



Markus Müller

Wireless Sensor Network Feasibility of the nRF24LE1D Microcontroller

Helsinki Metropolia University of Applied Sciences
Bachelor of Engineering
Degree Programme in Information Technology
Bachelor's Thesis
5th May 2011

Abstract

Author	Markus Müller
Title	Wireless sensor network feasibility of the nRF24LE1D microcontroller
Number of Pages	49 pages + 4 appendices
Date	5 May 2011
Degree Programme	Information Technology
Degree	Bachelor of Engineering
Instructor	Anssi Ikonen
<p>In this thesis the wireless sensor network feasibility of the nRF24LE1 microcontroller will be analysed. Due to the fast evolving field of embedded wireless applications, this project was initiated by the Helsinki Metropolia University of Applied Science in order to provide a platform for future projects.</p> <p>The main part of this thesis will discuss the the nRFgo software development kit and the hardware of the nRF24LE1 microcontroller. Tools that have been used are an integrated development environment provided by the Seinäjoki University of Applied Sciences and a C language compiler named SDCC.</p> <p>As a result, this study has deepened the understanding of the hardware and the nRFgo software development kit and their capabilities. The microcontroller has some excellent characteristics that makes it an interesting object in order to build wireless sensor networks. The system on a chip architecture makes it a meaningful tool for sensing and communication.</p> <p>This thesis gives an introduction into wireless sensor networks and the properties of the nRF24LE1 microcontroller with a build in radio frequency unit, demonstrating the capabilities of this promising system on a chip microcontroller and enabling further research in the field of wireless embedded networking and wireless sensor networks for the Helsinki Metropolia University of Applied Sciences</p>	
Keywords	WSN, sensor node, embedded networking, wireless networks

Table of Contents

Abstract.....	2
I List of Figures.....	5
II Index of Tables.....	6
III List of Abbreviations.....	7
1 Introduction.....	9
1.1 Tasks and Objectives.....	9
1.2 Structure of Thesis.....	9
2 Theoretical Background.....	11
2.1 Wireless Sensor Networks.....	11
2.1.1 Characteristics of a Wireless Sensor Network.....	12
2.1.2 Sensor Node.....	12
2.1.3 Sink Controller.....	13
2.1.4 Application Areas.....	14
2.2 Wireless Ad Hoc Networks.....	16
2.2.1 Characteristics of Wireless Sensor Networks.....	16
2.2.2 Network Topologies.....	16
2.2.3 Wireless Sensor Network Routing Protocols.....	19
3 Methods and Materials.....	20
4 The nRF24LE1 Microcontroller.....	22
4.1 nRF24LE1.....	22
4.2 nRF24LE1D Hardware.....	23
4.2.1 Operating conditions.....	23
4.2.2 Bus Interfaces.....	24
4.2.3 Power Management.....	28
4.2.4 Analog to Digital Converter.....	29
4.3 nRF24L01+.....	32
4.3.1 Radio Characteristics.....	32
4.3.2 RF Transceiver Properties.....	33
4.3.3 Enhanced ShockBurst.....	34
4.3.4 MultiCeiver™ Characteristics.....	36
4.4 The nRFgo SDK.....	38
4.4.1 Hardware Abstraction Layer.....	38
4.4.2 Serial Peripheral Interface (hal_spi).....	39
4.4.3 UART Interface (hal_uart).....	40
4.4.4 2-Wire (hal_w2).....	40
4.4.5 Power-fail Comparator (hal_pof).....	41
4.4.6 Analog to Digital Converter (hal_adc).....	42
4.4.7 Gazell Link Layer.....	43

5	Discussion.....	46
6	Conclusion.....	47
	References.....	49
	Appendices.....	50
	Appendix 1: Radio control state diagram.....	50
	Appendix 2: PTX operations in Enhanced ShockBurst.....	51
	Appendix 3: PRX operations in Enhanced ShockBurst.....	52
	Appendix 4: nRFgo SDK.....	53

I List of Figures

Figure 1: Wireless sensor network.....	11
Figure 2: Architecture of a node with microcontroller.....	13
Figure 3: Architecture of a sink with microcontroller.....	14
Figure 4: Single-hop and multi-hop network structure in WSN.....	18
Figure 5: SURFprogrammer user interface.....	20
Figure 6: Block diagram of test setup.....	21
Figure 7: nRF24LE1D pin assignment [6, 14].....	23
Figure 8: SPI functionality.....	25
Figure 9: UART frame structure.....	26
Figure 10: I2C frame structure.....	27
Figure 11: Power supply supervisor [6, 117].....	28
Figure 12: Block diagram of ADC [6, 165].....	29
Figure 13: ADC data register.....	31
Figure 14: 2.4 GHz Nordic nRF24LE1D channel scheme [7, 3].....	32
Figure 15: An Enhanced ShockBurst packet with variable payload.....	35
Figure 16: MultiCeiverTM data pipe model [5, p 37].....	36
Figure 17: Gazell protocol states for a Host	44
Figure 18: Gazell protocol states for a device.....	45
Figure 19: Radio control state diagram [6, 18].....	50
Figure 20: PTX operations in Enhanced ShockBurst [6, 30].....	51
Figure 21: PRX operations in Enhanced ShockBurst [6, 32].....	52

II Index of Tables

Table 1: Network topologies of WSN.....	17
Table 2: nRF24LE1 pin functions [6, 15].....	22
Table 3: SPI slave pin assignment	24
Table 4: SPI master pin assignment.....	25
Table 5: UART pin assignment.....	26
Table 6: 2-Wire slave pin assignment.....	27
Table 7: ADC output data coding.....	31
Table 8: Content of the enclosed CD.....	53

III List of Abbreviations

ACK	Acknowledgment
ADC	Analog to Digital Converter
AES	Advanced Encryption Standard
ALU	Arithmetic Logic Unit
BER	Bit Error Rate
BOR	Brown Out Reset
BS	Base Station
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CS	Chip Select
DECT	Digital Enhanced Cordless Telecommunications
ED	Event Detection
FIFO	First in First out
FSK	Frequency Shift Keying
GFSK	Gaussian Frequency Shift Keying
GZLL	Gazell Link Layer
HAL	Hardware Abstraction Layer
LSB	Least Significant Bit
IC	Integrated Circuit
IO	Input and Output
ISM	Industrial, Scientific and Medical
MCU	Microcontroller
MSB	Most Significant Bit
PC	Personal Computer
PID	Packet Identification
POF	Power Fail
POR	Power on Reset
PRX	Receiver
PTX	Transmitter
RF	Radio Frequency
RNG	Random Number Generator
SDK	Software Development Kit
SFR	Special Function Register
SPI	Serial Peripheral Interface
SPS	Samples per Second
SOC	System on a Chip
UART	Universal Asynchronous Receiver Transmitter
ULP	Ultra Low Power

USB	Universal Serial Bus
VDD	Positive Supply Voltage (of a Field Effect Transistor Device)
VSS	Negative Supply Voltage (of a Field Effect Transistor Device)
WSN	Wireless Sensor Network

1 Introduction

This bachelor thesis discusses the analysis and the prospects of a wireless sensor network with the nRF24LE1 microcontroller manufactured by Nordic Semiconductor. This study was done to provide an open environment for future use and development at the Helsinki Metropolia University of Applied Sciences. Furthermore, the analysis of the nRF software development kit and its capabilities for a wireless sensor network is part of this thesis. This project has its focus on the measurement system project where the knowledge about embedded systems, network architecture and interfaces is combined.

1.1 Tasks and Objectives

The goal of this bachelor thesis is to analyse the nRF24LE1 microcontroller (MCU) in its properties and specification, to test if the hard and software is qualified for a wireless sensor network and to implement this in a prototype. In the first step the microcontroller will be discussed in detail to find out the possibilities it provides. Subsequently the nRF software development kit (SDK) and its characteristics will be reviewed. Finally, the prospect will be investigated to build a wireless sensor network with the microcontroller nRF24Le1. The main task will be to demonstrate the performance of the nRF24Le1 microcontroller with an integrated radio unit.

1.2 Structure of Thesis

Chapter 2 will discuss wireless sensor networks on a general level, but it will also illustrate specific aspects which are significant for the understanding of details of the related to hardware and software, also discussed later. Chapter 2 will also shed light on some aspects about wireless ad hoc networks and how this technology is used in wireless sensor network (WSN). Chapter 4 will give an overview about the nRF24LE1 microcontroller from Nordic Semiconductor. The chapter will include details about hardware and also about the software. Especially, a software development kit called nRF SDK will be discussed in detail in order to understand the capabilities of the microcontroller.

Subsequently, the aspects of the wireless sensor network and capabilities of the nRF24LE1 microcontroller will be discussed and the functionality of this device will be shown. Chapter 5 will summarize the benefits but also demonstrate some drawbacks. Other possible applications for further projects will also be reviewed.

2 Theoretical Background

In the last recent decades, demand has increased for monitoring the environment. The aim of a WSN is to identify events and changes over time and to respond to these as soon as possible. The desire for wireless transmission of data is based on the fact that a wired network costs more and is more difficult to install, especially in rural regions.

2.1 Wireless Sensor Networks

A wireless sensor network (WSN) is a network of devices denoted as nodes which detects an analog physical value in an environment and transmits the measured data through wireless links to a central device also denoted as a sink. Figure 1 show a typical WSN setup with multiple nodes and a sink. Wireless sensor networks are the key to new connections between humans and the environment because there are thousands of potential applications for these networks. However, there are limitations in the underlying hardware and software as well. The challenge is to balance the limitations and the requirements of the system.

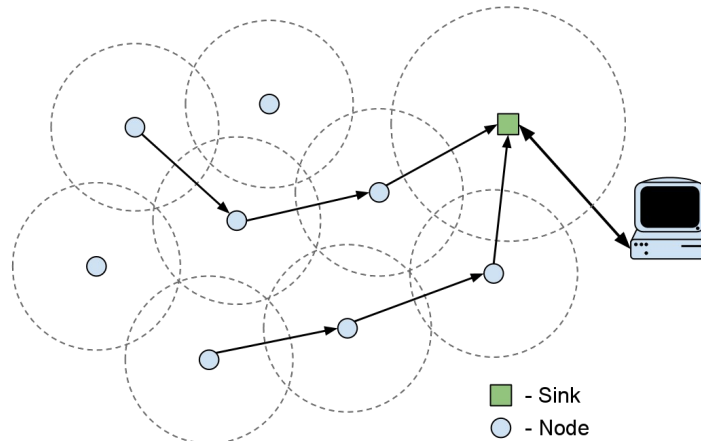


Figure 1: Wireless sensor network

The main features of a WSN are scalability, self-organization, low-complexity, self-healing, energy efficiency, low cost and size of nodes. All these features are designed from the lowest level of the hardware to the highest level of the user interface.

2.1.1 Characteristics of a Wireless Sensor Network

The key characteristics of a WSN are low cost devices which are very energy efficient. The connection between the devices can be wired; however, the connection is usually wireless and makes use of a self organising and self configuring routing protocol which allows each node to be rapidly repositioned. The availability and high integrity are part of the features of a WSN. The security aspect also plays a critical role in the concept of wireless sensor networking.

2.1.2 Sensor Node

A sensor node combines three main functionalities: sensing, calculations and communication. In order to provide this characteristic the node needs special parts for each of these functionalities. Often the calculations and computing are done by a microcontroller which also connects every part and represents the central core. For example, a microcontroller has special ports for sensing of values. The sensing must guarantee a minimum density which ensures the process or an event is precisely estimated.

These values are usually analog values, converted by analog to digital converter (ADC), which are integrated in most modern microcontrollers. Furthermore, part of the node is a radio-frequency (RF) element which handles the communication on a physical layer. The RF component is integrated in a modern microcontroller, such as the ADC. These microcontrollers which integrate all components such as memory, an arithmetic logical unit (ALU), an ADC, or an RF unit are called system on a chip (SOC). A typical node architecture is illustrated in figure 2 where all required components are placed.

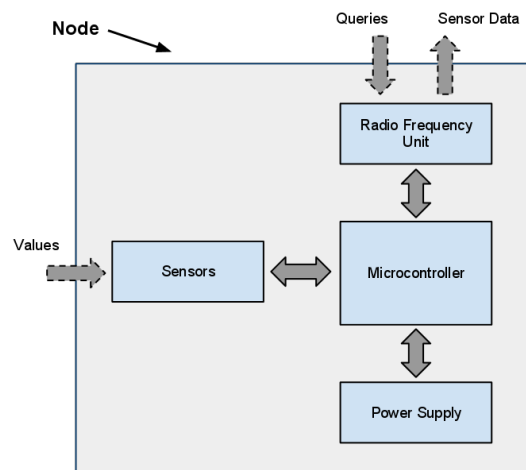


Figure 2: Architecture of a node with microcontroller

A system on a chip architecture qualifies to be the central part of a node because it combines everything that is needed for a wireless sensor node in a sensor network. A SOC microcontroller drops the manufacturing costs for a wireless node, so that a complete wireless sensor network becomes more attractive for many applications. Another benefit of the modern SOC systems is the low power consumption that makes it possible to operate these nodes with batteries over years.

2.1.3 Sink Controller

A sink controller in a wireless sensor network represents the collecting device which means that every node is sending its data through the network to the sink. The sink analyses, stores and monitors the data. The sink is usually connected to a storage system, a monitoring system or a personal computer (PC).

The microcontroller writes the collected data into a storage system in order to let the user of the system evaluate this information on a later date. This is especially interesting for applications in rough terrains where it is not possible to check the information every day and when the object of interest needs to be monitored over a long period. Another way to present the information is to display an event immediately on a monitor. A PC combines the discussed functions. In this case the interface between the sink and the PC can be a network connection, a link over a serial port or any other kind of data connection to a computer.

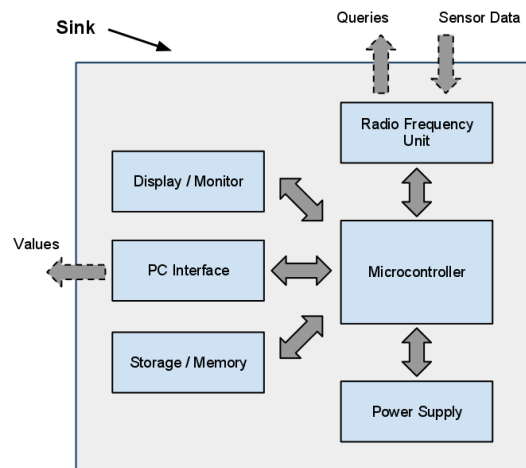


Figure 3: Architecture of a sink with microcontroller

Figure 3 shows a block diagram of a sink architecture and the relation of the contained components. The most important characteristics of the sink are the connection to a display, a storage system or a personal computer. Normally, a modern microcontroller provides interfaces such as the universal serial bus (USB), universal synchronous receiver transmitter (UART), Ethernet or equivalent which can be used for data exchange between these two devices.

A sink does not represent a central device which is responsible for the configuration in the network as there are for example routing and position changes. A sink presents an interface to other networks, a PC, storage systems or displays and it handles the sensor data which are measured.

2.1.4 Application Areas

Nowadays there are innumerable applications and situations where sensor networks can be found, even in everyday life. One reason for the big success of WSN in the past few years is the diversity of this wireless sensor networks. “The concept of wireless sensor networks is based on a simple equation:

$$\text{Sensing} + \text{CPU} + \text{Radio} = \text{Thousands of potential applications}$$

As soon as people understand the capabilities of a wireless sensor network, hundreds of applications spring to mind. It seems like a straightforward combination of modern technology.” [2, 20]. The major application for WSN infrastructures are the following:

- Environmental monitoring
- Health care
- Positioning and animals tracking
- Entertainment (home and office)
- Logistics and transportation
- Military use

All these completely different applications have two basic duties: event detection (ED) and spatial time random process estimation (PE) [3, 32].

Event Detection

Event detection sensor networks compare the measured values of the actual situation to a given threshold which can be a minimum or maximum value or both. ED sensor networks focus on quick detection of the occurring events. The coverage of the monitored area needs to have a high probability which means that the WSN communication protocol has to provide a low bit error rate (BER) and a solid bit error detection algorithm. In an event based sensor network only the event itself will be announced. This saves a lot of energy because the nodes are not frequently transmitting.

Time Random Process Estimation

The process estimation (PE) system records measured values over a period of time with a certain incidence. In addition, the sampling frequency has to be considered. It has to be high enough in a way that the process is tracked in an adequate detail. Because the measured data depends on time, time synchronization between the nodes and the sink plays an essential role.

Data collecting requires a signal processing algorithm that brings the data in the needed exposition before transmission which can reduce the amount of exchanged data. This also means saving energy because every packet that will be transmitted or received consumes resources. In the random process estimation system the accuracy of the values is significant.

2.2 Wireless Ad Hoc Networks

A wireless ad hoc network is a decentralized wireless network which consists of at least two nodes connected over a peer to peer connection also called a link. One property of an ad hoc network is that no existing base station (BS) such as a router or server is required for the configuration. This means the network has to be self-organising, which includes handling of addresses and routes in the network. The nodes establish and organise the links, routes and other configurations which may be necessary. In a WSN all this is required since the sink is usually not a central configuration device. The purpose of a sink is to collect the information which is floating in the network.

2.2.1 Characteristics of Wireless Sensor Networks


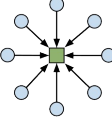
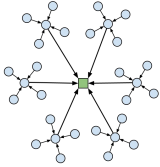
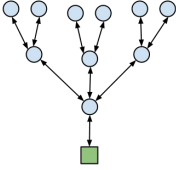
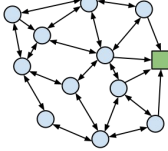
In ad hoc networks data is transmitted by establishing a peer to peer connection. After the transmission of every single packet, the link is released. This is especially important in wireless communication, because a channel can be used from different members of the network which are not involved in the conversation. Another way to propagate information in a wireless ad hoc network is to flood the network with the data instead of opening and closing a special route. Flooding the wireless ad hoc network with a data packet is also denoted as broadcasting.

These attributes of wireless ad hoc networks and the requirements of a sensor networks show that a wireless ad hoc network is the best solution for the purpose of a WSN.

2.2.2 Network Topologies

Different types of network topologies are possible to deploy. The simplest configuration is a **peer-to-peer** connection between a node and a sink which is a direct link between these devices. Another network topology is the star structure, which consists of one sink in the middle surrounded by nodes. The **star topology** is made out of multiple peer-to-peer connections. It is only suitable for small networks with fewer nodes, because it is, such as the peer-to-peer topology, a single-hop structure. The star topology is used to build an **extended star**, illustrated in table 1 Nodes are using another node as the next hop to reach a central device. The next hop nodes are grouped as a star around the sink.

Table 1: Network topologies of WSN

Topologies	Denomination
	Peer-to-Peer
	Star
	Extended Star
	Hierarchical
	Mesh

As table 1 also illustrates, a **hierarchical** network topology is close to the star topology; however, there are some crucial differences. For example, the hierarchical networks implements layers, where each layer represents a collection of devices with a different purpose. It is also possible to think about varying priorities for each layer. The picture of a tree illustrates the hierarchical network structure in a good way. Another possible configuration is a so called **mesh** network where every node is connected with its neighbour. This is also the most flexible and most dynamic network topology with the highest requirements concerning self-organisation of the network. This is because there are many more possible routes compared to the other structures. [1]

There are many more different types of network topologies such as the bus and ring topology, but the topologies listed above are the most important in the wireless sensor network field. These topologies can also be categorized into single-hop and multi-hop topologies as can be seen in figure 4.

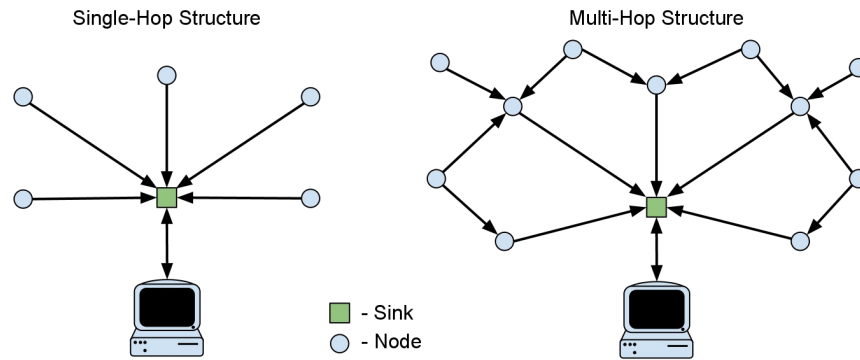


Figure 4: Single-hop and multi-hop network structure in WSN

Single-Hop

In a single-hop system, shown in figure 4, every delivered packet is transmitted at once. This means the transmitter, in case of a WSN a node, can reach the receiver or sink directly without any additional device in between this transmission line. This can be found in a peer-to-peer or in a star topology. In case of a single-hop system there is no need for dynamic routing because the network structure is elementary and has low complexity. This type of arrangement is only suitable for small sensor networks, because a single-hop structure does not provide high scalability. [2]

Multi-Hop

Multi-hop routing of the data in the network means that nodes can also act as a bridge between the source and the destination of a packet. In a situation where the sender is not able to reach the destination with a single-hop, a node which can reach both the transmitter and the receiver forwards the packages for the remote nodes. Multi-hop routing is essential in the extended-star, hierarchical and mesh networks and allows these networks a high scalability. This means that multi-hop systems are able to handle networks with up to thousands of sensor nodes. This is due to the self-management and dynamic routing which are part of a multi-hopping network protocol.

However, this dynamic multi-hop routing concept causes much more data traffic in the network compared to single-hop systems. Every change in the network configuration must be propagated in the network. This means that the nodes are more often busy transmitting configuration information, which leads to higher energy consumption of the devices.[2, 9]

2.2.3 Wireless Sensor Network Routing Protocols

Ad hoc network technologies are used to form the basis of a wireless sensor network. Wireless sensor networks can consist of thousands of nodes where the configuration and the routing are self-organized by the routing protocol. The wireless sensor network routing protocol has two main responsibilities which are self-organisation and data collection. A wireless ad hoc network protocol has to handle special situations and the needs of a wireless sensor network which could be caused by continuously changing network topologies. For example, the routing table has to be updated frequently. Each node learns about nodes nearby and how to reach them. In addition, this includes the self-announcements if a node enters a new network.

The collection of the data from the sensors and the transmission to the sink is also handled by the WSN protocol which includes the selection of the best way through the network or finding an alternative path if one node has an operational failure. It is also common that the occurrence of data acquisition is controlled by the protocol.

In order to meet the requirements, such as fast routing, scalability and energy efficiency of every single WSN application, the network protocol needs to be adjusted or reimplemented. As a result, there are numerous implementations of ad hoc network protocols.

3 Methods and Materials

For this project and its practical implementation a nRF24LE1D microcontroller was used. This microcontroller has been developed and manufactured by the company Nordic Semiconductor. Nordic Semiconductor is specialized on small SOC microcontrollers and ultra low power wireless communication solutions. This company offers devices, with support for Bluetooth, ANT and the Gazell protocol. The nRF24LE1D MCU includes the Gazell link layer protocol, which is discussed in section 4.4.7.

Any text editor can be used as a development environment in collaboration with a version of the Small Device C Compiler (SDCC) in order to write and compile software for the nRF24LE1D microcontroller. In this project a software called SURFprogrammer is used to accomplish these tasks. The SURFprogrammer is an open source integrated development environment and has been developed by the Seinäjoki University of Applied Sciences. The user interface of the SURFprogrammer is shown in figure 5.

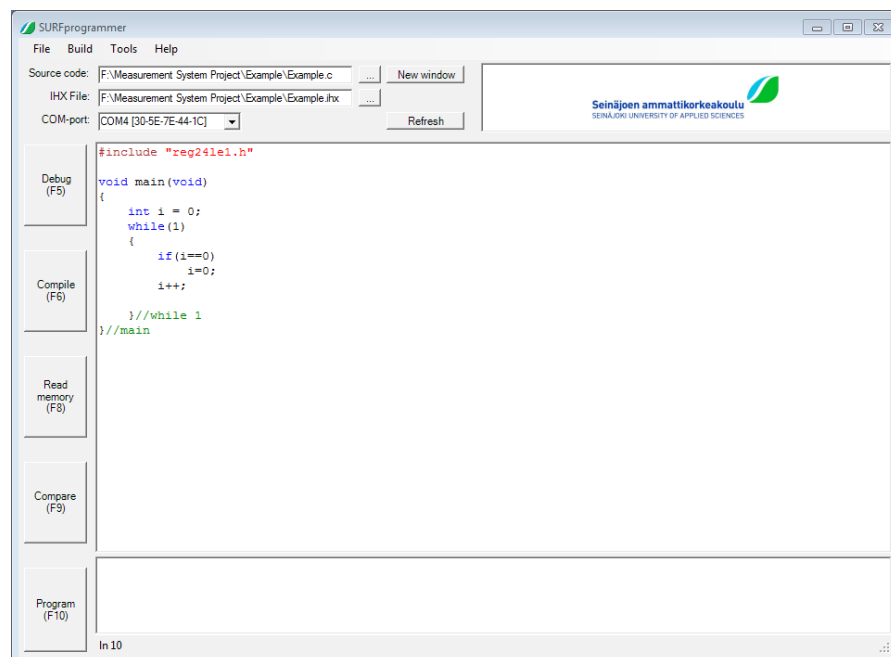


Figure 5: SURFprogrammer user interface

The SURFprogrammer software can be seen in the figure 5 above. It can connect to the programmer, compile c source files and send the compiled source code to the nRF24LE1D microcontroller. The SURFprogrammer uses the SDCC in order to generate hexadecimal files. For this study the version 1.1.9 of the SURFporgrammer was used and version 3.0.0 of the SDCC.

The SURFprogrammer connects through a programmer hardware to the nRF24LE1D MCU in order to send the source code to the microcontroller. The programmer hardware consists of a AT90USB162 microcontroller and runs a custom firmware. The firmware and the hardware layout of all parts can be found on the project homepage of the Seinäjoki University of Applied Sciences (<http://lompsa.seamk.fi/sulautetut/>). In figure 6 the structure of the test setup is illustrated.

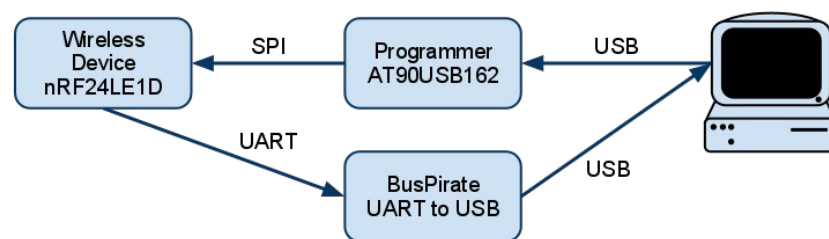


Figure 6: Block diagram of test setup

Figure 6 illustrates the principle setup of the test environment for this project. The computer on the right is connected to the programmer through a universal serial bus (USB). The programmer is connected over a serial peripheral interface with the nRF24LE1D microcontroller and it programmes the nRF24LE1D. During this the nRF24LE1D performs a program, and the microcontroller sends status information over the UART interface out to the BusPirate. The Bus Pirate is a UART to USB converter, which was necessary choice, because the available computer for this project has no UART interface.

4 The nRF24LE1 Microcontroller

The nRF24LE1 is a system on chip for the 2.4 GHz industrial, scientific and medical (ISM) band. This chip is suitable in a wide range of applications like PC peripherals, gaming, sports and fitness, toys, healthcare, automation or other consumer or industrial electronics [4]. This controller already offers a set of functionality which can also be used in WSNs.

4.1 nRF24LE1

The nRF24LE1 microcontroller integrates an RF unit called nRF24L01. The nRF24L01 is an ultra low power (ULP) RF transceiver integrated circuit. The nRF24LE1 is a 16Mhz processor with an 8-bit CPU which is compatible to the 8051 architecture. The nRF24LE1 also includes other components such as a 12-bit ADC, 16kB Flash storage, advanced encryption standard (AES) unit, a random number generator, an UART interface and a many other hardware features which will be discussed in this chapter. There are three different implementations of the nRF24LE1 which differ in the number of in and output (IO) pins. The implementations are listed here:

- nRF24LE1D - ultra compact 4×4mm 24 pin QFN (7 generic IO pins)
- nRF24LE1E - compact 5×5mm 32 pin QFN (15 generic IO pins)
- nRF24LE1F - 7×7mm 48 pin QFN (31 generic IO pins)

Further, the basic pins of all nRF24LE1s can be found in table 2.

Table 2: nRF24LE1 pin functions [6, 15]

Name	Type	Description
VDD	Power	Power Supply (+1.9V to 3.6V DC)
VSS	Power	Ground (0V)
PROG	Digital Input	Input to enable flash programming
RESET	Digital Input	Reset the microcontroller , active low
IREF	Analog Input	Device reference current output.
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 16MHz crystal

In this thesis the nRF24LE1D will be discussed in detail, because the microcontrollers are all using the same architecture and they only vary in the number of available IO pins. Every code or application can be assigned to the nRF24LE1E and nRF24LE1F.

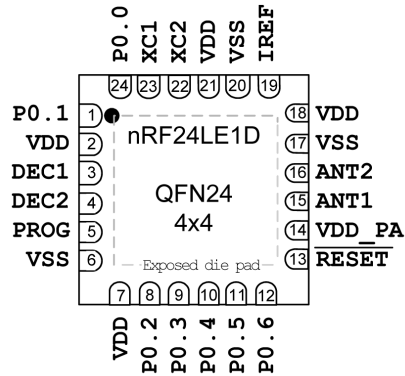


Figure 7: nRF24LE1D pin assignment [6, 14]

Figure 7 is the top view on the nRF24LE1D microcontroller and shows all pins and a related assignment. Every pin shown in this illustration is also available in the nRF24LE1E and nRF24LE1F microcontroller.

4.2 nRF24LE1D Hardware

In this chapter the specific capabilities of the MCU will be discussed in more detail; however, not all features of the nRF24LE1D MCU will be discussed in this thesis. This is because the focus lies on the hardware features which support the establishment build a WSNs.

4.2.1 Operating conditions

The nRF24LE1 microcontroller device class needs a minimum voltage of $1.9V$ up to a maximum of $3.6V$ for robust operation. The typical operation supply voltage is $3V$. The operating temperature is in a range of -40 and $+85$ °C, which makes the MCU suitable for a wide range of applications even in areas with strong temperature fluctuations like deserts.

The power consumption differs depending on the operation which is executed. That kind of operation can be an RF transmission or the reading of analog values with the analog to digital converter. In active RX or TX mode the power consumption is typically near to $13mA$. When reading analog values the consumption is approximately up to $1.3mA$. However, this MCU also provides a deep sleep mode with a power consumption average of $0.5\mu A$ which offers a long life time for battery applications.

4.2.2 Bus Interfaces

The nRF24LE1D provides a set of interfaces for the exchange of data between devices, such as a PC or other microcontrollers. The capability for interconnection with other devices is necessary to share the measured data with other networks, to store the data or to display them on a monitoring system.

Three different common types of interfaces are supported by the nRF24LE1D, which are serial peripheral interface (SPI), 2-Wire and UART. Since the nRF24LE1D is limited in the amount of available IO, it can only offer one connection type at the same time.

Serial Peripheral Interface Bus

The most common interface bus for inter microcontroller communication is the serial peripheral interface bus. It is a synchronous serial data connection with a master to slave relation. The nRF24LE1D can be operated as a master or slave, depending on the configuration.

Table 3: SPI slave pin assignment

Pin	Type	Description	Direction
P0.2	SCK/CLK	Serial Clock	Input
P0.3	MOSI	Slave Data Input	Input
P0.4	MISO	Slave Data Output	Output
P0.5	CS/SS	Chip Select / Slave Select	Input

Table 3 describes the pin assignment for the SPI bus if the MCU is configured as slave, while Table 4 describes the pin assignment of the SPI during a master mode.

Table 4: SPI master pin assignment

Pin	Type	Description	Direction
P0.2	SCK/CLK	Serial Clock	Output
P0.3	MOSI	Master Data Output	Output
P0.4	MISO	Master Data Input	Input

During the SPI master mode there is no chip select (CS) signal out, which is due to the signal being produced by the programmer and being assigned to any available pin which is not in use. This means a master can transfer data to different slave devices over varying chip select signals, each for every slave.

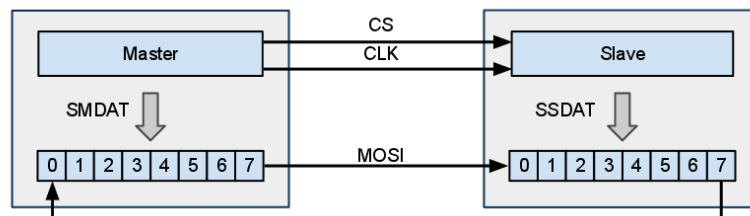


Figure 8: SPI functionality

The transmitted data is stored in the SMDAT (SPI Master DATA Register) or SSDAT (SPI Slave DATA Register) and can be read from there depending on the operation mode. The data register operates as a first in first out (FIFO) shift register which forms an inter-ship circular buffer. This shift register transmission process is illustrated in figure 8.

UART

Universal asynchronous receiver/transmitter is similar to the standard 8051 serial port and gets the clock signal from the internal microcontroller clock. The nRF24LE1D offers four different operation modes:

- 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

The S0CON register is the serial port configuration register, where the serial mode and clock source are configured. In order to transmit data, a write or read command to the S0BUF will be executed. If data is written to the S0BUF, the information is written to the serial output buffer and the transmission is initiated. A read command will receive the data from the serial receive buffer.

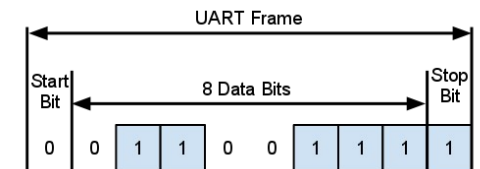


Figure 9: UART frame structure

Figure 9 shows a standard UART frame that illustrates the transmission process. The transmission is initiated with a start bit (0 or low), followed by eight data bits. The end of the transmission is indicated by a stop bit represented with a high value. The serial port baud rate can be configured using the S0RELL and S0RELH registers. Both registers are 8-bit registers, but only 10 of the 16 available bits are used, which means only the two least significant bits (LSB) are used in the S0RELH register.

Table 5: UART pin assignment

Pin	Type	Description	Direction
P0.5	TXD	UART Transmission Data	Output
P0.6	RXD	UART Receiving Data	Input

The pin assignment of the UART interface is described in table 5. Because the UART interface only has two wires and the UART frame (cf. figure 9) has a simple architecture compared to 2-Wire frame (cf. figure 10), it is easy to use for simple data exchanges.

2-Wire

The MCU has a single buffered 2-Wire interface which can be configured in order to transmit or receive data. The nRF24LE1D can act as a master or slave device, with two various data rates which are $100kHz$ and $400kHz$. The 2-Wire interface is also called I²C interface. As already mentioned, the 2-Wire interface has two pins for signalling. A pin assignment overview for the 2-Wire interface is shown in table 6.

Table 6: 2-Wire slave pin assignment

Pin	Type	Description	Direction
P0.5	SCL	2-Wire Clock Signal	Input
P0.6	SDA	2-Wire Data Signal	Input

The nRF24LE1D supports four different 2-Wire operational modes.

- Master transmitter
- Master receiver
- Slave transmitter
- Slave receiver

If the MCU is in the 2-Wire slave mode and the start condition is detected at the two input pins, the MCU enters the RX mode. The address byte (W2DAT[7] down to W2DAT[1]) of the received packet will be analysed and compared to the MCUs 2-Wire address which is stored in the W2SADR register. The received address has to match the proper address or the broadcast address 0x00. W2DAT[0] is the R/W bit and forces the slave to stay in a receive mode if it is zero or to switch to a transmission mode if W2DAT[0] is equal to one.

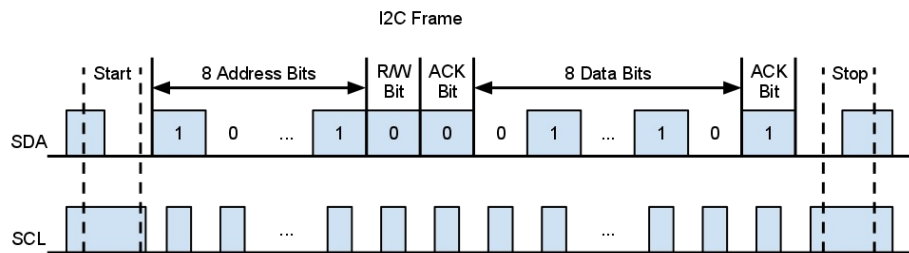


Figure 10: I2C frame structure

If the 2-Wire slave is too slow in processing the incoming data, it can force the master to decrease the rate instead. The 2-Wire frame structure illustrated in figure 10 is more complex due to its ability of addressing compared to the UART frame. This is also reflected in the complexity of the related libraries (cf. section 4.4).

4.2.3 Power Management

The MCU nRF24LE1D comes with an incorporated power supply supervisor which is an early warning system for erroneous function. It is initialized immediately after the MCU has applied the operational power on the voltage supply pin. The power management can be configured by setting the bits in the special function register (SFR) POFCON which has the address 0xDC.

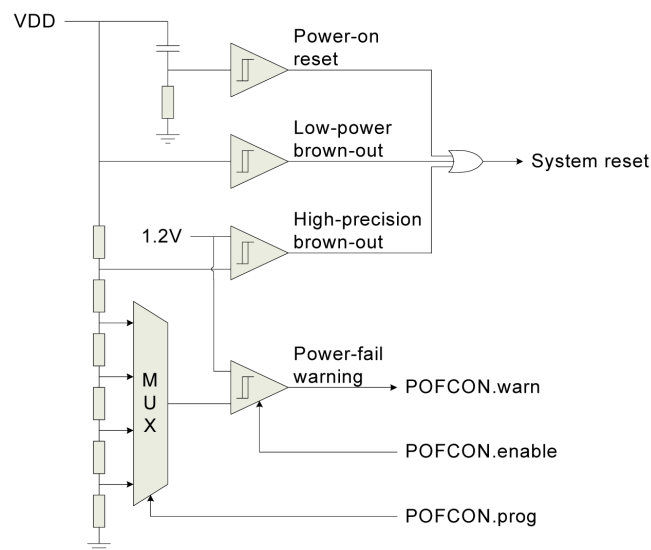


Figure 11: Power supply supervisor [6, 117]

The hardware implementation of the power supply supervisor is illustrated in figure 11 and provides an overview about the functioning. On an elementary level the power management compares two voltage levels and stores the result of this operation in a register.

Power-on Reset

After power on the MCU, the device will recognise if the supply voltage reaches a value of $1.9V$. After the required supply voltage is reached, the power on reset (POR) generator puts the system into a reset state for a minimum time period of $1ms$ before it will switch into an active mode which is the default.

Brown-out Reset

Another power monitoring element is the brown out reset (BOR) generator, which forces the MCU to go into the reset mode if the power supply voltage level falls under the BOR threshold. The BOR threshold is around $1.7V$. As already noted, the devices will go in the reset state. In order to bring the MCU back to active or standby state, the supply voltage has to reach at least $1.77V$ again. This means the BOR generator has a hysteresis with a difference of approximately $70mV$. The reason why there is a hysteresis is to prevent the input from oscillating between reset and active mode when the supply voltage comes close to the BOR threshold.

Power-fail Comparator

The MCU which is discussed in the thesis has an early power-fail warning system which means the MCU receives information instantly if the supplied voltage falls under a defined threshold. This power fail (POF) threshold can be set in the POFCON register which has the memory address $0xDC$. The nRF24LE1D can be configured with one of the four different thresholds which are $2.1V$, $2.3V$, $2.5V$, $2.7V$. The warning system allows the MCU to run an interrupt service routine that can save the program memory.

4.2.4 Analog to Digital Converter

The IO interface has eight pins which can be configured as inputs for the ADC which means the MCU can handle up to eight analog input sources such as sensors for environmental monitoring. This is one of the most important features for a node in a wireless sensor network. The ADC can be configured in a very flexible way for many different kinds of needs.

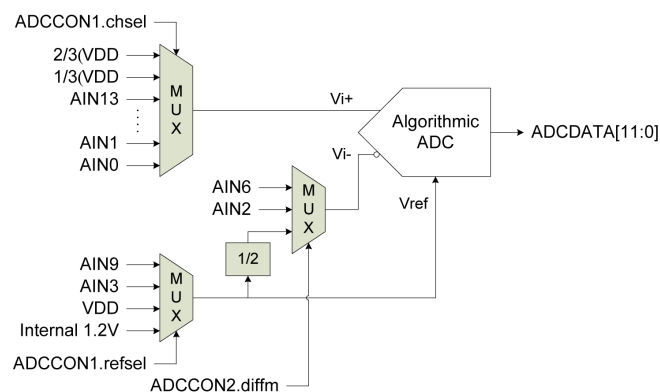


Figure 12: Block diagram of ADC [6, 165]

The block diagram in figure 12 shows the basic functionality of the ADC when an analog values is compared to a reference value. The result of this operation is stored as a binary integer in a provided data register.

Input Selection

Up to eight external and two internal input channels can be used and configured within the nRF24LE1D. The special function register ADCCON2 with the address 0xD2 can be used to configure the operational mode of the ADC. A single ended mode and a differential measurement mode are available. The **single ended** mode has the following range:

$$0V \dots V_{ref}$$

If the microcontroller's ADC is in the **difference measurement mode**, the available range is:

$$\frac{-V_{ref}}{2} \dots \frac{+V_{ref}}{2}$$

Reference Selection

The nRF24LED microcontroller can be configured with an external or also an internal reference voltage. The external reference source has to be connected with pin P0.3 and must be in range of at least 1.15V and maximum of 1.5V. The microcontroller itself offers in addition two reference sources. The supply voltage (VDD) or another source with 1.2V can be chosen as an inner source.

Resolution and Conversion Modes

The digital representation of the analog input values can have various resolutions, depending on the configuration of the register. The configurable resolutions are 6, 8, 10 and 12 bits. The higher the number of used bits, the higher the precision. This converts the analog value into a digital value.

The MCU can run in a continuous mode. which means the microcontroller frequently performs a conversion with a preconfigured sampling rate. The sampling rate can be specified with the values of 2000, 4000, 8000 and 16,000 samples per second (SPS). Furthermore, a single step mode can be used to take control over the moment of the conversion. The single step mode has to be triggered by the software.

Output Data Coding

The converted binary representation of the measured analog value is stored in the ADCDATA register which is a 16-bit register and consists of two 8-bit registers. Depending on the resolution more or less bits are needed. For example, in case of the 6-bit resolution configuration only the bits 0 up to 5 are used (starting with the LSB).

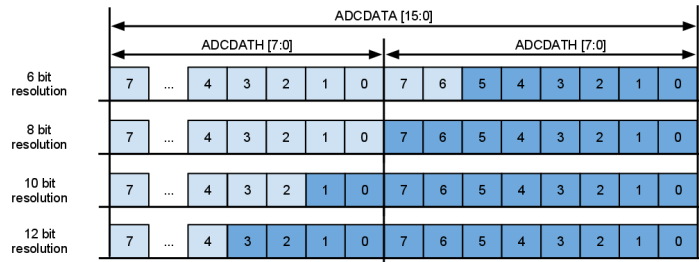


Figure 13: ADC data register

Figure 13 shows the use of the ADCDATH and ADCDATL bytes, where the highlighted bits are used for the data storage. The rest are unused and can contain any values.

Table 7: ADC output data coding

Mode	Voltage	Binary Representation ADCDATA
Single	$\leq 0V$	00 ... 00
	$\geq V_{ref}$	11 ... 11
Differential	$\leq -\frac{1}{2} V_{ref}$	00 ... 00
	$\geq +\frac{1}{2} V_{ref}$	11 ... 11

Table 7 shows how the analog input voltage is translated into digital values depending on the operational mode of the input pins.

4.3 nRF24L01+

The nRF24L01+ is an integrated but stand-alone operating radio frequency unit. The 2.4GHz ISM band is used by this microcontroller and supports *250kbs*, *1Mbps* and *2Mbps* transmission data rates. The nRF24L01+ can be programmed through the SPI interface by the main core of the nRF24LE1D MCU. This RF transceiver is able to handle up to six conversations simultaneously, which is due to six different data pipes of the nRF24L01+. This behaviour is nominated MultiCeiver™ by Nordic Semiconductor.

4.3.1 Radio Characteristics

The nRF24L01+ RF transceiver uses the gaussian frequency-shift keying (GFSK) modulation scheme. The GFSK is a special form of the normal frequency shift keying (FSK) modulation, where the binary information is formed into discrete frequency changes of the carrier wave. The GFSK is also used for other technologies like Bluetooth or Digital Enhanced Cordless Telecommunications (DECT). The nRF24L01+RF official supports up to 126 independent frequency channels and operates from *2400MHz* up to *2525 MHz*, but only the first 83 are allowed in all countries around the world. Especially in America it is not allowed to use the frequencies over *2483 MHz*. [7]

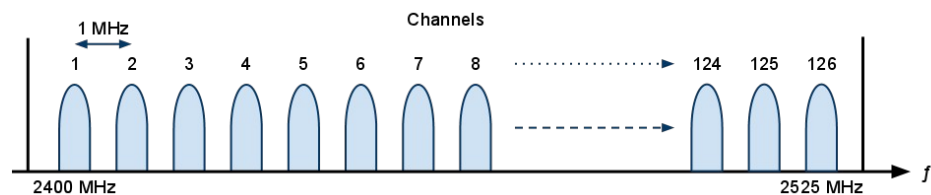


Figure 14: 2.4 GHz Nordic nRF24LE1D channel scheme [7, 3]

In both *250 kbps* and *1 Mbps* modes the nRF24L01+ uses a bandwidth of *1 MHz*, which differs from the *2Mbps* transmission mode where the *2 MHz* bandwidth is allocated. The channel occupancy for *1 Mbps* is *1 MHz* which brings a bandwidth efficiency of *1 bit/Hz*.

4.3.2 RF Transceiver Properties

The integrated RF transceiver has different states which are controlled by the main controller. These operational modes are the following:

- power down mode
- standby mode
- RX mode (receive)
- TX mode (transmit)

The different states and their relationships are shown in Appendix 1. The nRF24L01+ RF unit modes can be controlled through the SPI with the RFXCON register and the PWR_UP bit in the CONFIG register.

Power Down Mode

The power down mode is entered after a reset and disabling the RF transceiver. In this mode the RF unit consumes a minimum amount of energy. The RF unit can be activated by setting the PWR_UP bit high.

Standby Mode

After the transceiver has been in a TX or RX mode, the RF unit returns to the standby mode, which minimizes the power consumption while providing a short start up time.

RX Mode

The transceiver can be forced to switch into the RX mode if the PWR_UP bit, PRIM_RX bit and the rfce bit are set to 1 (high). In the RX mode the RF unit acts as a receiver and demodulates all incoming signals. If a valid packet is found, the user data (payload) are written to the RX FIFOs. The RF unit stays in the RX mode as long the MCU sets the PWR_UP bit to 0 or the Enhanced ShockBurst™ mode (cf. section 4.3.3) is activated.

TX Mode

During the TX mode the RF unit is transmitting packages. In order to enter the TX mode, the PWR_UP bit must be set to high and the PRIM_RX bit must be set to low. In addition, data must be available in the TX FIFO and a high pulse of more than 10 μ s must be present for the rfce bit. If there is no data left in the TX FIFO and there is no other high pulse on the rfce bit, the RF unit switches to the standby mode. If the Enhanced ShockBurst™ mode is activated, the modes are set by the ShockBurst protocol.

4.3.3 Enhanced ShockBurst

The Enhanced ShockBurst™ mode is a feature which can be enabled or disabled in the RF unit. This mode provides a set of features which allows an easy packet handling that automatically includes retransmissions if a packet is corrupted or lost. The automatic retransmission can be enabled or disabled, and it has some parameters for individual usage. These parameters are

- maximum number of retransmissions
- time between retransmissions

If a packet is lost or dropped because the maximum number of retries is reached, an interruption will occur and will be flagged.

After a transaction is initiated, a data packet is sent from the transmitter (PTX) to the receiver (PRX). If the Enhanced ShockBurst™ mode is activated, the nRF24L01+ RF unit automatically sets the PTX to receive mode to wait for the ACK packet. 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. 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 to receive mode to wait for the ACK packet [6 p. 22]. A detailed flowchart of the PTX and PRX operation during the Enhanced ShockBurst™ mode is listed in Appendix 2: PTX operations in Enhanced ShockBurst and Appendix 3: PRX operations in Enhanced ShockBurst.

In addition to the automatic acknowledgement of packages, the Enhanced ShockBurst™ mode offers a dynamic payload length from 1 to 32 bytes which makes the whole transmission process more energy efficient. All these functionalities are implemented in the Enhanced ShockBurst™ packet. The construct of this packet is shown in the figure 15.

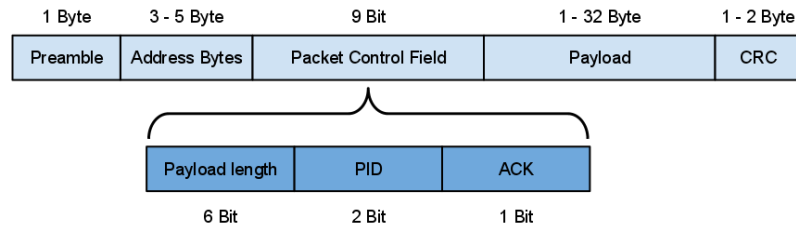


Figure 15: An Enhanced ShockBurst packet with variable payload

Preamble

The preamble is one byte long and can be *01010101* or *10101010* depending on the first bit of the following address bytes. The preamble is used to synchronize the demodulating receiver with the incoming bit stream.

Address Bytes

The five address bytes hold the unique address of the receiver.

Packet Control Field

The first 6 bits of the packet control field specify the length of the payload which can have a length of 1 up to 32 bytes. The dynamic packet length feature can also be disabled. If it is disabled, the payload length identifier will be ignored.

The following two bits are reserved for the packet identification (PID) which is increased on the PTX side for every new packet received from the MCU for transmission. This prevents the PRX MCU for receiving the same payload multiple times.

The no acknowledgment flag is presented in the last bit of the packet control field, and it controls if an acknowledgment is expected by the sender. If this bit is 1 (high), the receiver is forced to send an acknowledgement.

Payload

The Enhanced ShockBurst™ mode allows two different modes for the payload length. It can be set to a static length or to a dynamic one.

Cyclic Redundancy Check

The cyclic redundancy check (CRC) is a bit error detection algorithm. The checksum can be 1 or 2 bytes long, and it is calculated over the address of the receiving side. The polynomials for the calculation are the following, depending on the amount of bytes used:

$$CRC_{1\text{byte}} = x^8 + x^2 + x + 1 \quad \text{or} \quad CRC_{2\text{byte}} = x^{16} + x^{12} + x^5 + 1$$

These two formulas are implemented in hardware which offers fast calculations, and they are used to calculate the checksum of the payload.

4.3.4 MultiCeiver™ Characteristics

MultiCeiver™ is a trademark of Nordic Semiconductor it names a feature to connect to six other devices using data pipes. Each data pipe uses the same frequency channel and has its own physical address which is configured in the nRF24LE01+.

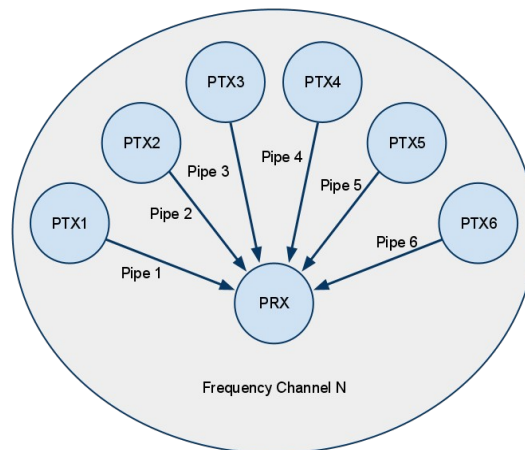


Figure 16: MultiCeiver™ data pipe model [5, p 37]

In figure 16 PRX (receiver) and PTX (transmitter) are connected through data pipe1, and they must use the same address in order to communicate. The transmitter address is configurable in the **TX_ADDR** register and the receiver address information is stored in the **RX_ADDR_P0**, **RX_ADDR_P1** up to the **RX_ADDR_P6** register depending on which data pipe is used. The address length is five bytes; however, only the whole address of the data pipe 0 is configurable. For the other pipes the least significant byte is changeable.

4.4 The nRFgo SDK

The nRFgo software development kit consists mainly of two parts, the hardware abstraction layer (HAL) and the Gazell link layer (gzll). This chapter will first discuss the HAL in more detail which is followed by a short introduction to the Gazell link layer. All parts of the nRFgo SDK are written in C language, and the source code can be directly used or if needed it can also be changed. All the files are written for the Keil C51 compiler. For this study the small device C compiler (SDCC) was used with the result that in some cases minor changes in the given source code were needed. All addressed files are listed in appendix 4.

The nRFgo SDK is under development by Nordic Semiconductor and can be downloaded from their website. In addition, the available packet consists also of some examples and a documentation as an *.chm* file (Microsoft Compiled HTML Help).

4.4.1 Hardware Abstraction Layer

The hardware abstraction layer offers a C language function set to access low level functions and registers. The functions are classified into architectural groups and consist usually of a C source (.c) file and a C header (.h) file. The header files builds the interface to the functionalities and the source files comprising the implementation. The file name starts with the leading letters **hal** followed by the group name. For example, the UART interface hardware abstraction source code consists of two files:

- hal_uart.h
- hal_uart.c

The functions implemented or the prototypes of these functions defined are named in a similar way. The name begins with **hal** following by the group name. The identifier comes after the group name which points at the feature the function provides. For example, the initialization function of the UART packet shows this name convention.

```
void hal_uart_init(hal_uart_baudrate_t baud);
```

The objective of this chapter is not to explain all function or details of the nRFgo software development kit; however, it should merely give an overview about their applicability.

4.4.2 Serial Peripheral Interface (*hal_spi*)

As already described in section 4.2.2, the nRF24LE1D provides an SPI interface which can operate in two modes, the master and slave mode. The clock speed can be configured and the settings are described in specific enumerations. These enumerations are defined in the *hal_spi.h* file together with the function prototypes. This header and source code file is part of the HAL of the nRF24LE1D software development kit.

```
enum hal_spi_clkdivider_t
enum hal_spi_byte_order_t
enum hal_spi_mode_t
```

In addition to the already specified capabilities, the bit order can also be defined with the *hal_spi_byte_order_t* enumeration. This enumeration stores the bit order, which defines if the LSB or the most significant bit (MSB) is the first bit transmitted or received, and it must be passed to the initialization function.

The functions in the SPI hardware abstraction layer are fragmented into master and slave specific functions which give access to the settings of the SPI functionalities depending on the mode which is needed during the operation of the MCU. These SPI related functions are implemented in the *hal_spi.c* file and are listed in the appendices (cf. Appendix 4).

The settings are set for both modes with an initial function. The nRFgo SDK provides a specific predefined function for each mode, which allows fast configuration of the SPI bus. Both initialization functions need the enumerations passed as parameters in order to preconfigure the bus.

```
void hal_spi_master_init
void hal_spi_slave_init
```

After the first setup has been done, the read and write functions can be used in order to transmit data to the SPI bus. The read function performs a read operation to the SPI FIFO shift register and returns the values while the write operation puts a character to the SPI output register.

```
uint8_t hal_spi_master_read_write
uint8_t hal_spi_slave_rw
```

The above listed functions are not all of the given ones but are the most notable. It is out of the scope of this thesis to discuss all libraries in full detail.

4.4.3 UART Interface (`hal_uart`)

As mentioned before, the MCU can be configured with an UART interface which is identical to the standard 8051 serial port. The UART interface is a simple and widespread technology. All the source codes of the `hal_uart` are also written in the C language and the implementation can be found in the files `hal_uart.c`, where the prototypes of the functions and the enumeration are defined in the `hal_uart.h`. These files are listed in the Appendix 4.

```
enum hal_uart_baudrate_t
```

The most important setting is the baudrate of the serial connection. This setting is configured with the initializing function for the UART interface.

```
void hal_uart_init (hal_uart_baudrate_t baud);
```

The other functions provide read and write access to the transmission and receive buffer or allow to monitor if a character is received. In order to send a character to the output buffer, the information is passed to the function `hal_uart_putchar()`. The function call also initiates the content of the output buffer will be transmitted.

```
void hal_uart_putchar (uint8_t ch);
uint8_t hal_uart_getchar (void);
```

The UART function which reads the content of the input buffer waits until information is available and returns the value.

```
bool hal_uart_tx_complete ();
```

This function can be used to check if the entire transmission output buffer is empty. The return value is true if all data have successfully been transmitted.

4.4.4 2-Wire (`hal_w2`)

As stated in chapter 4.2.2, the nRF24LE1D supports different operation modes of the 2-Wire interface which are the master and slave mode. In order to configure the settings of the MCU, the nRFgo SDK provides easy access with some prewritten functions as well. These settings are described in the following enumerations and defined in the `hal_w2.h` header file.

```
enum hal_w2_clk_freq_t
enum hal_w2_op_mode_t
enum hal_w2_irq_source_t
```


The functions of the *hal_w2* library are classified into common, slave and master specific functions. The common functions can be used to set and control the properties of the 2-Wire interface such as the clock frequency or the operational mode. The following functions are listed as examples and do not represent the full set of available configuration functions. sending and transmitting of data are also part of the common functions.

```
void hal_w2_set_clk_freq (hal_w2_clk_freq_t freq)
void hal_w2_set_op_mode (hal_w2_op_mode_t mode)
void hal_w2_enable (bool en)
void hal_w2_write_data (uint8_t tx_data)
uint8_t hal_w2_read_data (void)
```

The set of slave functions allows to configure the slave with an address as well as other slave specific options. It can also alter the clock frequency given by the master if the slave is not able to process the incoming data. This set of functions is only available during the slave mode.

```
void hal_w2_alter_clock (bool alt_clk)
void hal_w2_irq_stop_cond_enable (bool stop_cond)
void hal_w2_irq_adr_match_enable (bool addr_match)
void hal_w2_set_slave_address (uint8_t address)
```

Master functions can initiate the data transmission with a start condition or stop the transmission with a stop condition. These are only available in the master mode of the 2-Wire interface.

```
void hal_w2_transmit_start_cond (void)
void hal_w2_transmit_stop_cond (void)
```

During the whole transmission process the master is responsible for controlling the communication. This includes initiating the transmission but also terminating the transmission.

4.4.5 Power-fail Comparator (*hal_pof*)

The power fail comparator can be controlled over the *hal_pof* part of the SDK. Analogously to the other already noted parts of the provided libraries, the *hal_pof* is divided into the *hal_pof.h* header file and a source code file named *hal_pof.c* (cf. Appendix 4). The threshold is described by the following enumeration:

```
enum hal_pof_threshold_t
```

The threshold can be set to one of the four values, which are *2.1 V*, *2.3 V*, *2.5 V*, *2.7 V*. These can be configured and maintained by the given functions which are implemented and listed in the *hal_pof* files. These functions can enable the power-fail monitoring, set the threshold voltage level and check if a power-fail occurred.

```
void hal_pof_enable (bool enable)
void hal_pof_set_threshold (hal_pof_threshold_t threshold)
bool hal_pof_warning (void)
```

This part of the library enables functionalities in order to monitor the power supply source. Functionality such as the power fail monitoring system is significant for battery powered applications.

4.4.6 Analog to Digital Converter (*hal_adc*)

One of the essential components of the SDK is the *hal_adc* part. The ADC library provides an easy access to the settings and the general task of the analog to digital conversion process. The functionality of the ADC is already described in section 4.2.4. All configuration options are described with c style enumerations in all the other components. Enumerations exist for a couple of settings which are

```
enum hal_adc_input_channel_t
enum hal_adc_reference_t
enum hal_adc_sampling_rate_t
enum hal_adc_resolution_t
enum hal_adc_conversion_mode_t
```

These are not all of the enumerations but the most interesting ones. The ADC module of the SDK contains functions for configuration, starting the ADC, reading status information and reading the sampled values. The functions of the ADC are structured into setup functions and overall operation functions. The setup functions are meant to facilitate the startup configuration of the ADC and to ease the access to the properties. For example, the following extract of the **setup functions** illustrates this.

```
void hal_adc_set_input_channel (hal_adc_input_channel_t chsel)
void hal_adc_set_reference (hal_adc_reference_t refsel)
void hal_adc_set_input_mode (hal_adc_input_mode_t input_mode)
void hal_adc_set_sampling_rate (hal_adc_sampling_rate_t rate)
```

The operation functions are used to read the ADC values from the register ADCDATA which is described earlier in this thesis (cf. section 4.2.4).

```
void hal_adc_start (void)
uint8_t hal_adc_read_LSB (void)
uint8_t hal_adc_read_MSB (void)
```

The result of the actual analog to digital conversion is returned as a *16bit* value separately as the LSByte and MSByte.

4.4.7 Gazell Link Layer

The Gazell Link Layer is a library of functions for controlling and configuring the link layer of the wireless connection between two nRF24LE1 devices. The Gazell Link Layer also handles the underlying Enhanced ShockBurst mode (cf. section 4.3.3) and the MultiCeiver™ (cf. section 4.3.4) capabilities of the nRF24L01+ RF unit.

Properties of Gazell

The Gazell library makes use of all the features which lower energy consumption. Therefore, the Gazell offers different modes such as of the symmetric mode where the host and device are in sync with a clock. As a result, the device and the host do not need synchronization messages which save energy and which are imported specially in a battery based application.

The Gazell link layer also supports mechanism against interference over frequency hopping which means that the device would switch the channel for each transmission. AES encryption is also supported. The encryption can be configured for each link if required.

Protocol States

The Gazell link layer can set the MCU in idle, host or device state. These states represent the discussed states in the Enhanced ShockBurst™ (cf. section 4.3.3) of the nRF24L01+ RF unit on a higher level.

The **IDLE** state is an abstraction on the C language level of the standby or PowerDown mode. This mode can be forced with the following function at any time.

```
void gzll_goto_idle (void)
```

During the **HOST** state the device is in the listening mode for incoming data. Received packages are stored into the RX FIFO. In order to check if any data is available, the following function can be used:

```
bool gzll_rx_data_ready (uint8_t pipe)
```

Data can be read from the RX FIFO with the following command:

```
bool gzll_rx_fifo_read (  uint8_t *dst,
                          uint8_t *length,
                          uint8_t *pipe)
```

Figure 17 offers an overview about the relation between the states and function calls for the IDLE and HOST mode. It is illustrated that if the function `gzll_rx_start()` is called and the device is in IDLE mode it will switch into HOST mode. In addition, the reasons for a transaction from Host to IDLE mode are described.

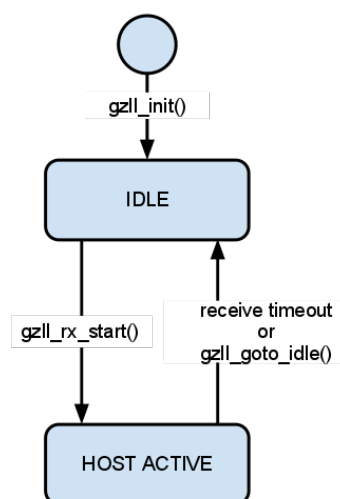


Figure 17: Gazell protocol states for a Host

The Gazell protocol functions provide functionality for putting the MCU into the **DEVICE** state. This state of the Gazell link layer can be entered with the following function call:

```
bool gzll_tx_data (  uint8_t * src,
                    uint8_t   length,
                    uint8_t   pipe )
```

The microcontroller stays in the device state as long as there is data available in the TX FIFO or a transmit timeout occurs. The state diagram is similar to the host states and is shown in the Figure 18.

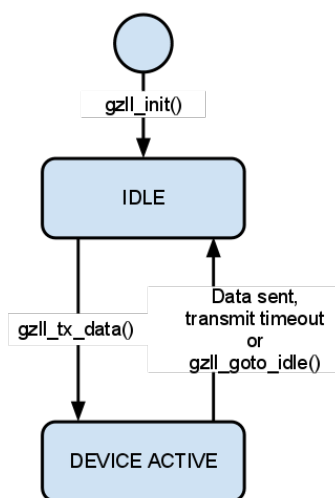


Figure 18: Gazell protocol states for a device

The Gazell link layer can be configured by setting static and dynamic parameters so that the Gazell protocol can be customized for a wide range of different applications. A selection of examples is listed below

- TX_TIMEOUT defines the total number of transmit attempts
- CRYPT_PIPES defines which data pipe is encrypted
- OUTPUT_POWER defines the radio output power
- GZLL_DATARATE defines the radio data rate

The Gazell link layer is a flexible protocol that helps the user to build a stable communication network with characteristics such as robust against interference, low power consumption and easy to setup and maintain.

5 Discussion

The nRFgo SDK source code files are precompiled and optimised for the Keil51 compiler. The Keil51 C Compiler is one of the first C compilers available for the 8051 microcontroller architecture, and it is a commercial product. During the test implementation for this study version 3.0.0 of the Small Device C Compiler (SDCC) was available. The SDCC is an open source compiler which is optimised for the C language. Its targets are 8-bit microprocessors with Intel's MCS51 architecture including the 8031, 8032, 8051 and 8052 structures [8].

During the development of some practical test programs it appeared that the Small Device C Compiler cannot compile without changing the nRF SDK library files, because there are some mismatches in the required C language implementations. There is incompatibility between the Keil51 and the SDC compiler, for example, in the storage class language extensions. This storage class identifier can be the term *data* which is usually the default storage class. This is declared in the **Keil51** compatible code as

```
data unsigned char test_data;
```

However, for the **SDCC** the code has to be the following

```
__data unsigned char test_data;
```

Another example of the incompatibility is in the definition of interrupted service routines. The **Keil51** accepts the following code example:

```
static void uart0_isr(void) interrupt 4
```

However, the **SDCC** requires the following syntax for interrupted service routines:

```
static void uart0_isr(void) __interrupt 4
```

As a result, in almost all nRFgo SDK source files adjustments are required in order to use the provided nRFgo functionality.

Another problem was noticed during the analysis of the test programs. Troubleshooting of running software can be difficult especially if the microcontroller makes use of the wireless links. Searching for errors between two devices with a wireless connection requires special equipment which was not available during the tests. It has been observed that the incompatibility of the SDCC and the developing wireless communication links are the most challenging problems.

6 Conclusion

Because the nRF24LE1 microcontroller has a system on a chip architecture it comes with several features implemented in hardware and software ,which opens a wide range of prospects. Especially in the field of small devices where a wireless communication is required for different kind of reasons, this microcontroller is suitable for many applications. A typical area where small low cost devices are needed is the wireless sensor network field.

The nRF24LE1 can be used as a sensor node, because it provides an analog to digital converter paired with a library that gives fast access to this hardware feature. In addition, it also has a built in radio frequency unit which allows to transmit data between these devices. An already implemented wireless network protocol with automatic acknowledgement system and frequency hopping is provided which makes this microcontroller suitable for the adoption in a sensor node hardware.

This MCU is also deployable as a sink, because it has built in capability for a PC communication. The processor is fast enough to handle the collected data in a WSN it also has the air interface speed of up to *2Mbps*.

The RF unit is also available as a stand alone microcontroller and it can be connected to other types of microcontrollers in order to use for example an Ethernet network interface to connect to other kinds of networks or even to the Internet. Another microcontroller similar to nRF24LE1 is available, which offers in addition a Universal Serial Bus (USB).

This small low cost and low energy consumption microcontroller offers a wide range of applications. Especially the built in functionality joint with the provided SDK makes this microcontroller a very interesting device for further development in the field of embedded wireless communication.

Future Prospects

The wireless sensor networking technology is a growing market especially in the field of consumer products which will require solutions more similar to the nRF24LE1 microcontroller in the future. The nRF24LE1 microcontroller is only one of many products developed by Nordic Semiconductor. The study showed that the MCU can be used in various of applications.

For example, a connection to a smartphone could be built and used to send data from a wireless sensor network through a mobile network to a sever or another data storage. The interface for this type of connectivity is available in the nRF24LE1 and could be used in future projects. Modern smartphones offer a powerful SDK that allows the programmer interconnection to external hardware such as the discussed microcontroller. These handheld devices can replace PCs in the process of data analysis.

Another possible project for future research is the integration of the Self-Organizing Mobile Routing Network (SOMORON) protocol developed by the student Olaitan Olamilehin at the Helsinki Metropolia University of Applied Sciences during her final year project. Because the nRF SDK provided by Nordic Semiconductor is optimised for the Keil51 compiler, there is a need for optimisation if using other compilers, such as the Small Device C Compiler (SDCC). The Small Device C Compiler is an open source project and is based on the GCC compiler. There is a project which rebuilds the nRF SDK to be compatible with the SDCC compiler. This could also be an interesting field for future research on the nRF24LE1 microcontroller.

References

- [1] Lewis F.L. Wireless sensor networks [online]. Online Education. University of Texas, Arlington; 2004.
URL: <http://arri.uta.edu/acs/networks/WirelessSensorNetChap04.pdf> .
Accessed on 10 April 2011.
- [2] Dargie W. Poellabauer C. Fundamentals of wireless sensor networks theory and practice. 1st ed John Wiley & Sons Ltd; 2010
- [3] Verdone R, Dardari D, Mazzini G, Conti A. Wireless sensor and actuator networks. 1st ed. Oxford UK. Linacre House Jordan Hill; 2008
- [4] Nordic Semiconductor. NRF24LE1 2.4GHz RF System-on-Chip with Flash [online].
URL: <http://www.nordicsemi.com/eng/Products/2.4GHz-RF/nRF24LE1> .
Accessed on 13 April 2011.
- [5] Nodic Semiconductor. nRF24L01+ Single Chip 2.4GHz Transceive [online].
URL:
http://www.nordicsemi.com/eng/nordic/download_resource/8765/2/13195198 .
Accessed on 20 April 2011.
- [6] Nodic Semiconductor. nRF24LE1 Ultra-low Power Wireless System On-Chip Solution [online]
URL: http://www.nordicsemi.com/eng/nordic/download_resource/7493/3/20300954
Accessed on 14 April 2011.
- [7] Nodric Semiconductor. White Paper LowCostNetworks,ZigBeeTM &802.15.4 [online]
URL: http://www.nordicsemi.com/index_popup.cfm?obj=product&act=displayWhitepapers&pro=89&con=white_papers .
Accessed on 10 April 2011.
- [8] SDCC Compiler User Guide, [online]
URL: <http://sdcc.sourceforge.net/doc/sdccman.pdf> .
Accessed on 2 May 2011.

Appendices

Appendix 1: Radio control state diagram

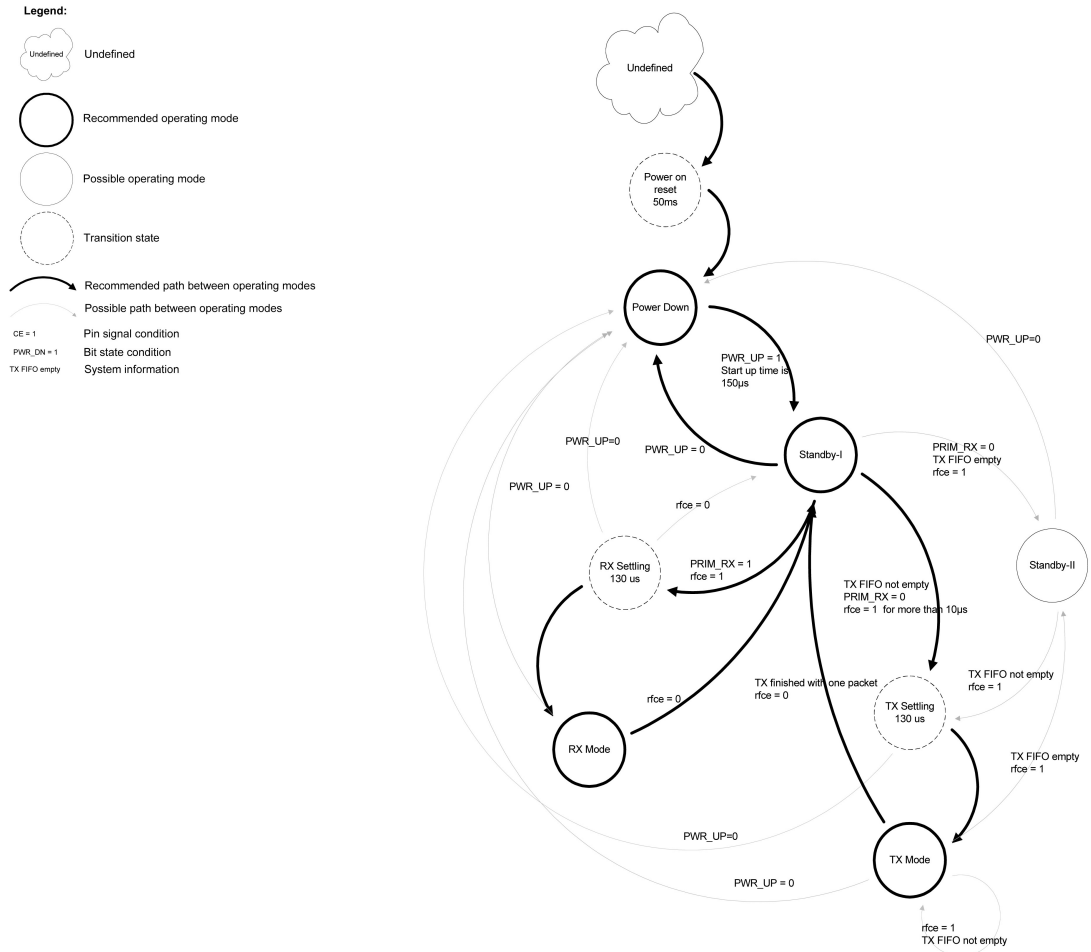


Figure 19: Radio control state diagram [6, 18]

Appendix 2: PTX operations in Enhanced ShockBurst

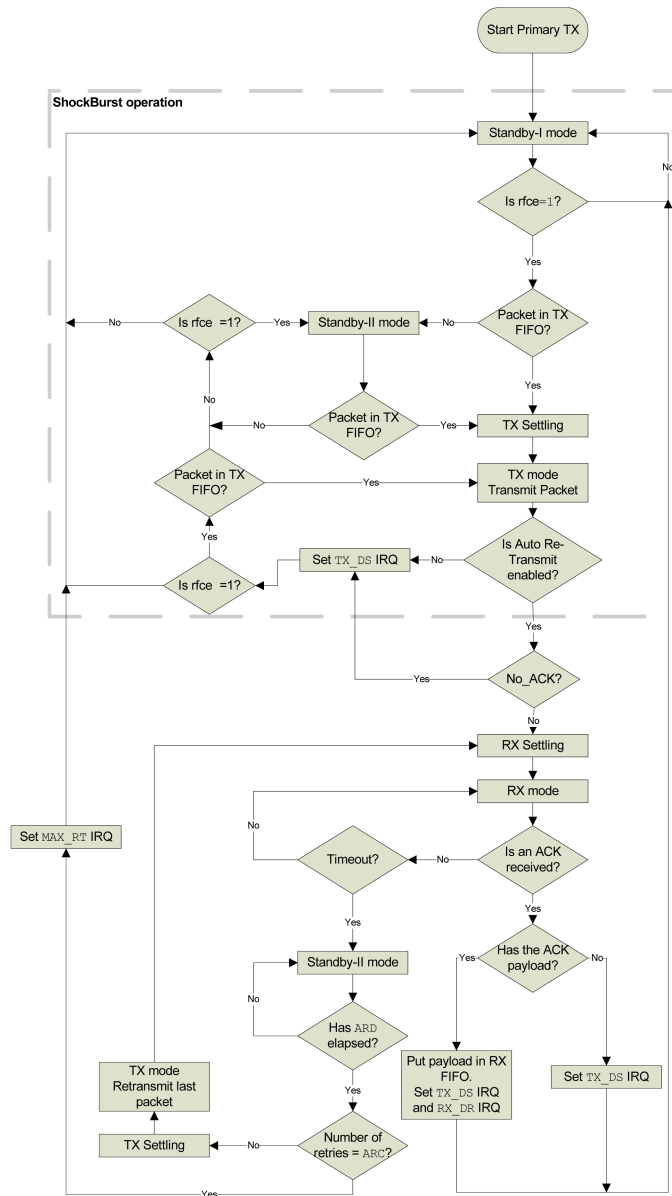


Figure 20: PTX operations in Enhanced ShockBurst [6, 30]

Appendix 3: PRX operations in Enhanced ShockBurst

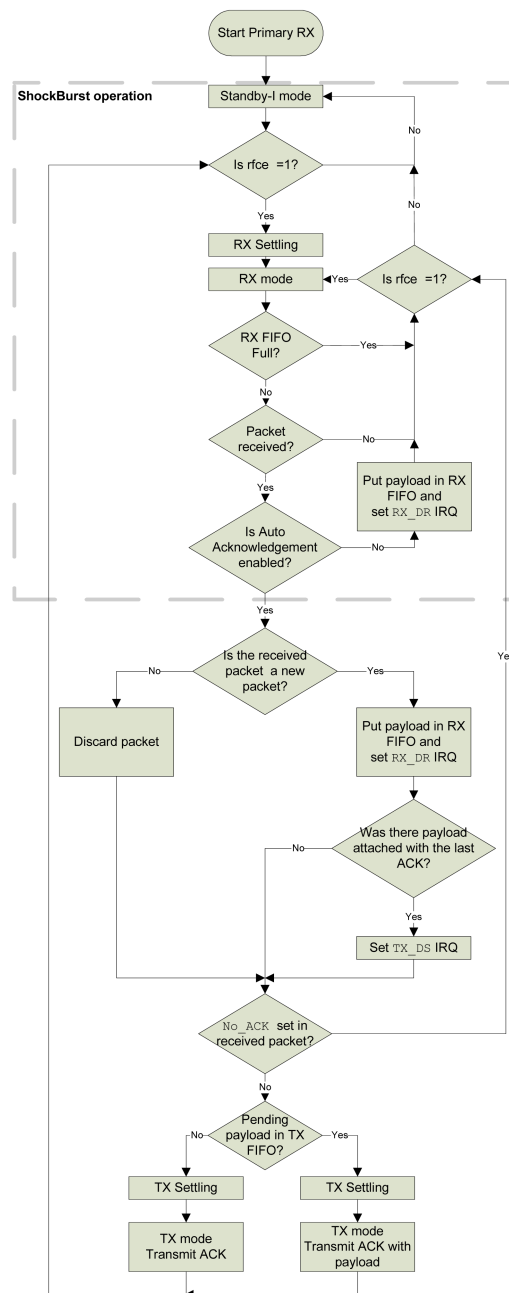





Figure 21: PRX operations in Enhanced ShockBurst [6, 32]

Appendix 4: nRFgo SDK

Table 8: Content of the enclosed CD

Directory	File Name	Description
 hal		
 nrf24le1	hal_spi.h	Interface functions for the Serial Peripheral Interface (SPI)
	hal_spi.c	Implementation of hal_spi
	hal_uart.h	Interface functions for the Universal Asynchronous Receiver-Transmitter (UART)
	hal_uart.c	Implementation of hal_uart
	hal_adc.h	Interface functions for the analog-to-digital converter (ADC)
	hal_adc.c	Implementation of hal_adc
	hal_2w.h	Interface for the 2-Wire module
	hal_2w.c	Implementation of hal_w2
 documents	nRF24LE1_Product_Specification_rev1_6	Product specification and datasheet of the nRF24LE1 microcontroller
	nRF24L01Plus_Preliminary_Product_Specification_v1_0	Product specification and datasheet of the nRF24L01+ Radio-frequency microcontroller
	nrf_sdk.chm	nRF Software development Kit documentation and development guidelines
	sdccman.pdf	Documentation and manual of the SDCC compiler version 3.0.1