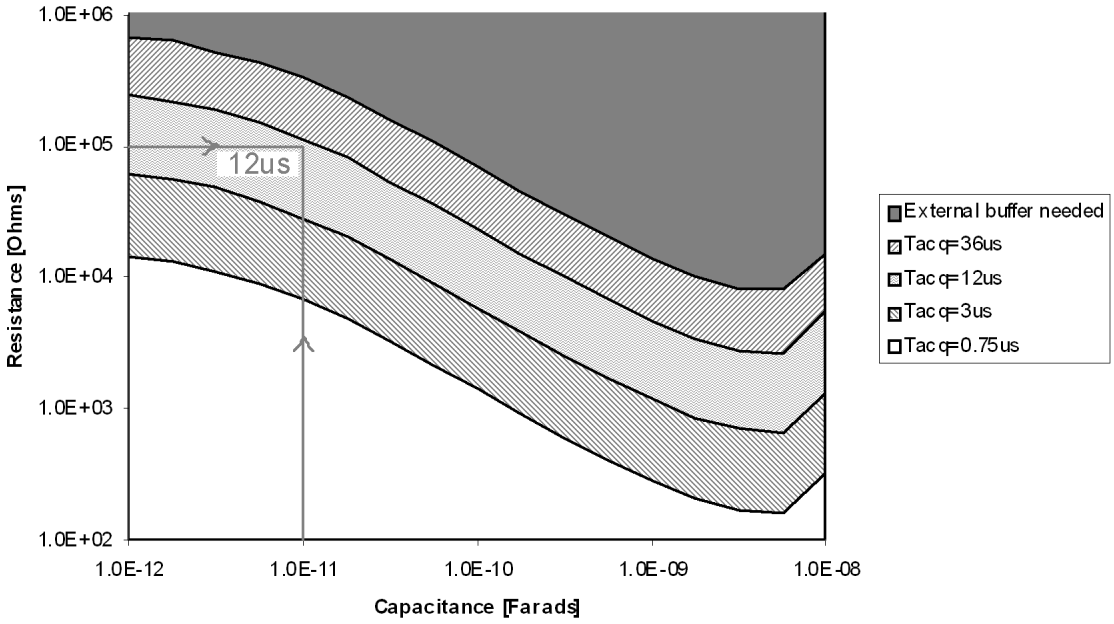



*Figure 71. Recommended acquisition time versus source resistance and capacitance (10-bit conversions)*

### 21.3.8 SFR registers

The ADC is interfaced to the MCU through five registers; ADCCON1, ADCCON2, ADCCON3, ADCDATH and ADCDATL. ADCCON1, ADCCON2 and ADCCON3 contain configuration settings and status bits. The conversion result is contained in the ADCDATH and ADCDATL registers.

| Addr | Bit | Name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| 0xD3 | 7 | pwrup | RW | Power-up control:<br>0: Power down ADC<br>1: Power up ADC and configure selected pin(s) as analog input | |
| | 6 | busy | R | ADC busy flag:<br>0: No conversion in progress<br>1: Conversion in progress<br>The busy bit is cleared when a conversion result becomes available in the ADCDATH / ADCDATL registers. | |
| | 5:2 | chsel | RW | Input channel select:<br>0000: AIN0<br>0001: AIN1<br>:<br>1101: AIN13<br>1110: 1/3×VDD<br>1111: 2/3×VDD | |
| | 1:0 | refsel | RW | Reference select:<br>00: Internal 1.2V reference<br>01: VDD<br>10: External reference on AIN3<br>11: External reference on AIN9 | |

*Table 100. ADCCON1 register*

| Addr | Bit | Name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| 0xD2 | 7:6 | diffm | RW | Selects single ended or differential mode:<br>00: Single ended<br>01: Differential with AIN2 as inverting input<br>10: Differential with AIN6 as inverting input<br>11: Not used | |
| | 5 | cont | RW | Selects single step or continuous conversion mode:<br>0: Single step conversion<br>1: Continuous conversion with sampling rate defined by **rate** | |
| | 4:2 | rate | RW | Selects sampling rate in continuous conversion mode:<br>000: 2 ksps<br>001: 4 ksps<br>010: 8 ksps<br>011: 16 ksps<br>1XX: Reserved<br>Selects power-down delay in single-step mode:<br>000: 0 µs<br>001: 6 µs<br>010: 24 µs<br>011: Infinite (clear **pwrup** to power down)<br>1XX: Reserved | |

| Addr | Bit | Name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| | 1:0 | tacq | RW | Duration of input acquisition window ($T_{ACQ}$):<br>00: 0.75 µs<br>01: 3 µs<br>10: 12 µs<br>11: 36 µs | |

*Table 101. ADCCON2 register*

| Addr | Bit | Name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| 0xD1 | 7:6 | resol | RW | ADC resolution:<br>00: 6 bits<br>01: 8 bits<br>10: 10 bits<br>11: 12 bits | |
| | 5 | rljust | RW | Selects left or right justified data in ADCDATH / ADCDATL:<br>0: Left justified data<br>1: Right justified data | |
| | 4 | uflow | R | ADC underflow when set (conversion result is all zeroes) | |
| | 3 | oflow | R | ADC overflow when set (conversion result is all ones) | |
| | 2 | range | R | ADC overflow or underflow when set (equals **oflow** OR **uflow**) | |
| | 1:0 | - | - | Not used | |

*Table 102. ADCCON3 register*

| Addr | Bit | Name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| 0xD4 | 7:0 | - | R | Most significant byte of left or right justified ADCDATA<br>(see Table 105.) | |

*Table 103. ADCDATH register*

| Addr | Bit | Name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| 0xD5 | 7:0 | - | R | Least significant byte of left or right justified ADCDATA<br>(see Table 105.) | |

*Table 104. ADCDATL register*

| rljust | resol | ADCDATH[7:0] | | ADCDATL[7:0] |
|---|---|---|---|---|
| 0 | 00 | ADCDATA[5:0] | | 0 |
| 0 | 01 | ADCDATA[7:0] | | 0 |
| 0 | 10 | ADCDATA[9:0] | | 0 |
| 0 | 11 | ADCDATA[11:0] | | 0 |
| 1 | 00 | 0 | | ADCDATA[5:0] |
| 1 | 01 | 0 | | ADCDATA[7:0] |
| 1 | 10 | 0 | | ADCDATA[9:0] |
| 1 | 11 | 0 | | ADCDATA[11:0] |

*Table 105. Left or right justified output data*

# 22 Analog comparator

The analog comparator is used as a wakeup source. It allows a system wakeup to be triggered by the voltage level of a differential or single ended analog input applied through the port pins. The comparator has very low current consumption, and is operational in the register retention mode and memory retention mode timer on.

## 22.1 Features

- Low current consumption (0.75µA typical)
- Differential or single-ended input
- Single-ended threshold programmable to 25%, 50%, 75% or 100% of VDD or an arbitrary reference voltage from pin
- 14-channel input multiplexer
- Rail-to-rail input voltage range
- Programmable output polarity

## 22.2 Block diagram

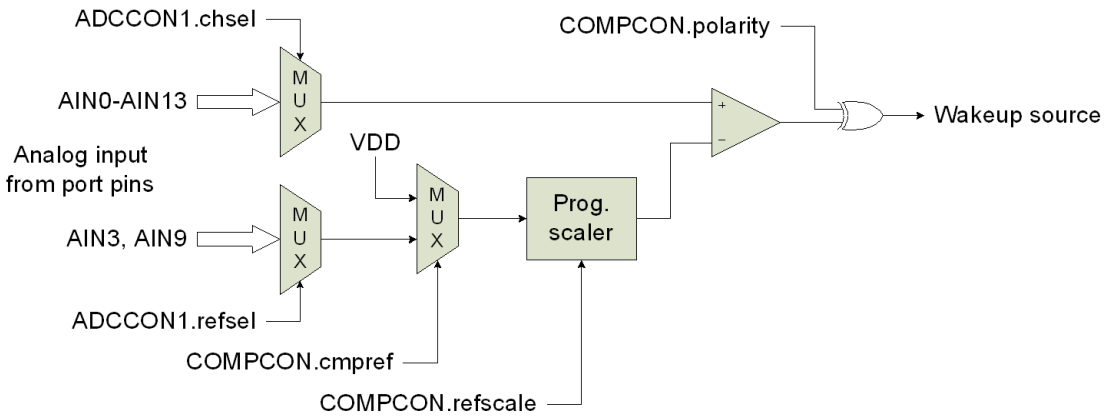



*Figure 72. Block diagram of analog comparator*

## 22.3 Functional description

### 22.3.1 Activation

Enable the comparator by setting the `enable` bit in the `COMPCON` register. The comparator is activated when the system enters register retention mode or memory retention mode timer on. It is not operational in any other system modes. In order to use the comparator a 32 kHz clock source must also be activated.

### 22.3.2 Input selection

Depending on package variant, one out of up to 14 different port pins may be used to apply a voltage to the non-inverting comparator input. Configure the `chsel` bits in the `ADCCON1` register to select one of AIN0 through AIN13 as input. Note that '1110' and '1111' are illegal values; if these are specified the non-inverting comparator input will float. The `pwrup` bit in ADCCON1 does not have to be set.

Refer to chapter 17 on page 127 for a description of the mapping between port pins and AIN0-AIN13.

### 22.3.3 Reference selection

The inverting comparator input can be connected to 25%, 50%, 75% or 100% of either VDD or an arbitrary reference voltage from AIN3 or AIN9. Configure the `refscale` bits in `COMPCON` to select scaling factor. To use VDD as a reference, set `cmpref` to '0'. To use an arbitrary reference, set `cmpref` to '1' and configure `refsel` in ADCCON1 to choose between AIN3 and AIN9 as input pin for the reference. Note that '00' and '01' are illegal values for `refsel`; if these are specified the inverting comparator input will float.

Differential input mode is configured by setting `refscale` to 100% and choosing AIN3 or AIN9 as inverting input.

### 22.3.4 Output polarity

The polarity of the comparator output is programmable. The default behavior is that a wakeup is triggered when the non-inverting input rises above the inverting input. However, if the `polarity` bit is set a wakeup is triggered when the non-inverting input drops below the inverting input.

### 22.3.5 Input voltage range

The input voltage range on AIN0-AIN13 is from VSS to VDD+100mV. However, the input voltage must never exceed 3.6V.

### 22.3.6 Configuration examples

| Wakeup criterion | ADCCON1 | | COMPCON | | |
|---|---|---|---|---|---|
| | chsel | refsel | polarity | refscale | cmpref |
| AIN0 > 0.25×VDD | 0000 | XX | 0 | 00 | 0 |
| AIN13 < 0.5×VDD | 1101 | XX | 1 | 01 | 0 |
| AIN2 > 0.75×AIN3 | 0010 | 10 | 0 | 10 | 1 |
| AIN3 < AIN9 | 0011 | 11 | 1 | 11 | 1 |

*Table 106. Configuration examples*

### 22.3.7 Driving the analog input

The comparator has a switched capacitor input clocked at 32 kHz. It is recommended to connect a 330pF bypass capacitor between the analog input pin(s) and VSS. This reduces voltage transients introduced by the switching. The capacitor may be omitted if the signal source has an output resistance smaller than 100kΩ. The input bias current of the comparator is typically below 100nA.

### 22.3.8 SFR registers

The comparator is interfaced through two registers. ADCCON1 configures the multiplexing of external inputs. Other functions are controlled by the COMPCON register.

| Addr | Bit | Name | RW | Function | Reset value: 0x00 |
|---|---|---|---|---|---|
| 0xDB | 7:5 | - | - | Not used | |
| | 4 | polarity | RW | Output polarity:<br>0: Non-inverting<br>1: Inverting | |
| | 3:2 | refscale | RW | Reference voltage scaling:<br>00: 25%<br>01: 50%<br>10: 75%<br>11: 100% | |
| | 1 | cmpref | RW | Reference select:<br>0: VDD<br>1: External reference on AIN3 or AIN9 | |
| | 0 | enable | RW | Enable/disable comparator:<br>0: Disable comparator<br>1: Enable comparator and configure selected pin(s) as analog input | |

*Table 107. COMPCON register*

# 23 PWM

The nRF24LE1 OTP includes a two channel Pulse-Width Modulation (PWM) module. The two channels (PWM0 and PWM1) share a common programmable frequency and resolution register and have an individually controlled duty cycle, as described in section 23.3 and each channel is available at output port pins PWM0 and PWM1.

## 23.1 Features

- Two-channel output
- Frequency-range from 4 kHz to 254 kHz (when MCU is clocked at 16 MHz)
- Compact control using few registers for enabling, length-setting and prescaler

## 23.2 Block diagram

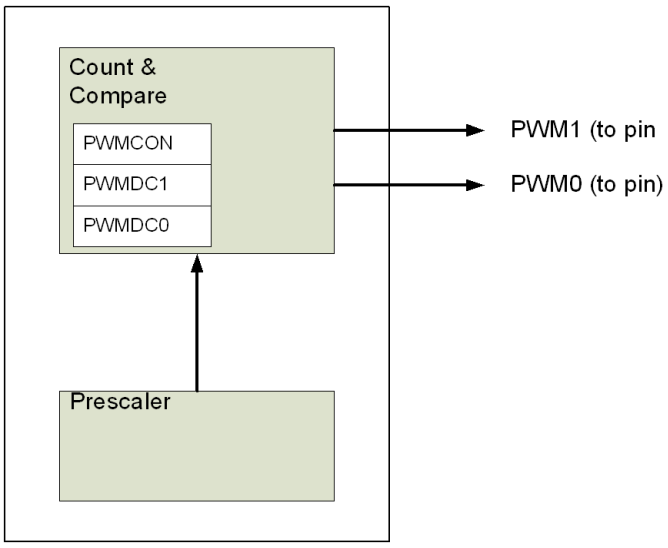



*Figure 73. Block diagram of PWM*

## 23.3 Functional description

The nRF24LE1 OTP PWM is a two-channel PWM with a three register interface. The first register, `PWMCON`, enables the PWM function and sets the PWM period length, which is the number of clock cycles for one PWM period, as shown in Table 108. on page 170 The registers, `PWMDC0` and `PWMDC1`, control the duty cycle for each PWM channel. When one of these registers is written, the corresponding PWM signal changes immediately to the new value. This can result in four transitions within one PWM period, but the transition period will always have a “DC value” between the old sample and the new sample.

The following table shows how the PWM frequency (or period length) and the PWM duty cycle are controlled by the PWM SFR registers. PWM frequency range is approximately 4 kHz–254 kHz.

| PWMCON 7:6 (Number of bits) | PWM frequency | PWM duty cycle |
|---|---|---|
| 00 (5) | $Cclk \bullet \frac{1}{31(PWMCON[5:2]+1)}$ | $\frac{PWMDC[4:0]}{31}$ |
| 01 (6) | $Cclk \bullet \frac{1}{63(PWMCON[5:2]+1)}$ | $\frac{PWMDC[5:0]}{63}$ |
| 10 (7) | $Cclk \bullet \frac{1}{127(PWMCON[5:2]+1)}$ | $\frac{PWMDC[6:0]}{127}$ |
| 11 (8) | $Cclk \bullet \frac{1}{255(PWMCON[5:2]+1)}$ | $\frac{PWMDC}{255}$ |

*Table 108. PWM frequency and duty-cycle setting*

The PWM is controlled by SFR 0xB2, 0XA1 and 0xA2.

| Addr SFR (HEX) | R/W | #bit | Reset (HEX) | Name | Function |
|---|---|---|---|---|---|
| 0xB2 | R/W | 8 | 0 | PWMCON | PWM control register<br>7–6: Period length select<br>00: Period length is 5 bit<br>01: Period length is 6 bit<br>10: Period length is 7 bit<br>11: Period length is 8 bit<br>5–2: PWM frequency pre-scale factor (see table above)<br>1: Select output port pin for pwm1:<br>0: pwm1 disabled<br>1: pwm1 enabled and available on port<br>0: Select output port pin for pwm0:<br>0: pwm0 disabled<br>1: pwm0 enabled and available on port |
| 0xA1 | R/W | 8 | 0 | PWMDC0 | PWM duty cycle for channel 0 (5 to 8 bits according to period length) |
| 0xA2 | R/W | 8 | 0 | PWMDC1 | PWM duty cycle for channel 1 (5 to 8 bits according to period length) |

*Table 109. PWM control registers*

# 24 Absolute maximum ratings

Maximum ratings are the extreme limits to which the nRF24LE1 OTP can be exposed without permanently damaging it. Exposure to absolute maximum ratings for prolonged periods of time may affect device reliability.

The device is not guaranteed to operate properly at the maximum ratings.

| Operating conditions | Minimum | Maximum | Units |
|---|---|---|---|
| **Supply voltages** | | | |
| VDD | -0.3 | +3.6 | V |
| VSS | | 0 | V |
| **I/O pin voltages** | | | |
| PROG/VPP | -0.3 | +7.0 | V |
| Other I/O | -0.3 | VDD +0.3, max 3.6 | V |
| **Temperatures** | | | |
| Operating temperature | -40 | +85 | °C |
| Storage temperature | -40 | +125 | °C |

*Table 110. Absolute maximum ratings*

**Note:** Stress exceeding one or more of the limiting values may cause permanent damage to the device.

**Attention!**
Observe precaution for handling
Electrostatic Sensitive Device.

HBM (Human Body Model): Class 2

# 25 Operating condition

| Symbol | Parameter | Notes | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| VDD | Supply voltage | | 1.9 | 3.0 | 3.6 | V |
| $t_{R_VDD}$ | Supply rise time (0V to 1.9V) | a | 1 µs | | 50 ms | µs and ms |
| $T_A$ | Operating temperature | | -40 | | +85 | °C |

a. The on-chip power-on reset circuitry may not function properly for rise times outside the specified interval.

*Table 111. Operating conditions*

# 26 Electrical specifications

This section contains electrical and timing specifications.

Conditions: VDD = 3.0V, $T_A$ = –40ºC to +85ºC (unless otherwise noted)

| Symbol | Parameter (condition) | Notes | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $V_{IH}$ | Input high voltage | | 0.7×VDD | | VDD | V |
| $V_{IL}$ | Input low voltage | | VSS | | 0.3× VDD | V |
| $V_{OH}$ | Output high voltage (std. drive, 0.5mA) | | VDD - 0.3 | | VDD | V |
| $V_{OH}$ | Output high voltage (high-drive, 5mA) | a | VDD - 0.3 | | VDD | V |
| $V_{OL}$ | Output low voltage (std. drive, 0.5mA) | | VSS | | 0.3 | V |
| $V_{OL}$ | Output low voltage (high-drive, 5mA) | a | VSS | | 0.3 | V |
| $R_{PU}$ | Pull-up resistance | | 11 | 13 | 16 | kΩ |
| $R_{PD}$ | Pull-down resistance | | 11 | 13 | 16 | kΩ |

a. Maximum number of pins with 5mA high drive is 3.

*Table 112. Digital inputs/outputs*

| Symbol | Parameter (condition) | Notes | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | **General RF conditions** | | | | | |
| $f_{OP}$ | Operating frequency | a | 2400 | | 2525 | MHz |
| $PLL_{res}$ | PLL Programming resolution | | | 1 | | MHz |
| $f_{XTAL}$ | Crystal frequency | | | 16 | | MHz |
| $\Delta f_{250}$ | Frequency deviation @ 250 kbps | | | ±160 | | kHz |
| $\Delta f_{1M}$ | Frequency deviation @ 1 Mbps | | | ±160 | | kHz |
| $\Delta f_{2M}$ | Frequency deviation @ 2 Mbps | | | ±320 | | kHz |
| $R_{GFSK}$ | Air data rate | b | 250 | | 2000 | kbps |
| $F_{CHANNEL\ 1M}$ | Non-overlapping channel spacing @ 250 kbps/1 Mbps) | c | | 1 | | MHz |
| $F_{CHANNEL\ 2M}$ | Non-overlapping channel spacing @ 2 Mbps | | | 2 | | MHz |
| | **Transmitter operation** | | | | | |
| $P_{RF}$ | Maximum output power | d | | 0 | +4 | dBm |
| $P_{RFC}$ | RF power control range | | 16 | 18 | 20 | dB |
| $P_{RFCR}$ | RF power accuracy | | | | ±4 | dB |
| $P_{BW2}$ | 20dB bandwidth for modulated carrier (2 Mbps) | | | 1800 | 2000 | kHz |
| $P_{BW1}$ | 20dB bandwidth for modulated carrier (1 Mbps) | | | 950 | 1100 | kHz |
| $P_{BW250}$ | 20dB bandwidth for modulated carrier (250 kbps) | | | 700 | 800 | kHz |
| $P_{RF1.2}$ | $1^{st}$ Adjacent Channel Transmit Power 2 MHz (2Mbps) | | | | -20 | dBc |
| $P_{RF2.2}$ | $2^{nd}$ Adjacent Channel Transmit Power 4 MHz (2 Mbps) | | | | -45 | dBc |
| $P_{RF1.1}$ | $1^{st}$ Adjacent Channel Transmit Power 1 MHz (1 Mbps) | | | | -20 | dBc |

| Symbol | Parameter (condition) | Notes | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $P_{RF2.1}$ | 2nd Adjacent Channel Transmit Power 2 MHz (1 Mbps) | | | | -40 | dBc |
| $P_{RF1.250}$ | 1st Adjacent Channel Transmit Power 1 MHz (250 kbps) | | | | -25 | dBc |
| $P_{RF2.250}$ | 2nd Adjacent Channel Transmit Power 2 MHz (250 kbps) | | | | -40 | dBc |
| | **Receiver operation** | | | | | |
| $RX_{MAX}$ | Maximum received signal at < 0.1% BER | | | 0 | | dBm |
| $RX_{SENS}$ | Sensitivity (0.1% BER) @ 2 Mbps | | | -82 | | dBm |
| $RX_{SENS}$ | Sensitivity (0.1% BER) @ 1 Mbps | | | -85 | | dBm |
| $RX_{SENS}$ | Sensitivity (0.1% BER) @ 250 kbps | e | | -94 | | dBm |
| **RX selectivity according to ETSI EN 300 440-1 V1.3.1 (2001-09) page 27** | | | | | | |
| $C/I_{CO}$ | C/I co-channel (2 Mbps) | | | 7 | | dBc |
| $C/I_{1ST}$ | 1st ACS (Adjacent Channel Selectivity), C/I 2MHz (2 Mbps) | | | 3 | | dBc |
| $C/I_{2ND}$ | 2nd ACS, C/I 4 MHz (2 Mbps) | | | -17 | | dBc |
| $C/I_{3RD}$ | 3rd ACS, C/I 6 MHz (2 Mbps) | | | -21 | | dBc |
| $C/I_{Nth}$ | Nth ACS, C/I $f_i$ > 12 MHz (2 Mbps) | f | | -40 | | dBc |
| $C/I_{Nth}$ | Nth ACS, C/I $f_i$ > 36 MHz (2 Mbps) | | | -48 | | dBc |
| $C/I_{CO}$ | C/I co-channel (1 Mbps) | | | 9 | | dBc |
| $C/I_{1ST}$ | 1st ACS, C/I 1 MHz (1 Mbps) | | | 8 | | dBc |
| $C/I_{2ND}$ | 2nd ACS, C/I 2 MHz (1 Mbps) | | | -20 | | dBc |
| $C/I_{3RD}$ | 3rd ACS, C/I 3 MHz (1 Mbps) | | | -30 | | dBc |
| $C/I_{Nth}$ | Nth ACS, C/I $f_i$ > 6 MHz (1 Mbps) | | | -40 | | dBc |
| $C/I_{Nth}$ | Nth ACS, C/I $f_i$ > 25 MHz (1 Mbps) | f | | -47 | | dBc |
| $C/I_{CO}$ | C/I co-channel (250 kbps) | | | 12 | | dBc |
| $C/I_{1ST}$ | 1st ACS, C/I 1 MHz (250 kbps) | | | -12 | | dBc |
| $C/I_{2ND}$ | 2nd ACS, C/I 2 MHz (250 kbps) | | | -33 | | dBc |
| $C/I_{3RD}$ | 3rd ACS, C/I 3 MHz (250 kbps) | | | -38 | | dBc |
| $C/I_{Nth}$ | Nth ACS, C/I $f_i$ > 6 MHz (250 kbps) | | | -50 | | dBc |
| $C/I_{Nth}$ | Nth ACS, C/I $f_i$ > 25 MHz (250 kbps) | f | | -60 | | dBc |
| **RX selectivity with nRF24L01 equal modulation on interfering signal (Pin = -67dBm for wanted signal)** | | | | | | |
| $C/I_{CO}$ | C/I co-channel (2 Mbps) (modulated carrier) | | | 11 | | dBc |
| $C/I_{1ST}$ | 1st ACS (Adjacent Channel Selectivity), C/I 2MHz (2 Mbps) | | | 4 | | dBc |
| $C/I_{2ND}$ | 2nd ACS, C/I 4 MHz (2 Mbps) | | | -18 | | dBc |
| $C/I_{3RD}$ | 3rd ACS, C/I 6 MHz (2 Mbps) | | | -24 | | dBc |
| $C/I_{Nth}$ | Nth ACS, C/I $f_i$ > 12 MHz (2 Mbps) | | | -40 | | dBc |
| $C/I_{Nth}$ | Nth ACS, C/I $f_i$ > 36 MHz (2 Mbps) | | | -48 | | dBc |
| $C/I_{CO}$ | C/I co-channel (1 Mbps) | | | 12 | | dBc |
| $C/I_{1ST}$ | 1st ACS, C/I 1 MHz (1 Mbps) | | | 8 | | dBc |

| Symbol | Parameter (condition) | Notes | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $C/I_{2ND}$ | $2^{nd}$ ACS, C/I 2 MHz (1 Mbps) | | | -21 | | dBc |
| $C/I_{3RD}$ | $3^{rd}$ ACS, C/I 3 MHz (1 Mbps) | | | -30 | | dBc |
| $C/I_{Nth}$ | $N^{th}$ ACS, C/I $f_i$ > 6 MHz (1 Mbps) | | | -40 | | dBc |
| $C/I_{Nth}$ | $N^{th}$ ACS, C/I $f_i$ > 25 MHz (1 Mbps) | | | -50 | | dBc |
| $C/I_{CO}$ | C/I co-channel (250 kbps) | | | 7 | | dBc |
| $C/I_{1ST}$ | $1^{st}$ ACS, C/I 1 MHz (250 kbps) | | | -12 | | dBc |
| $C/I_{2ND}$ | $2^{nd}$ ACS, C/I 2 MHz (250 kbps) | | | -34 | | dBc |
| $C/I_{3RD}$ | $3^{rd}$ ACS, C/I 3 MHz (250 kbps) | | | -39 | | dBc |
| $C/I_{Nth}$ | $N^{th}$ ACS, C/I $f_i$ > 6 MHz (250 kbps) | | | -50 | | dBc |
| $C/I_{Nth}$ | $N^{th}$ ACS, C/I $f_i$ > 25 MHz (250 kbps) | | | -60 | | dBc |
| **RX intermodulation performance in line with Bluetooth specification version 2.0, $4^{th}$ November 2004, page 42** | | | | | | |
| P_IM(6) @ 2Mbps | Input power of IM interferers at 6 and 12 MHz distance from wanted signal | g | | -42 | | dBm |
| P_IM(8) @ 2Mbps | Input power of IM interferers at 8 and 16 MHz distance from wanted signal | g | | -38 | | dBm |
| P_IM(10) @ 2Mbps | Input power of IM interferers at 10 and 20 MHz distance from wanted signal | g | | -37 | | dBm |
| P_IM(3) @ 1Mbps | Input power of IM interferers at 3 and 6 MHz distance from wanted signal | g | | -36 | | dBm |
| P_IM(4) @1Mbps | Input power of IM interferers at 4 and 8 MHz distance from wanted signal | g | | -36 | | dBm |
| P_IM(5) @1Mbps | Input power of IM interferers at 5 and 10 MHz distance from wanted signal | g | | -36 | | dBm |
| P_IM(3) @ 250kbps | Input power of IM interferers at 3 and 6 MHz distance from wanted signal | g | | -36 | | dBm |
| P_IM(4) @ 250kbps | Input power of IM interferers at 4 and 8 MHz distance from wanted signal | g | | -36 | | dBm |
| P_IM(5) @ 250kbps | Input power of IM interferers at 5 and 10 MHz distance from wanted signal | g | | -36 | | dBm |
| | **ADC** | **h** | | | | |
| DNL | Differential nonlinearity. | i j | | 0.5 | | LSB |
| INL | Integral nonlinearity. | i k | | 0.75 | | LSB |
| $V_{OS}$ | Offset error. | i l | | 1.3 | | % FS |
| $\varepsilon_G$ | Gain error. | i m | | –2.5 | | % FS |
| SINAD | Signal-to-noise and distortion ratio ($f_{IN}$ = 1 kHz, $f_S$ = 16 ksps). | i | | 57 | | dB |
| SFDR | Spurious free dynamic range ($f_{IN}$ = 1 kHz, $f_S$ = 16 ksps). | i | | 65 | | dB |
| $V_{REF_INT}$ | Internal reference voltage | | | 1.2 | | V |
| $TC_{REF_INT}$ | Internal reference voltage drift | | | 300 | | ppm/°C |
| $V_{REF_EXT}$ | External reference voltage | | 1.15 | | 1.5 | V |
| | **Analog comparator** | | | | | |
| $V_{OS}$ | Input offset voltage | n | -50 | | +50 | mV |
| | **Program memory and non-volatile data memory** | | | | | |
| $T_{PROG}$ | Byte write time | | | | 874 | µs |

| Symbol | Parameter (condition) | Notes | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| $T_{RET}$ | Data retention ($T_A$ = +25°C) | | 100 | | | years |
| | | | | | | |
| | **16 MHz crystal** | | | | | |
| $f_{NOM}$ | Nominal frequency (parallel resonant) | | | 16.000 | | MHz |
| $f_{TOL}$ | Frequency tolerance | o p | | | ±60 | ppm |
| $C_L$ | Load capacitance | | | 9 | 16 | pF |
| $C_0$ | Shunt capacitance | | | 3 | 7 | pF |
| ESR | Equivalent series resistance | | | 50 | 100 | Ω |
| $P_D$ | Drive level | | | | 100 | µW |
| $L_S$ | Equivalent series inductance | q | | 30 | | mH |
| | **32 kHz crystal** | | | | | |
| $f_{NOM}$ | Crystal frequency (parallel resonant) | | | 32.768 | | kHz |
| $C_L$ | Load capacitance | | | 9 | 12.5 | pF |
| $C_0$ | Shunt capacitance | | | 1 | 2 | pF |
| ESR | Equivalent series resistance | | | 50 | 80 | kΩ |
| $P_D$ | Drive level | | | | 1 | µW |
| | **16 MHz RC oscillator** | | | | | |
| $f_{NOM}$ | Nominal frequency | | | 16 | | MHz |
| $f_{TOL}$ | Frequency tolerance | | | ±1 | ±5 | % |
| | **32 kHz RC oscillator** | | | | | |
| $f_{NOM}$ | Nominal frequency | | | 32.8 | | kHz |
| $f_{TOL}$ | Frequency tolerance | | | ±1 | ±10 | % |
| | **Power-Fail Comparator** | | | | | |
| $V_{POF}$ | Nominal thresholds (falling supply voltage) | | | 2.1, 2.3, 2.5, 2.7 | | V |
| $V_{TOL}$ | Threshold voltage tolerance | | | | ±5 | % |
| $V_{HYST}$ | Threshold voltage hysteresis | | | 100 | | mV |

a. Usable band is determined by local regulations.
b. Data rate in each burst on-air.
c. The minimum channel spacing is 1 MHz.
d. Antenna load impedance = 15Ω + j88Ω.
e. For 250 kpbs sensitivity, frequencies which are integer multiples of 16 MHz (2400, 2416 and so on) sensitivity are reduced.
f. Narrow Band (In Band) Blocking measurements**:**
0 to ±40 MHz; 1 MHz step size
For Interferer frequency offsets n*2*fxtal, blocking performance is degraded by approximately 5dB compared to adjacent figures.
g. Wanted signal level at Pin = -64dBm. Two interferers with equal input power are used. The interferer closest in frequency is unmodulated, the other interferer is modulated equal with the wanted signal. The input power of interferers where the sensitivity equals BER = 0.1% is presented.
h. VDD is limited from 1.9V to 3.4V in the temperature range of -20°C to -30 °C and from 1.9V to 3.2V in the temperature range -30°C to -40°C.
i. Measured with 10-bit resolution, single-ended input and VDD as reference.
j. DNL given as (abs($DNL_{max}$)+ abs($DNL_{min}$))/2
k. INL given as (abs($INL_{max}$)+ abs($INL_{min}$))/2
l. Defined as the deviation of the first code transition (000...000) to (000...001) from the ideal.
m. Defined as the deviation of the last code transition (111...110) to (111...111) from the ideal, after correcting for offset error.
n. Measured with 100kΩ source resistance and a 330 pF bypass capacitor between the analog input and VSS.

o. Includes initial accuracy, stability over temperature, aging and frequency pulling due to incorrect load capacitance.
p. Frequency regulations in certain regions set tighter requirements on frequency tolerance (for example Japan and South Korea max ±50 ppm).
q. Startup time from power down to standby mode depends on the $L_s$ parameter.

*Table 113. Electrical specifications*

## 26.1 Power consumption

The power consumption is always a sum of the current draw from all modules active at the time of measurement and is very application dependent.

To calculate a peak current draw summarize the currents from all modules that can be active at the same time in a given application.

Conditions: VDD = 3.0V, TA = +25ºC

| Symbol | Parameter (condition) | Notes | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | **Core functions**[a] | | | | | |
| | Deep sleep mode | | | 0.5 | | µA |
| | Memory retention mode, timers off | | | 1.0 | | µA |
| | Memory retention mode, timers on (CLKLF from XOSC32K) | | | 1.6 | | µA |
| | Memory retention mode, timers on (CLKLF from RCOSC32K) | | | 1.8 | | µA |
| | Register retention mode, timers off | | | 2.0 | | µA |
| | Register retention mode, timers on (CLKLF from XOSC32K) | | | 3.0 | | µA |
| | Register retention mode, timers on (CLKLF from RCOSC32K) | | | 3.2 | | µA |
| | Register retention mode, timers on (CLKLF from XOSC32K, XOSC16M running) | | | 0.05 | | mA |
| | Register retention mode, timers on (CLKF synthesized from XOSC16M) | | | 87 | | µA |
| | Standby mode (XOSC16M running) | | | 1 | | mA |
| | Active mode (8 MHz MCU clock, 4 MIPS) | | | 3.8 | | mA |
| | **Peripherals** | | | | | |
| | OTP byte write | | | 3.8 | | mA |
| | RF transceiver in TX mode ($P_{OUT}$ = 0 dBm) | b | | 11.1 | | mA |
| | RF transceiver in TX mode ($P_{OUT}$ = -6 dBm) | | | 8.8 | | mA |
| | RF transceiver in TX mode ($P_{OUT}$ = -12 dBm) | | | 7.3 | | mA |
| | RF transceiver in TX mode ($P_{OUT}$ = -18 dBm) | | | 6.8 | | mA |
| | RF transceiver in TX mode ($P_{OUT}$ = -6 dBm) Average current with ShockBurst™ | c | | 0.12 | | mA |

| Symbol | Parameter (condition) | Notes | Min. | Typ. | Max. | Units |
|---|---|---|---|---|---|---|
| | RF transceiver during TX settling | d | | 7.8 | | mA |
| | RF transceiver in RX mode (2 Mbps) | | | 13.3 | | mA |
| | RF transceiver in RX mode (1 Mbps) | | | 12.9 | | mA |
| | RF transceiver in RX mode (250 kbps) | | | 12.4 | | mA |
| | RF transceiver during RX settling | e | | 8.7 | | mA |
| | ADC when busy | | | 1.5 | | mA |
| | ADC in standby mode | | | 0.6 | | mA |
| | ADC in continuous mode @ 2 ksps (average current) | f | | 0.1 | | mA |
| | Random number generator | | | 0.5 | | mA |
| | Analog comparator | | | 0.8 | | µA |

a. Please note that all pins must be set to inputs, and the pinMode input mode bits (refer to Table 81. on page 138, Table 82. on page 139, Table 83. on page 140, and Table 84. on page 141) must be set to 11 (Digital input buffer off) if the pins are not controlled externally.
b. Antenna load impedance = 15Ω + j88Ω.
c. Average data rate 10 kbps and full packets.
d. Average current consumption for TX startup (130 µs), and when changing mode from RX to TX (130 µs).
e. Average current consumption for RX startup (130 µs), and when changing mode from TX to RX (130 µs).
f. 10-bit resolution, 0.75 µs acquisition time.

*Table 114. Power consumption*

# 27 Mechanical specifications

nRF24LE1-O is packaged in QFN-versions. For information regarding ordering codes and package markings, see Table 120. on page 188.:

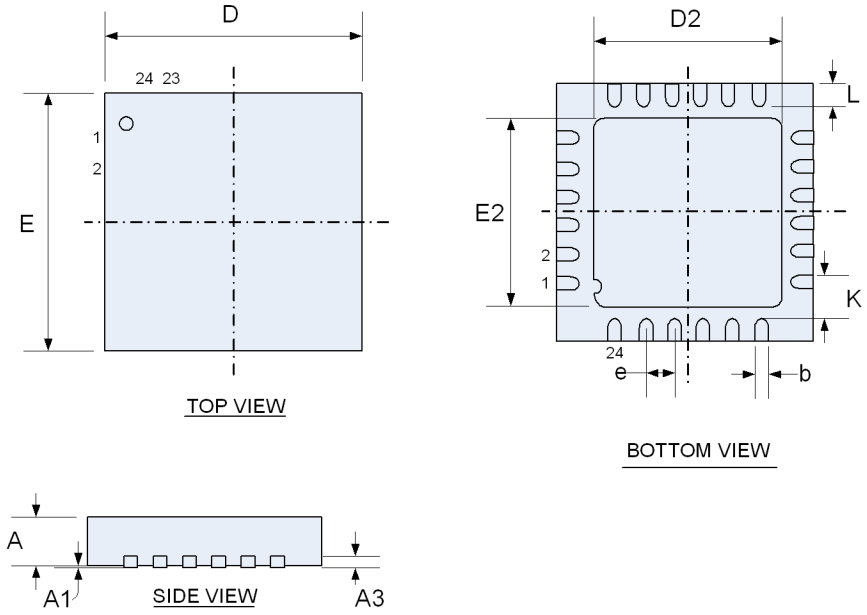



*Figure 74. QFN24 pin 4x4mm*

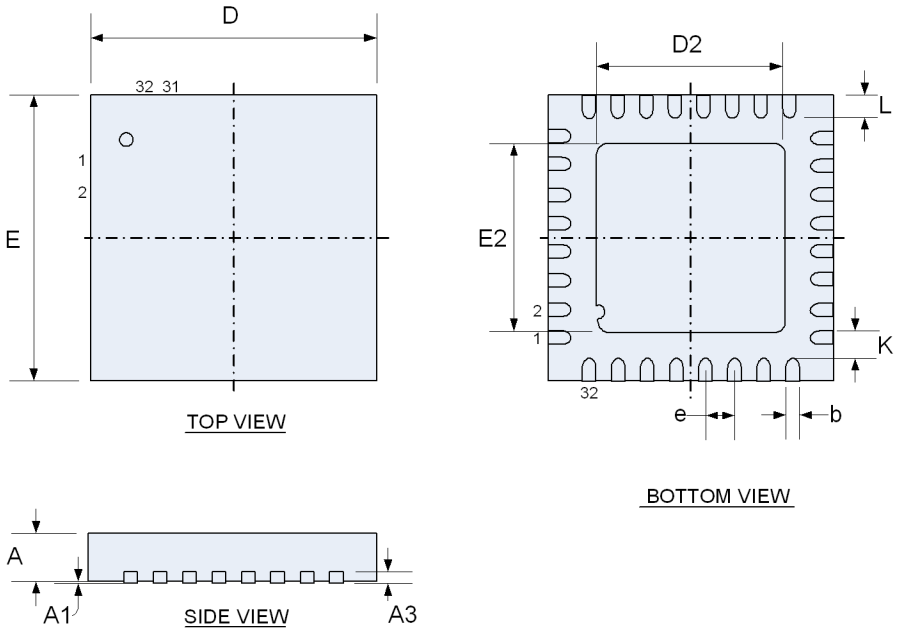



*Figure 75. QFN32 pin 5x5mm*

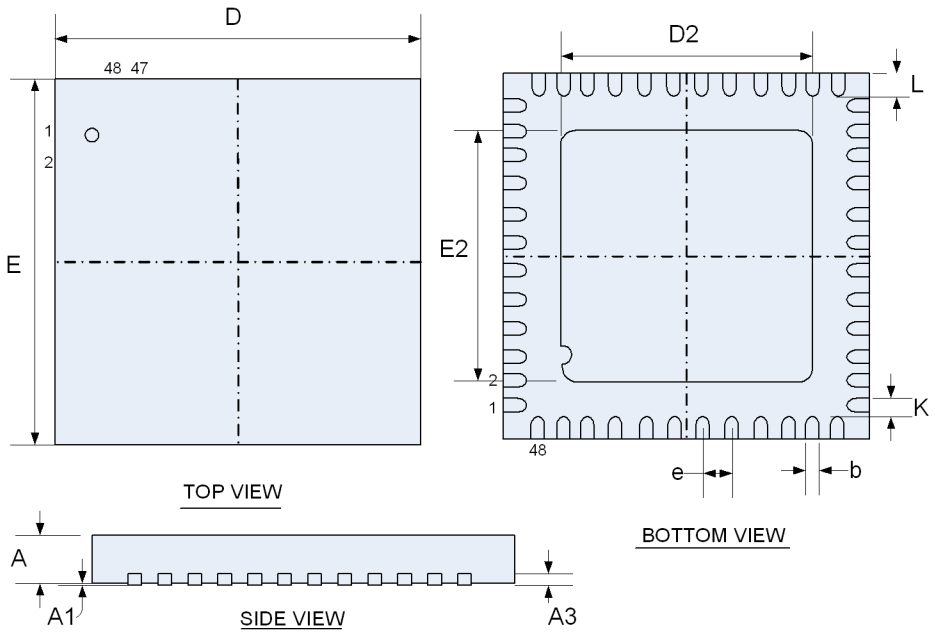



*Figure 76. QFN48 pin 7x7mm*

| Package | A | A1 | A3 | b | D, E | D2, E2 | e | K | L | |
|---|---|---|---|---|---|---|---|---|---|---|
| QFN24 | 0.80 | 0.00 | 0.203 | 0.18 | 3.9 | 2.60 | 0.5 | 0.20 | 0.35 | Min |
| | 0.85 | 0.02 | | 0.25 | 4.0 | 2.70 | | | 0.40 | Typ |
| | 0.90 | 0.05 | | 0.30 | 4.1 | 2.80 | | | 0.45 | Max |
| QFN32 | 0.80 | 0.00 | | 0.18 | 4.9 | 3.50 | | 0.20 | 0.35 | Min |
| | 0.85 | 0.02 | 0.20 | 0.25 | 5.0 | 3.60 | 0.5 | | 0.40 | Typ |
| | 0.90 | 0.05 | | 0.30 | 5.1 | 3.70 | | | 0.45 | Max |
| QFN48 | 0.80 | 0.00 | | 0.18 | 6.9 | 3.90 | | 0.20 | 0.35 | Min |
| | 0.85 | 0.02 | 0.203 | 0.25 | 7.0 | 4.00 | 0.5 | | 0.40 | Typ |
| | 0.90 | 0.05 | | 0.30 | 7.1 | 4.10 | | | 0.45 | Max |

*Table 115. QFN32/48 dimensions in mm*

# 28 Reference circuits

## 28.1 nRF24LE1 OTP, 7×7 mm QFN48

### 28.1.1 Schematics

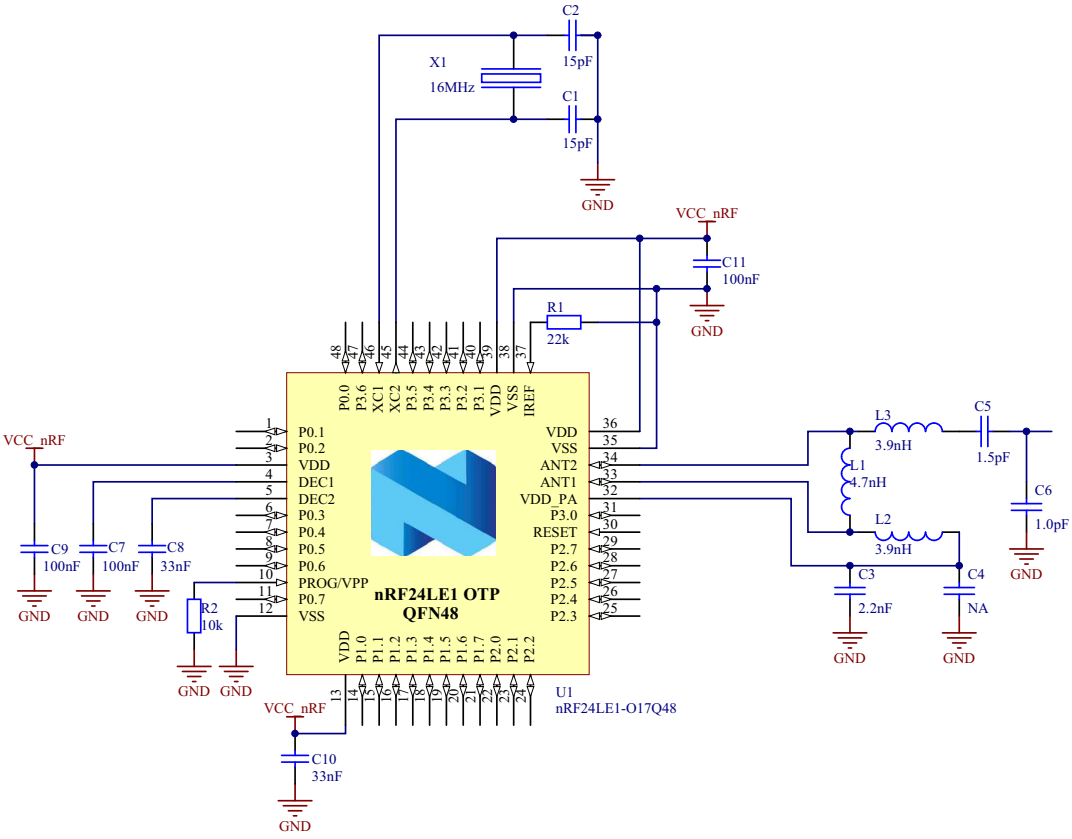
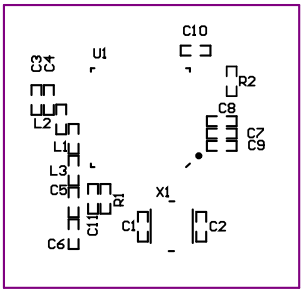


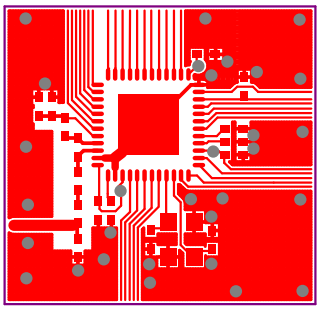

*Figure 77. nRF24LE1 OTP, 7×7 mm QFN48 schematics*

### 28.1.2 Layout

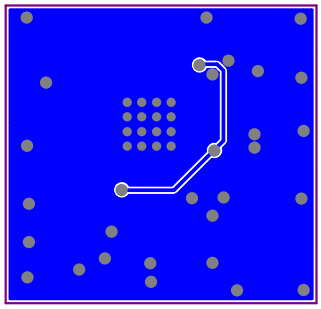



Top silk screen

No components in bottom layer



Top view



Bottom view

### 28.1.3 Bill of Materials (BOM)

| Designator | Value | Footprint | Comment |
|---|---|---|---|
| C1, C2 | 15pF | 0402 | NP0 +/- 2% |
| C3 | 2.2nF | 0402 | X7R +/- 10% |
| C4 | Not mounted | 0402 | |
| C5 | 1.5pF | 0402 | NP0 +/-0.1pF |
| C6 | 1.0pF | 0402 | NP0 +/-0.1pF |
| C7, C9, C11 | 100nF | 0402 | X7R +/- 10% |
| C8, C10 | 33nF | 0402 | X7R +/- 10% |
| L1 | 4.7nH | 0402 | High frequency chip inductor +/-5% |
| L2, L3 | 3.9nH | 0402 | High frequency chip inductor +/-5% |
| R1 | 22k | 0402 | 1% |
| R2 | 10k | 0402 | If external programming is not required, the **PROG/VPP** pin can be connected directly to ground (GND) and R2 is not needed. |
| U1 | nRF24LE1-O17Q48 | QFN48 | QFN48 7x7 mm package |
| X1 | 16 MHz | 3.2×2.5 mm | SMD-3225, 16 MHz, CL=9pF, +/-60 ppm |
| PCB substrate | FR4 laminate | 20.8×19.9 mm | 2 layer, thickness 1.6 mm |

*Table 116. nRF24LE1 OTP, 7x7 mm QFN48 Bill of Materials*

## 28.2 nRF24LE1 OTP, 5×5 mm QFN32

### 28.2.1 Schematics

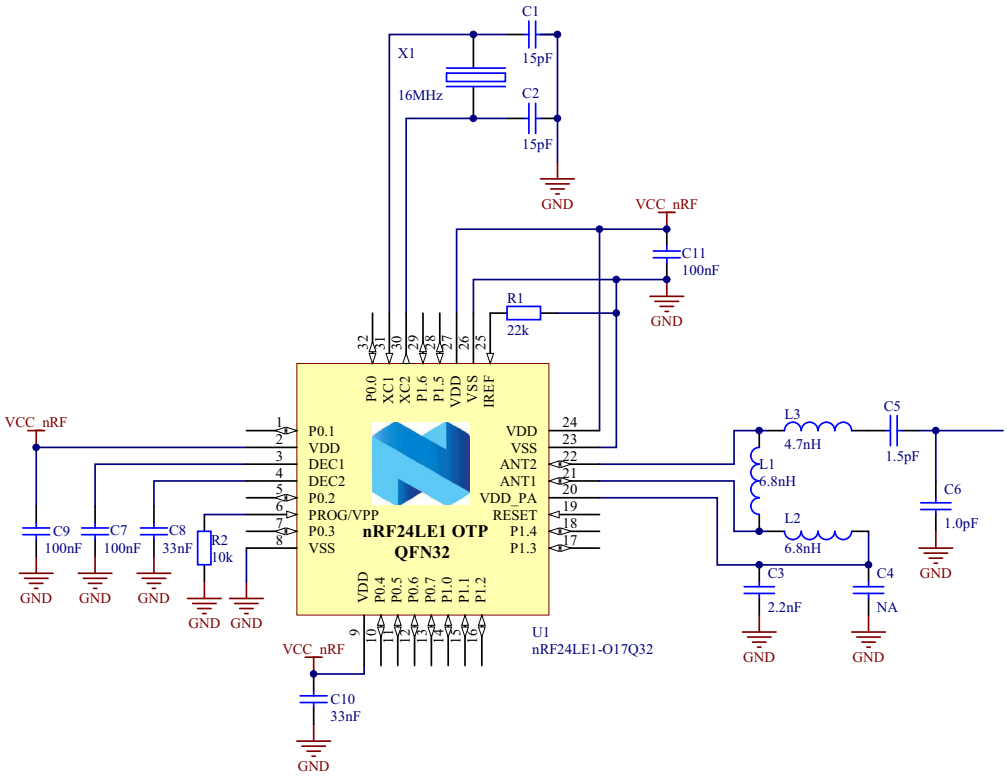
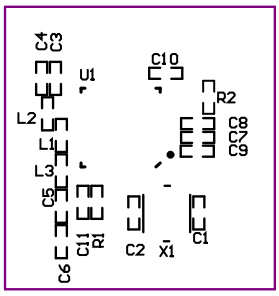


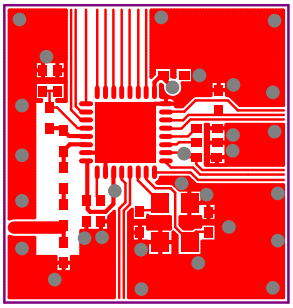

*Figure 78. nRF24LE1 OTP, 5×5 mm QFN32 schematics*

### 28.2.2 Layout

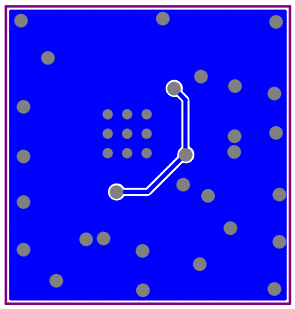



Top silk screen

No components in bottom layer



Top view



Bottom view

### 28.2.3 Bill Of Materials (BOM)

| Designator | Value | Footprint | Comment |
|---|---|---|---|
| C1, C2 | 15pF | 0402 | NP0 +/- 2% |
| C3 | 2.2nF | 0402 | X7R +/- 10% |
| C4 | Not mounted | 0402 | |
| C5 | 1.5pF | 0402 | NP0 +/-0.1pF |
| C6 | 1.0pF | 0402 | NP0 +/-0.1pF |
| C7, C9, C11 | 100nF | 0402 | X7R +/- 10% |
| C8, C10 | 33nF | 0402 | X7R +/- 10% |
| L1, L2 | 6.8nH | 0402 | High frequency chip inductor +/-5% |
| L3 | 4.7nH | 0402 | High frequency chip inductor +/-5% |
| R1 | 22k | 0402 | 1% |
| R2 | 10k | 0402 | If external programming is not required, the **PROG/ VPP** pin can be connected directly to ground (GND) and R2 is not needed. |
| U1 | nRF24LE1-O17Q32 | QFN32 | QFN32 5x5 mm package |
| X1 | 16 MHz | 3.2×2.5 mm | SMD-3225, 16 MHz, CL=9pF, +/-60ppm |
| PCB substrate | FR4 laminate | 16.9×17.8 mm | 2 layer, thickness 1.6 mm |

*Table 117. nRF24LE1 OTP, 5×5 mm QFN32 Bill of Materials*

## 28.3 nRF24LE1 OTP, 4×4 mm QFN24

### 28.3.1 Schematics

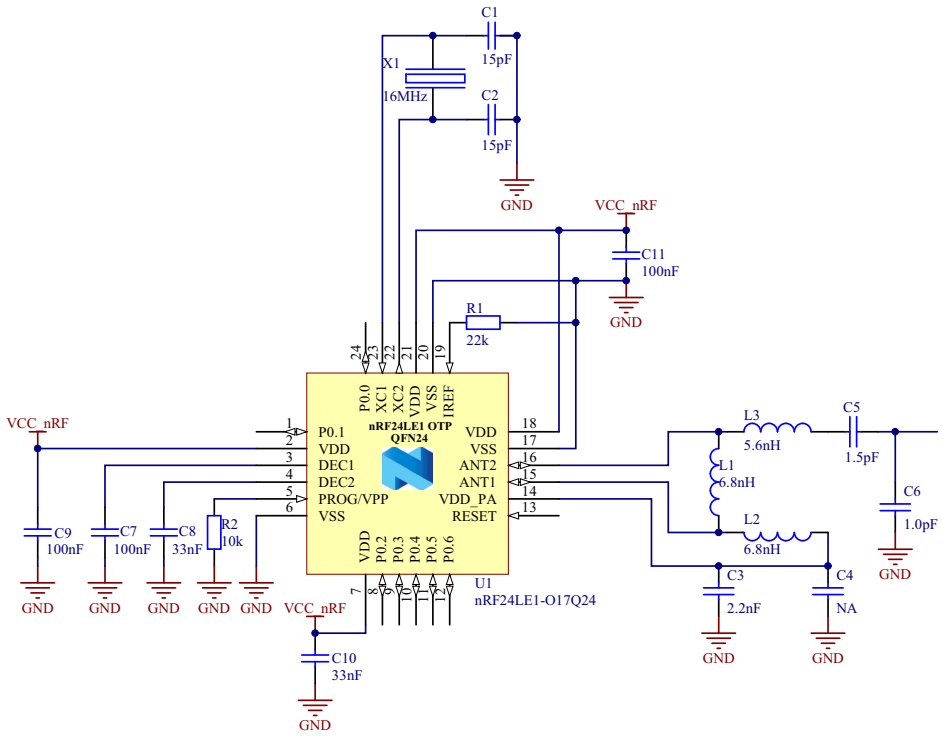
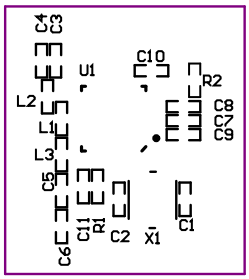


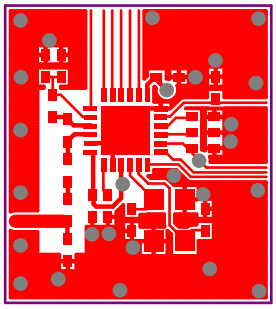

*Figure 79. nRF24LE1 OTP, 4×4 mm QFN24 schematics*

### 28.3.2 Layout

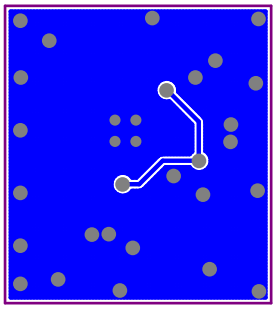



Top silk screen

No components in bottom layer



Top view



Bottom view

### 28.3.3 Bill Of Materials (BOM)

| Designator | Value | Footprint | Comment |
|---|---|---|---|
| C1, C2 | 15pF | 0402 | NP0 +/- 2% |
| C3 | 2.2nF | 0402 | X7R +/- 10% |
| C4 | Not mounted | 0402 | |
| C5 | 1.5pF | 0402 | NP0 +/-0.1pF |
| C6 | 1.0pF | 0402 | NP0 +/-0.1pF |
| C7, C9, C11 | 100nF | 0402 | X7R +/- 10% |
| C8, C10 | 33nF | 0402 | X7R +/- 10% |
| L1, L2 | 6.8nH | 0402 | High frequency chip inductor +/-5% |
| L3 | 5.6nH | 0402 | High frequency chip inductor +/-5% |
| R1 | 22k | 0402 | 1% |
| R2 | 10k | 0402 | If external programming is not required, the **PROG/VPP** pin can be connected directly to ground (GND) and R2 is not needed. |
| U1 | nRF24LE1-O17Q24 | QFN24 | QFN24 4×4 mm package |
| X1 | 16 MHz | 3.2×2.5 mm | SMD-3225, 16 MHz, CL=9pF, +/-60ppm |
| PCB substrate | FR4 laminate | 14.6×16.6 mm | 2 layer, thickness 1.6 mm |

*Table 118. nRF24LE1 OTP, 4×4 mm QFN24 Bill of Materials*

# 29 Ordering information

## 29.1 Package marking

| | | | | | |
|---|---|---|---|---|---|
| N | R | F | | A | X |
| 2 | 4 | L | E | 1 | Z |
| Y | Y | W | W | L | L |

### 29.1.1 Abbreviations

| Abbreviation | Definition |
|---|---|
| 24LE1 | Product number |
| X | "X" grade, that is, Engineering Samples (optional) |
| Z | Package type."G" = 32 pin, "H" = 48 pin and “I” = 24 pin |
| YY | Two digit Year number |
| WW | Two digit week number |
| LL | Two letter wafer lot number code |
| A | Build Code, that is, unique code for production sites, package type and test platform |

*Table 119. Abbreviations*

## 29.2 Product options

### 29.2.1 RF silicon

| Ordering code | Package | Container | MOQ |
|---|---|---|---|
| nRF24LE1-O17Q24-T | 4×4mm 24-pin QFN, lead free (green) | Tray | 490 |
| nRF24LE1-O17Q24-R7 | 4×4mm 24-pin QFN, lead free (green) | Tape-and-reel 7” | 1500 |
| nRF24LE1-O17Q24-R | 4×4mm 24-pin QFN, lead free (green) | Tape-and-reel 13” | 4000 |
| nRF24LE1-O17Q24-SAMPLE | 4×4mm 24-pin QFN, lead free (green) | Sample box | 5 |
| nRF24LE1-O17Q32-T | 5×5mm 32-pin QFN, lead free (green) | Tray | 490 |
| nRF24LE1-O17Q32-R7 | 5×5mm 32-pin QFN, lead free (green) | Tape-and-reel 7” | 1500 |
| nRF24LE1-O17Q32-R | 5×5mm 32-pin QFN, lead free (green) | Tape-and-reel 13” | 4000 |
| nRF24LE1-O17Q32-S | 5×5mm 32-pin QFN, lead free (green) | Sample box | 5 |
| nRF24LE1-O17Q48-T | 7×7mm 48-pin QFN, lead free (green) | Tray | 260 |
| nRF24LE1-O17Q48-R7 | 7×7mm 48-pin QFN, lead free (green) | Tape-and-reel 7” | 1000 |
| nRF24LE1-O17Q48-R | 7×7mm 48-pin QFN, lead free (green) | Tape-and-reel 13” | 3000 |
| nRF24LE1-O17Q48-S | 7×7mm 48-pin QFN, lead free (green) | Sample box | 5 |

*Table 120. nRF24LE1 OTP RF silicon options*

### 29.2.2 Development tools

The nRF24LE1 OTP will not have separate development kits. For development use nRF24LE1 devices with flash memory. If multiple memory write functions are not used, code written for the flash memory versions of nRF24LE1 will run on nRF24LE1 OTP without changes. Socket programming adapters for the nRF24LE1 OTP are available from Nordic Semiconductor. These are intended for use in engineering samples and pilot runs. For volume production use commercially available OTP programming tools.

| Type Number | Description |
|---|---|
| nRF6700 | nRFgo Starter Kit |
| nRF24LE1-F16Q24-DK | nRFgo Development Kit for nRF24LE1 4×4mm 24 pin QFN (requires nRFgo Starter Kit) |
| nRF24LE1-F16Q32-DK | nRFgo Development Kit for nRF24LE1 5×5mm 32 pin QFN (requires nRFgo Starter Kit) |
| nRF24LE1-F16Q48-DK | nRFgo Development Kit for nRF24LE1 7×7mm 48 pin QFN (requires nRFgo Starter Kit) |
| nRF6701 | nRFgo nRF24LE1 48-pin Flash/OTP Programming Adapter Kit (requires nRFgo Starter Kit) |
| nRF6702 | nRFgo nRF24LE1 32-pin Flash/OTP Programming Adapter Kit (requires nRFgo Starter Kit) |
| nRF6703 | nRFgo nRF24LE1 24-pin Flash/OTP Programming Adapter Kit (requires nRFgo Starter Kit) |

*Table 121. nRF24LE1 OTP development tools*

# 30 Glossary

| Term | Description |
|---|---|
| ACK | Acknowledgement |
| ADC | Analog to digital converter |
| ART | Auto Re-Transmit |
| BOR | Brown-Out Reset |
| CE | Chip Enable |
| CLK | Clock |
| CRC | Cyclic Redundancy Check |
| CSN | Chip Select NOT |
| ESB | Enhanced ShockBurst™ |
| GFSK | Gaussian Frequency Shift Keying |
| IRQ | Interrupt Request |
| ISM | Industrial-Scientific-Medical |
| LNA | Low Noise Amplifier |
| LSB | Least Significant Bit |
| LSByte | Least Significant Byte |
| Mbps | Megabit per second |
| MCU | Microcontroller |
| MISO | Master In Slave Out |
| MOQ | Minimum Order Quantity |
| MOSI | Master Out Slave In |
| MSB | Most Significant Bit |
| MSByte | Most Significant Byte |
| NV | Non-Volatile (memory) |
| PCB | Printed Circuit Board |
| PER | Packet Error Rate |
| PID | Packet Identity Bits |
| PLD | Payload |
| POF | Power Fail |
| POR | Power On Reset |
| PRX | Primary RX |
| PTX | Primary TX |
| PWR_DWN | Power Down |
| PWR_UP | Power Up |
| RCOSC16M | 16 MHz RC oscillator |
| RCOSC32K | 32 KHz RC oscillator |
| RNG | Random Number Generator |
| RX | Receive |
| RX_DR | Receive Data Ready |
| SPI | Serial Peripheral Interface |
| TX | Transmit |
| TX_DS | Transmit Data Sent |
| XOSC16M | 16 MHz crystal oscillator |
| XOSC32K | 32 KHz crystal oscillator |

*Table 122. Glossary*

## Looking for pricing, stock, or lifecycle information?

Click below to explore more details on WIN SOURCE:

- View NRF24LE1-O17Q32-T on WIN SOURCE
- Nordic Semiconductors Information

## Optimize Your Supply Chain with WIN SOURCE Solutions

- Global Sourcing Solution
- Obsolete Management
- Cost Control Management
- Shortage Management
- Alternative Solution
- Excess Inventory Management