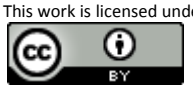
Available online at: https://ijact.in

Date of Submission	11/09/2018
Date of Acceptance	23/09/2018
Date of Publication	30/09/2018
Page numbers	2807-2813 (7 Pages)



ISSN:2320-0790

An International Journal of Advanced Computer Technology

DETECTION AND DECODING OF ASK SIGNALS USING AN SDR RECEIVER

Dr.K.V.Ranga Rao[1], A.Ravikumar[2], R.Kumaraiah[3], Venkat Ritesh Ghanta[4]

[1]Professor, CSE Dept,Vidya Jyothi Institute of Technology, Aziz nagar
[2]Assistant professor, Vidya Jyothi Institute of Technology, Aziz Nagar, Hyderabad, Telangana, India
[3]Lab Assistant, Maturi Venkata Subba Rao Engineering College, Nadergul, Saroornagar Mandal, Dist. Ranga Reddy
[4]M.tech 2nd Year, IIT Bombay, Maharastra, India

Abstract: On-Off Keying (OOK) is one of the most basic form of communication widely used today. The objective of this paper is to receive, detect and decode an OOK signal in the radio spectrum. For the reception of the radio signal, an SDR receiver is used. This device receives the RF signal and down-converts it to a frequency usable by an inbuilt ADC. The ADC outputs I/Q samples which are then forwarded to a computer via the USB port. By processing the received signal, it is determined whether or not, the received signal is an OOK signal and if it is, then it is further processed and decoded. There are two units in the setup: transmitter and the receiver. The transmitting side consists of an MCU connected to a transmitting circuit. The MCU outputs the digital information to be transmitted. On the receiver side are a computer and an SDR receiver. The receiver used here is a RTL-SDR. The major part of this paper is the signal processing at the receiver end. The processing is done in the computer in C language. The samples from the RTL-SDR are forwarded to a C program. Various algorithms are then implemented in the program for detection and decoding.

Keywords: On-Off Keying, RTL-SDR, MCU, MATLAB, C program, Python

I. INTRODUCTION

The primary objective of this paper is to demonstrate the capabilities of an SDR receiver by automatically detecting and decoding various signals in the radio spectrum. A multitude of tasks can be done by using an SDR receiver but this paper focuses mainly on spectrum usage and signal discovery. The ideas put forth in this paper can also be used to build an RF data acquisition system. The information acquired by this system can be fed to a cognitive radio engine which can make appropriate decisions. This paper aims at reducing the complexity of the physical layer of a communication system by passing on some of its functions to higher levels. The basic blocks in the kind of system proposed are an RF front end, an ADC and DAC and a processing unit. This puts a lot of importance on the RF front end which performs the first primary reception of the signal and also the ADC/DAC which must be fast enough to convert the high frequency signals to samples in order to support higher bandwidths.

A. Existing Model:

The radio receivers currently being used perform the following main functions:

i. Allow us to receive the desired signal by manually tuning.
ii. Amplification of the received signal
iii. Demodulation

These functions imply that the user of the receiver has a prior knowledge of the signal and the frequency at which it exists. A better receiver would be one which can receive a wide range of signals automatically, list out the different signals and the frequencies at which they are present and also have the capability of demodulating it and further decoding it. Also Most of the RF communication systems used today is built on a conventional model which is specifically designed to work as part of a certain network and with a fixed modulation technique.

This model leads to a lot of dependency on the hardware and the lack of certain hardware modules in devices will not let those devices use the features that these modules have to offer. This dependency is evident in today's smart phones, some of which support 4G and others which don't because they lack the newer hardware modifications required for 4G.

B. Proposed Model:

Radios can be more useful if they perform multiple tasks on the same minimal hardware. Therefore, this paper proposes to replace the current RF hardware with SDR hardware wherever possible and hence attempt to eliminate the hardware dependency of modulation techniques, communication protocols and type of network on specific hardware. This can be achieved by making a receiver which can detect and identify a signal being sent and take action accordingly. For instance, if a Wi-Fi signal is detected, the receiver will perform the functions that a typical Wi-Fi signal receiver would have performed, and if a Bluetooth signal is detected, then the receiver performs the functions that a dedicated Bluetooth module would've performed. The functions mentioned above are possible because in an SDR, some essential hardware blocks are implemented in software and hence can be programmatically controlled.

Figure 1: Software Defined Radio Block Diagram

In this paper, the use of SDR for the detection and decoding purposes is proposed mainly because the down-converted RF signals are directly processed in software which is easier to do than constructing hardware blocks which do the same.

II. HARDWARE/SOFTWARE SETUP

The paper setup consists of a transmitting side and a receiving side however; the receiving end is the one central to the paper. The transmitters are present only to test the receiver. Both the transmitters transmit ASK signals.

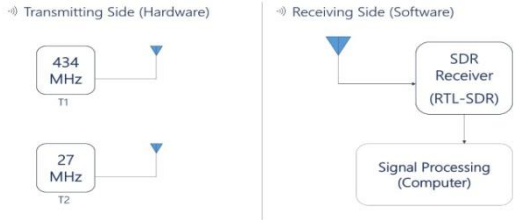


Figure 2: Overview of setup

A. Transmitter Side:

For Testing: The transmitter side comprises of two transmitters: one transmitting at 27 MHz and another at 434 MHz. The spectra of the signals transmitted by these transmitters are the same. The message signal is fed to both the transmitters from an ATMEGA16A MCU.

i. **27 MHz Transmitter**: The 27 MHz transmitter is simply a Pierce crystal oscillator connected to 4N35 optocoupler for ASK modulation. The supply to the oscillator circuit is controlled by the optocoupler and the digital input at its 2.

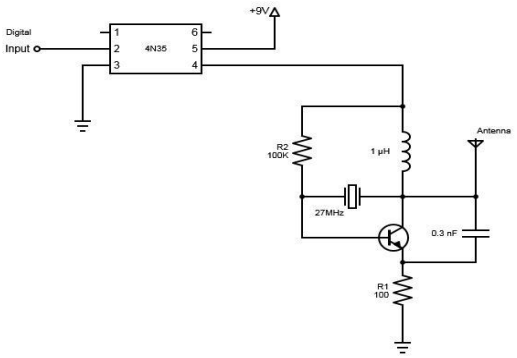


Figure 3: Circuit Schematic: 27 MHz Transmitter

ii. **434 MHz Transmitter:**

This is a generic 434 MHz ASK transmitter module. The output from the MCU (PORTB.0) is connected to the Data pin of this transmitter.

B. Receiver Side:

i. **SDR Receiver:** The RF signal at the antenna is filtered and amplified and is converted to an Intermediate Frequency (IF). IF is usually 0 Hz (DC). This intermediate signal is sampled by the 8-bit ADC inside the receiver at a sampling rate defined by software on the computer. Also, the receiver can be tuned using software.

Figure 4: Receiver Block Diagram

The receiver outputs 8 bit unsigned char interleaved I/Q samples which can be processed in the computer.

ii. **Computer: Signal Processing:**

The samples can be processed using just about any capable programming language. For detection purposes, Python programming is used. For decoding, C programming is used. The complex unsigned char samples are converted to float magnitude samples for processing because for ASK modulation, phase doesn't contain any information. The process of detection is illustrated using the block diagram given below.

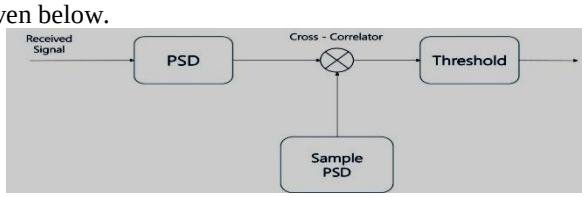


Figure 5: Block Diagram: Signal Detection

After a signal is detected and its frequency is found, the receiver is tuned to it and the samples are forwarded to a C program for decoding. In the context of this paper, decoding can be defined as the process of recovering binary bits from the samples of the received signal and then obtain the characters which the bits represent. In this paper, characters are ASCII coded. The process of signal decoding can be understood by the block diagram below.

Figure 6: Block Diagram: Signal Decoding

III. Hardware And Design

The hardware used in this paper is minimal and is used only for demonstration and testing purposes however, information about it is outlined in the rest of the chapter.

A. Hardware: Receiver Side:

The main piece of hardware used in the paper is the RTL-SDR Receiver which supplies the samples of the received signal to the computer via a USB port. The two main ICs used in the receiver are the R820T and the RTL2832U. The receiver derives its name from the latter.

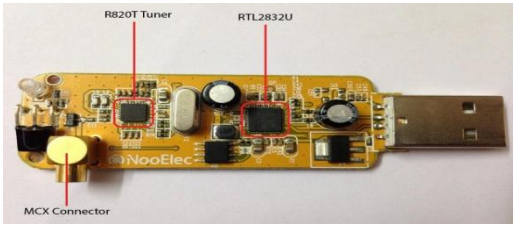


Figure 7: RTL-SDR Receiver

The receiver supplies samples in blocks and uses buffers. The data can be received in two ways: synchronous mode and asynchronous mode. In this paper, the samples are read in synchronous mode for signal detection and in asynchronous mode while decoding the signal.

i. **R820T Tuner:**

The R820T chip is a tuner that tunes to a particular frequency and receives the RF signal. It has built-in Low Noise Amplifier (LNA), mixer, fractional PLL, Variable Gain Amplifier (VGA), voltage regulator and tracking filter.

Manufacturer	Rafael Micro
Tuning Range	25 MHz – 1750 MHz
Noise Figure	3.5 dB
Image Rejection	65 dBc

Table 1: R820T Tuner Specifications

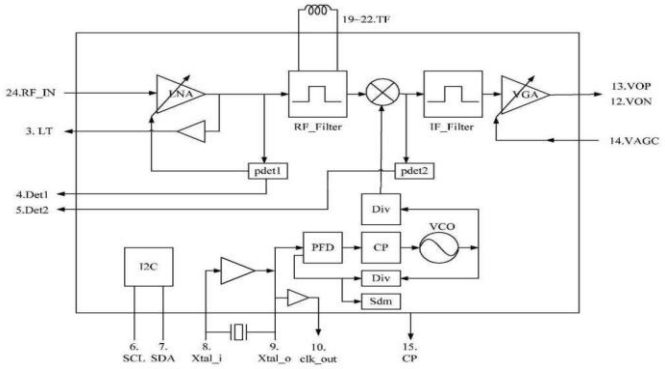


Figure 8: Block Diagram: R820T

ii. **RTL2832U:**

The RTL2832U is actually a DVB-T COFDM demodulator but it is used differently in this paper. It has an in-built 8-bit ADC. Complex Sampling (I/Q sampling) is performed in this chip and the interleaved real and imaginary sample values are sent to the computer via the USB port.

B. Hardware: Transmitter Side:

There are mainly three units of hardware on the transmitting side:

a. ATMEGA16A
b. 434 MHz ASK Tx Module
c. Custom 27 MHz ASK Module

i. **AVR ATMEGA16A:**

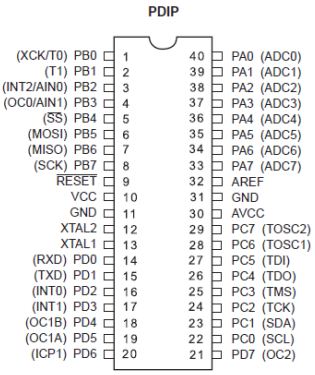


Figure 9: Pin Diagram: ATMEGA16A

In this paper, the MCU is the digital information source. '1' and '0' are represented using rectangular pulses of 75% and 25% duty cycle respectively. The receiver is made independent of the duration of these pulses. It is operated at 1 MHz using the default internal oscillator.

434 MHz ASK Transmitter (WS-TX-01)	
Frequency	433.87 MHz
Modulation	ASK
Data Rate	?
Supply Voltage	3 – 12 V

Table 2: 434 MHz Tx Specifications

27 MHz ASK Transmitter	
Frequency	27 MHz
Modulation	ASK
Data Rate	400-500 bps
Supply Voltage	9 V

Table 3: Custom 27 MHz Tx Specifications

C. *Design Methodology:*

i. **Signal Detection:** The primary function of the receiver section is the detection of an ASK signal in the available radio spectrum. The spectra of various ASK signals very closely resemble each other therefore; this fact can be exploited to make detection of these signals possible. The process of detection happens in 4 steps which are listed below:
   a. Tuning to a frequency
   b. Acquiring the signal, then finding its PSD
   c. Cross-Correlating this PSD with a predefined sample PSD
   d. Examining the result of Correlation to determine if it's a match

Before these steps are performed however, the sample PSD needs to be defined. To obtain this sample PSD, the receiver is tuned to a frequency at which it is known that a strong enough ASK signal already exists. Then this signal is acquired and its PSD is found and saved to a local file. The sample PSD used in this paper is shown below.

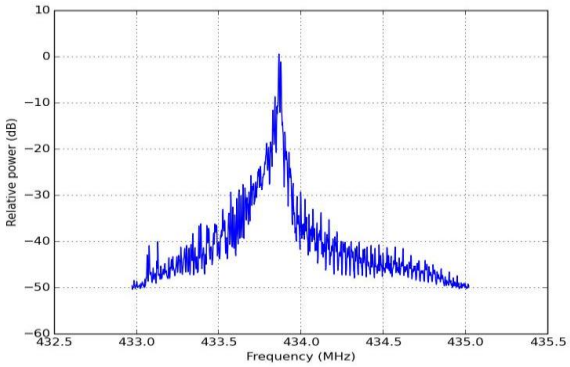


Figure10: The ASK Sample PSD Plot

ii. **Signal Decoding:**

After the frequencies at which the desired signals are present are obtained, the receiver is tuned to any one of the frequencies and then the samples of the received signal are forwarded to a C program which performs the decoding operation. In the C program, the decoding is done by first evaluating the magnitudes of the complex samples and then passing them through a 40 point averaging filter to smoothen the received signal. To obtain the binary bits from the smoothened signal, the duty cycle of each pulse is evaluated and it is determined if a 1 or 0 is encountered by the following rule:

$$\text{duty cycle} > 50\% \rightarrow 1$$

$$\text{duty cycle} < 50\% \rightarrow 0$$

This process is advantageous because it eliminates the dependency of decoding process on the duration of the pulses used to represent the binary bits. For example, a pulse with ON time of 25us and OFF time of 75us and another pulse with an ON time of 500us and OFF time of 1500us both represent a 0 and the process of decoding described above works equally well for both.

iii. **Communication Protocol:**

The pulses representing 1s and 0s of arbitrary duration are transmitted one after the other without delay. To transmit a character, 8 bits are serially transmitted from LSB to MSB. To indicate the end of a character, a pulse with an arbitrary ON duration is sent and is followed by an amount of delay greater than the duration of the pulses. This works because the duty cycle of this indicator pulse is well less than the 25% and hence can be easily detected by code.

IV. SOFTWARE SPECIFICATIONS:

The functions of the receiver are completely defined in software using two programming languages: C and Python. The RTL-SDR receiver is interfaced to the computer using the drivers and libraries written for it. The library that provides access to the receiver is called 'librtlsdr'. This library provides the various functions used to control the receiver like tuning, setting gain and sampling rate, etc.

A. Signal Detection in Python:

'librtlsdr' can be directly used in C programs by including the necessary files but the same cannot be done with Python. So there is a wrapper of this library for python called 'pyrtlsdr' which provides all the functions in 'librtlsdr' in python.

B. Signal Decoding in C:

The decoding process is performed by a C program but this program does not directly receive samples from the receiver. Another program called 'rtl_sdr' is used to receive the samples and then they are forwarded to the C program through piping.

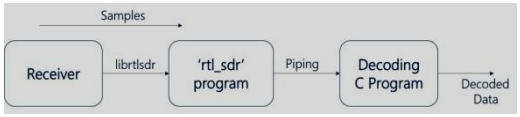


Figure.11 Sample Flow: Block Diagram

The following is the syntax to use the 'rtl_sdr' program:

rtl_sdr -f [center_frequency] -g [gain] -s [sampling_rate] -n [no. of samples]

The samples are read and written to the file specified in the command. Alternately, the command can be terminated with a '-' instead of the file name to dump the samples to stdout. This alternative is used in the paper. The standard output of the rtl_sdr program is fed as standard input to the decoding C program. The C program is compiled using the gcc compiler in Ubuntu.

V. RESULTS

A. Signal Detection:

i. **Final Result:**

The following was the final result of signal detection when two transmitters at 434 MHz and 315 MHz were present:

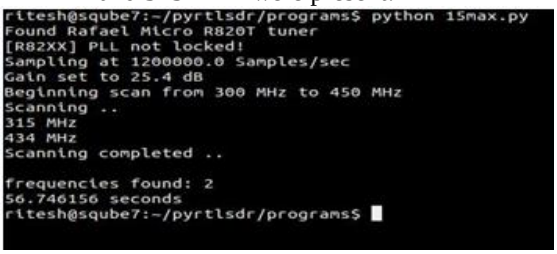


Figure12: Result-Signal Detection

It can be seen that the time taken for scanning from 300 MHz to 450 MHz (150 MHz) is around 57 seconds which is quite large. This is because Python is not totally performance efficient. The speed can be increased by increasing the sampling frequency. Implementing the same program in C will also increase the speed and performance.

ii. **Intermediate Results:**

These are the results obtained when tuned to two different frequencies: 200 MHz where there is no ASK signal and 434 Mhz where the signal is present.

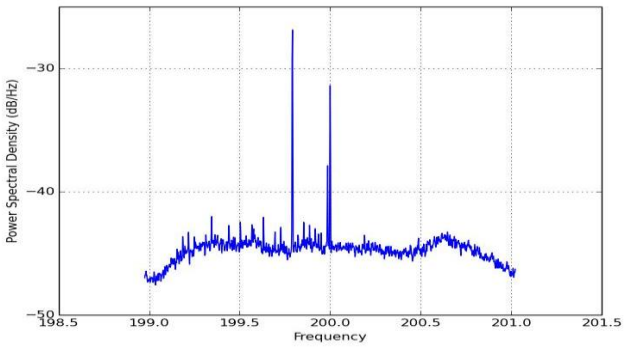


Figure 13: PSD of signal at 200 MH

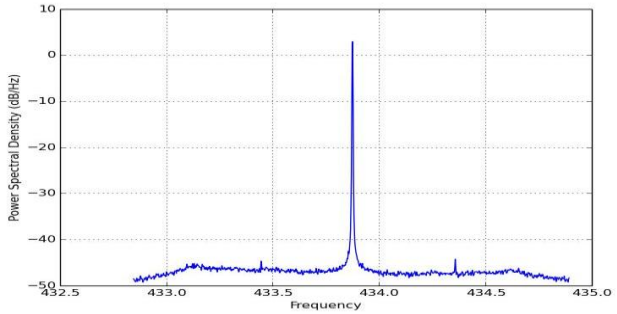


Figure 14: PSD of signal a434MHz

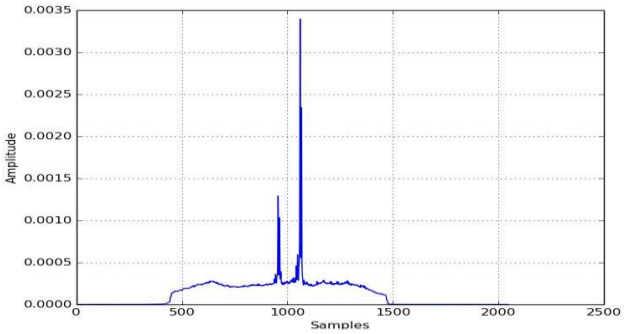


Figure 15: Cross-correlation plot at 200 Mhz

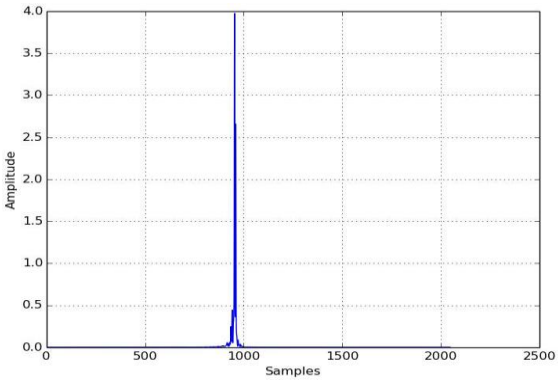


Figure 16: Cross-correlation plot at 434 MHz

The important conclusion to be drawn from the last two figures is that in the second plot, the maximum value is of a considerably higher value than that in the first plot. From this we can conclude that the second signal at 434 Mhz resembles the sample ASK signal more closely than the signal at 200 MHz.

B. Signal Decoding:

Now that the frequencies at which the ASK signals are present are found, the receiver is tuned to one of these frequencies for decoding the signal. The following are the real time snapshots of the received signal:

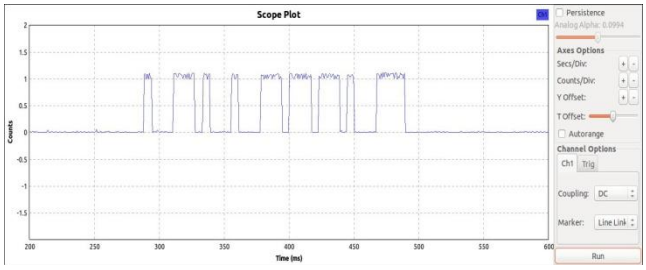


Figure 17: Received signal at 434 MHz

The signal shown above is less noisy and hence can be decoded easily but to also facilitate decoding of very noisy signals, the received signal is first passed through a 40-point averaging filter which smoothen the signal.

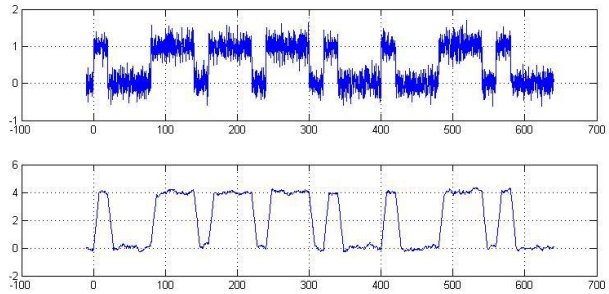


Figure 18: 40-point Averaging Filter: Input and Output

After the received signal is smoothened using the averaging filter, a pure digital signal is obtained by passing it through a threshold block. In this zeros, 1s and 0s are identified by computing the duty cycles of pulses and comparing them with a threshold. The threshold used in this paper is 0.5 because a 0 is a pulse with 25% duty cycle and a 1 is a pulse with 75% duty cycle. The ASCII characters are then determined from the received ones and zeros. The following is the final decoded message from the transmitter in real time:

Figure 19: Decoded Signal

It can be seen that the transmission is error free but it is very slow and hence there is room for improvement.

VI. CONCLUSION AND FUTURE SCOPE

The results from the previous section show that, although not entirely performance efficient, the system proposed in this paper can be of much use with some hardware and software optimizations. It is also clear that the primary objective of the paper, which is to demonstrate the capabilities of an SDR receiver, has been achieved. This also draws attention to the immense potential of a software defined radio. More modulation and encoding techniques can be incorporated into this paper and this will ultimately lead to an intelligent radio. A radio which can detect, demodulate and if possible decrypt any type of signal. In the future, if all portable electronic devices like a laptop computer, mobile phones, etc. are equipped with an in-built

SDR, then communication between devices can be made easy. If a cognitive radio engine is also added to devices, then the communication between devices becomes both intelligent and power efficient and eliminates any spectrum hassle.

The following improvements can be made to the paper to make it better:

a. S/N Maximization using Matched Filter.
b. Bandwidth determination
c. Adaptive decoding
d. Speed of scanning
e. Support to other modulation techniques (FSK, PSK, Spread Spectrum)

Matched filter greatly increases the signal to noise ratio but is useful only when one has knowledge about the kind of signal being received. Currently, the detection algorithm only determines the single frequency at which the signal is present. It would be much more useful if it could also determine the bandwidth of the received signal. For example, when the scan is run for FM signals, the frequency at which the signal is present and also its bandwidth must be found.

VII. REFERENCES

[1] Theodore S. Rappaport, Theodore Rappaport, 2002 Wireless Communications: Principles and Practice (2nd Edition). Prentice Hall PTR, ISBN: 0130422320

[2] Donno D, Ricciato F, Catarinucci L: Challenge: towards distributed RFID sensing with software-defined radio. In *MobiCom '10. Proceedings of the Sixteenth Annual International Conference in Mobile Computing and Networking, Chicago, Sept 2010*. ACM, New York; 2010:97-104.

[3] Software Defined Radio: Architectures, Systems and Functions (Markus Dillinger, Kambiz Madani, Nancy Alonistioti) Page xxxiii (Wiley & Sons, 2003, ISBN 0-470-85164-3).

[4] *Kennedy, G.; Davis, B. (1992). Electronic Communication Systems (4th ed.). McGraw-Hill International*. ISBN 0-07-112672-4., p 509

[5] FSK: Signals and Demodulation by B. Watson

[6] F. K. Jondral, "Parametrization—a technique for SDR implementation," in Software Defined Radio—Enabling Technologies, W. Tuttlebee, Ed., pp. 232–256, John Wiley & Sons, London, UK, 2002.

[7] A. Wiesler and F. K. Jondral, "A software radio for second-and third-generation mobile systems," IEEE Trans. Veh. Technol., vol. 51, no. 4, pp. 738–748, 2002.

[8] P. Rykaczewski, D. Pienkowski, R. Circa, and B. Steinke, "Signal path optimization in software defined radio systems," IEEE Trans. Microwave Theory Tech., vol. 53, no. 3, pp. 1056– 1064, 2005.

[9] J. Mitola III and Z. Zvonar, Eds., Software Radio Technologies: Selected Readings, John Wiley & Sons, New York, NY, USA, 2000.

[10] J. Mitola III and W. Tuttlebee, Eds., Software Defined Radio: Origins, Drivers and International Perspectives, John Wiley & Sons, Chichester, UK, 2002.

[11] W. Tuttlebee, Ed., Software Defined Radio: Enabling Technologies, John Wiley & Sons, Chichester, UK, 2002.

[12] M. Dillinger, K. Madani, and N. Alonistioti, Eds., Software De- fined Radio: Architectures, Systems and Functions, John Wiley & Sons, Chichester, UK, 2003.

[13] J. Reed, Software Radio—a Modern Approach to Radio Engineering, Prentice-Hall, Upper Saddle River, NJ, USA, 2002.