Development of an 1-Seg ISDB-T Receiver Using Low-cost SDR Hardware

Leonardo Muttoni and Antônio C. Veiga

Abstract— **This paper presents the development of an one-segment (1-seg) digital TV receiver for the Integrated Services Digital Broadcasting - Terrestrial (ISDB-T) standard, based on Software Defined Radio (SDR). The proposed receiver works in real-time and requires a computer and a low-cost device known as RTL-SDR, for the RF front-end, mixer and analog to digital conversion stages. The rest of signal processing were all done in software, which provides an important framework for study of the ISDB-T standard in practice.**

Keywords— **ISDB-T, SDR, OFDM, RTL-SDR.**

I. Introduction

The digital television system Integrated Services Digital Broadcasting - Terrestrial (ISDB-T) was adopted in Brazil, by almost all South-America, in some Central-America and Asian countries.

The knowledge appropriation of this technology by the academic and scientific community in telecommunications engineering field in these countries is critical, especially in universities outside the traditional technological centers, where financial resources are tight.

Software Defined Radio (SDR), with its associated flexibility, redefined how we can research and teach radio systems. Despite the growing number of SDR kits available in the market, the mass use of this technology for a telecommunications class is still limited, mainly because of its cost.

In this paper we describe the development of a complete and functional SDR receiver software for the ISDB-T 1-seg. The receiver works in real-time and requires a personal computer and an ultra low-cost (about 8 dollars) USB dongle connected to an antenna. This device is known as RTL-SDR [1] and it is lowest cost SDR receiver hardware available so far. Despite the modest features of this SDR hardware, it is fully capable of performing this task.

The proposed receiver can be used to study the ISDB-T system in practice and its parts in detail (to the level of an RF sample, bit or byte), some of which are common to other modern communication standards such as WiFi, ADSL and Powerline Communications.

The following sections of this paper gives an overview of the ISDB-T system and its transmission. The SDR receiver description follows and finally results are presented.

II. The ISDB-T system

ISDB-T standard is based on *Band Segmented Transmission - Orthogonal Frequency Division Multiplexing* (BST-OFDM).

Leonardo Muttoni and Antônio C. Veiga¸ Electrical Engineering School, Federal University of Uberlândia, Uberlândia-MG, Brazil, E-mails: muttoni@ufu.br, acpveiga@ufu.br.

This technology divides the useful TV bandwidth (5.57 MHz in Brazil) into 13 segments of 428.571 KHz. Each segment contains 108, 216 or 432 OFDM subcarriers, depending on the transmission mode chosen.

Segments can be clustered in up to 3 groups known as hierarchical layers A, B and C. Each layer can transmit distinct signals with different modulation and encoding configurations, depending on the type of service.

The most commonly used configuration by broadcasters contains two hierarchical layers: Layer A is composed by only the central segment (number 0) which transmits a low bit rate signal to the portable receivers (i.e. cell phones) using robust modulation and coding. Layer B is composed by the remaining 12 segments which transmits a high definition signal (with a much higher bit rate) to the fixed TV receivers.This arrangement is illustrated in Figure 1.

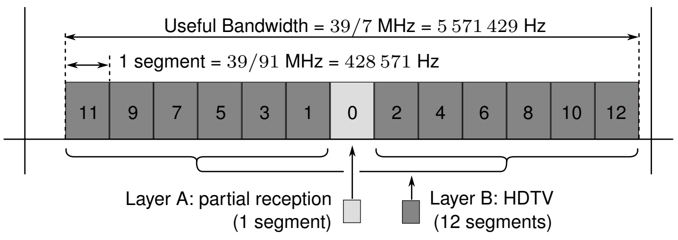
Useful Bandwidth = 39/7 MHz = 5 571 429 Hz
1 segment = 39/91 MHz = 428 571 Hz
11
9
7
5
3
1
0
2
4
6
8
10
12
Layer A: partial reception (1 segment)
Layer B: HDTV (12 segments)

Fig. 1. Segment arrangement example for two hierarchical layers

In the cases where there is a hierarchical layer composed only by the central (zero) segment, this layer is known as the *partial reception* layer, as it allows portable receivers to access digital signal by demodulating and decoding only the spectrum central segment. These receivers are called *1-seg* or *one-seg* while fixed receivers capable of demodulating all 13 segments are known as *full-seg*.

The ISDB-T transmission system is standardized by the [2] and [3]. The radiofrequency signal is generated from several signal processing steps, which starts with an input of up to three MPEG-2 Transport Streams (MPEG-2 TS), according to Figure 2. The main transmission blocks are explained in the remainder of this section.

Each MPEG-2 TS input stream contains data packets of 188 bytes, with compressed audio and video signals, subtitles, electronic program guide, and more. Each stream is intended for a distinct configured hierarchical layer.

The MPEG-2 TS streams are multiplexed and then delivered to the external Reed-Solomon encoder, where 16 bytes of parity are generated for each 188-byte MPEG-2 TS packet, composing a 204-byte Transport Stream Packet (TSP). At the

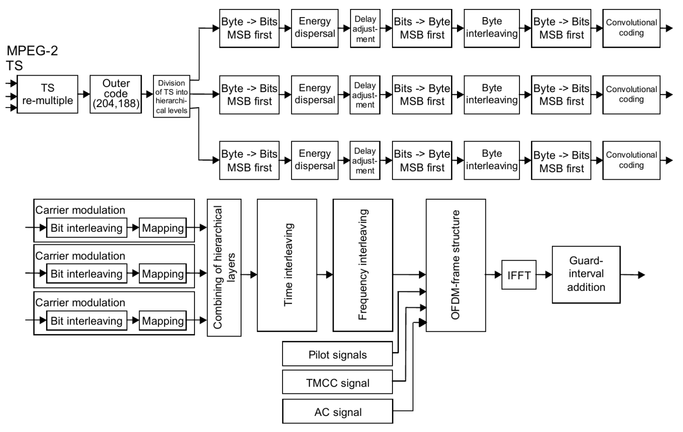

Fig. 2. Block diagram for the ISDB-T transmitter [2]

receiver, this code is capable of correcting up to 8 corrupted bytes within a TSP.

TSPs are demultiplexed into their configured hierarchical layers, and follow an independent processing flow: (1) the energy dispersal performs the bit-by-bit *exclusive OR* logical operation of the input signal with a Pseudo-Random Binary Sequence (PRBS) in order to prevent long sequences of identical bits thus making bit values 0 and 1 equally likely; (2) the convolutional byte interleaver spreads the adjacent bytes in order to mitigate burst-type errors at reception; (3) the bits are delivered to the internal convolutional encoder of constraint length 7 and basic rate of 1/2 (configurable for 2/3, 3/4, 5/6 or 7/8 by puncturing its outputs); (4) the bits of each layer are then mapped (modulated) independently in DQPSK, QPSK, 16QAM or 64QAM, generating complex vectors (constellation points).

The complex vectors of each layer are then combined and follow a single stream which passes through the time and frequency interleavers, generating data symbols for OFDM multiplexing. Both interleavers are of the convolutional type and aim to tackle the effects of signal fading.

The data, pilot, Transmission and Multiplexing Configuration Control (TMCC) and Auxiliary Channel (AC) symbols are distributed over 108 (mode 1), 216 (mode 2) or 432 (mode 3) subcarriers per segment, forming one OFDM symbol. Each OFDM frame are formed by a sequence of 204 OFDM symbols. The OFDM signal waveform is then generated by applying Inverse Fast Fourier Transform (IFFT) to each OFDM symbol which are interpreted as a complex vector. Then a cyclic prefix of configurable length (1/32, 1/16, 1/8 or 1/4) is inserted in order to make the signal robust for the multipath fading effects.

The carrier is modulated by the ISDB-T signal, generating the $s(t)$ waveform. According to [2] and [3], this waveform is given by the equations 1 and 2, where k is the OFDM subcarrier index number; n is the symbol sequential number; K is the total number of OFDM subcarriers; T_s is the duration of one symbol; T_g is the duration of the guard-interval; T_u is the symbol useful duration; f_c is the RF central frequency; K_c is the central subcarrier index; and $c(n,k)$ is the complex vector for the symbol n and subcarrier index k.

$$s(t) = Re\left\{ e^{j2\pi f_c t} \sum_{n=0}^{\infty} \sum_{k=0}^{K-1} c(n,k)\Psi(n,k,t) \right\} \tag{1}$$

$$\Psi(n,k,t) = \begin{cases} e^{j2\pi \frac{k-K_c}{T_u}(t-T_g-nT_s)}, & nT_s \leq t < (n+1)T_s \\ 0, & t < nT_s,\ t \geq (n+1)T_s \end{cases} \tag{2}$$

III. Receiver implementation

The developed SDR receiver architecture is shown in Figure 3. All blocks (except the first one, RTL-SDR USB dongle) were realized in software using the C language, running on Linux operating system. In the following subsections, the operation of the main blocks are explained.

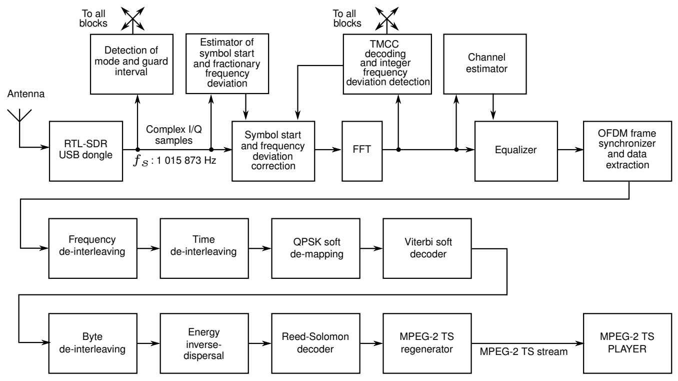

Fig. 3. Block diagram for the developed ISDB-T 1-seg SDR receiver

A. Hardware

The hardware chosen for the SDR receiver was the RTL-SDR [1]. This device is a low-cost (about USD 8.00) USB receiver that uses a Realtek RTL2832U [4] chip and a Rafael Micro R820T [5] tuner chip. It contains basically an RF front-end with LNA, a mixer (for down conversion) and an Analog to Digital Converter (ADC). The digital samples are read from this USB device thorough the *librtlsdr* [6] library and fed to the rest of the signal processing chain.

The main features of RTL-SDR are: (1) Quadrature sampling up to 2.4 Million Samples Per Second (MSPS), with bandwidth of 2.4 MHz; (2) 8-bit resolution; (3) Tuning range: 24 MHz – 1766 MHz. Figure 4 shows the internal block diagram of the RTL-SDR — specifically the version equipped with the R820T or R820T2 tuner chip.

A peculiarity about the RTL-SDR device is that it was not originally designed and marketed as an SDR receiver, but as a USB adapter for DVB-T reception on the computer. Through reverse engineering [7] [8], it was discovered how to transfer the raw samples from the RTL2832U internal A/D converter to the USB, allowing its use as a generic SDR receiver. The low cost can be explained by the high volume of production of this mass-market device (considering its original function).

It is important to note that the RTL2832U's internal DVB-T decoder has a fixed function and can not be used in any stage of ISDB-T signal decoding, even if there are matching blocks in different standards. In this development (such as in any other SDR that uses the aforementioned device), the internal DVB-T block is completely disabled, and the in-phase and quadrature samples of the tuned signal are sent directly to the host computer via USB for processing.

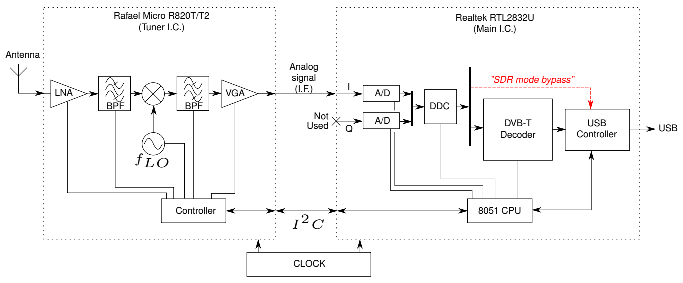

Fig. 4. RTL-SDR internal diagram

B. OFDM synchronization

The joint maximum likelihood estimators for the OFDM symbol-time $\hat{\theta}$ and for the fractional carrier-frequency offset $\hat{\varepsilon}$ are obtained by correlating two samples separated by the symbol length, through the method described in [9], based on redundant information contained within the cyclic prefix. The said method are improved by averaging the results over a number of symbols. This works when the assumption of slowly varying parameters θ and ε are true.

The estimator $\hat{\varepsilon}$ is called *fractional* because it is able to estimate only fraction of the frequency spacing between adjacent subcarriers ($|\hat{\varepsilon}| \leq 0.5$). The integer carrier-frequency offset δ is obtained after the Fast Fourier Transform (FFT) by searching for a TMCC subcarrier in a range of subcarriers, and it is measured in units of FFT bin size.

The detection of transmission mode and guard interval duration are based on a search that is done only in the first moment of the receiver operation. Each possible combination for $\{mode, guard\ interval\}$ are tried, and the one which has the highest peak to average ratio for the $\Lambda(\theta)$ function (notation according with [9]) are selected.

The $\hat{\theta}$, $\hat{\varepsilon}$ and δ are passed to the *Symbol start and frequency deviation correction* block, that corrects these deviations in software through vector shift by $\hat{\theta}$ samples and complex multiplication of each sample by $e^{-j2\pi b(\hat{\varepsilon}+\delta)t_s n}$ (t_s is the sample period, n is the sample number, b is the FFT's bin size in Hz).

C. TMCC decoding

Some of the OFDM subcarriers continuously transmits the TMCC signal. These subcarriers are modulated though DBPSK and conveys information about the hierarchical layers configuration, coding, interleaving and modulation parameters, that are essential information for the receiver operation.

One TMCC frame comprise 204 bits, of which 102 bits that carries important informations are error-correction coded by a (184, 102) shortened *difference-set cyclic code* (273,191) that generates 82 parity bits.

The implemented decoder is based on variable threshold decoding [10], and can correct up to 11 errors [11] in a block.

D. Equalization

The channel estimation and equalization are conducted entirely in frequency domain, after the FFT, with the aid of the scattered pilots that are inserted by the transmitter into the OFDM frames through the BPSK modulation of a PRBS signal. Its constellation contains the point $(-4/3, 0)$ for bit 1 and $(+4/3, 0)$ for bit 0.

The estimate of the channel response is obtained by comparing the pilot received with the known transmitted one. For the subcarriers than do not contain pilot symbols, the estimate are obtained through linear interpolation.

From the estimation of the channel response, the equalization is done by the complex multiplication of the input signal with the inverse of the channel response.

E. Demodulation

The ISDB-T standard specifies that the QPSK modulation maps a pair of bits $\mathbf{V} = (b_0, b_1)$ in a complex symbol $s = a+bi$ in the constellation as follows: $(0,0) \mapsto 1/\sqrt{2}+1/\sqrt{2}i$; $(0,1) \mapsto 1/\sqrt{2} - 1/\sqrt{2}i$; $(1,0) \mapsto -1/\sqrt{2} + 1/\sqrt{2}i$ and $(1,1) \mapsto -1/\sqrt{2} - 1/\sqrt{2}i$.

Demodulation does the reverse mapping, of the received complex symbol $\hat{s} \mapsto \hat{\mathbf{V}}$. When demodulation produces binary values ($\hat{b_0}, \hat{b_1} \in [0, 1]$), it is called *hard decision*. Another option is *soft decision* demodulation, when output is composed of integer values in a larger range, quantized by n bits. In this case $\hat{b_0}, \hat{b_1} \in [0, 2^{n-1}]$, and $\hat{b_0}, \hat{b_1}$ carry the *reliability* of the received bit. This extra information increases the error correction power of the Viterbi decoder [12, p. 533–535] [13, p. 580].

The developed demodulator allows the quantization in $n = 1$, 2 or 3 resolution bits, so its output vector $\hat{\mathbf{V}} = (\hat{b_0}, \hat{b_1})$, $\hat{b_0}, \hat{b_1} \in \mathbb{Z} \cap \{[0, 1],\ [0, 3]$ or $[0, 7]\}$, being the case $n = 1$ equivalent to *hard* demodulation. This configurable parameter n allows to evaluate, in practice, the error correction improvement for the following block.

F. Viterbi decoding

The internal code (convolutional) is decoded using the Viterbi algorithm, a maximum likelihood decoder [13, p. 616]. This block have a configurable number of quantization levels, matched to the demodulator.

At startup, a table is created in memory to speed up the calculation of *branch metrics*. Upon the reception of a input symbol, the first step of the decoder is to perform the depuncturing of the input stream, according to the rate of the code indicated by the TMCC. *Erasures* symbols, that does not change the *branch metric*, are generated at the same positions where puncturing was performed on the transmitter.

The *traceback length* is configurable and has been set to 128. The traceback algorithm operates with two-stage sliding windows: (1) the current window is run to converge to the best trellis path, without the emission of output; (2) in a previous window, the decoding phase is started from the best metric state obtained in the last step, and the decoded bits are emitted.

The performance of this algorithm proved to be equivalent to that of the *vitdec()* function of Matlab. The results for different input quantizations can be seen in the figure 5.

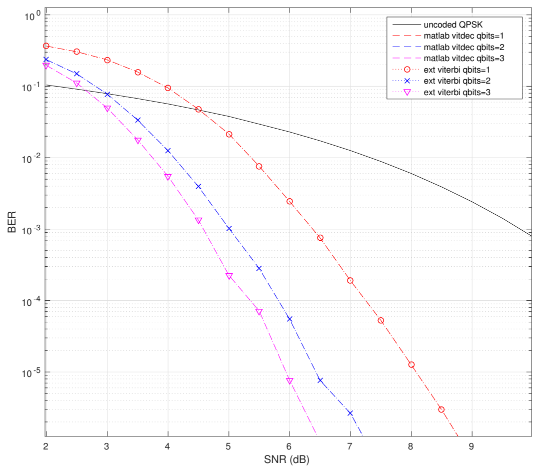

Fig. 5. Performance of the developed Viterbi decoder (called *ext viterbi* in the plot, dotted curves with markers) for different number of quantization bits (qbits). Matlab *vitdec()* reference curves are plotted with dashed lines. Code rate was $2/3$.

G. Reed-Solomon decoding

The Reed-Solomon [14] decoder is the most complex part of the receiver, in terms of the number of lines of code and mathematics knowledge needed. Its realization requires a wide range of auxiliary functions to work with *Galois Field* arithmetic $GF(256)$, such as sum, multiplication, division, power, and logarithm. Above this framework, there are also other functions that work with polynomials over the $GF(256)$, such as: polynomial division, multiplication, roots calculation, evaluation and formal derivative.

The decoding, in short, consists of the steps: (1) the syndrome $S(x)$ is calculated from the received code; (2) the error locator polynomial $\Lambda(x)$ are obtained through the *Berlekamp-Massey* algorithm [15, p. 184] [16] [17, p. 187] [18, p. 209] [19, p. 258]; (3) the roots of $\Lambda(x)$ are calculated (which are the inverses of the errors locators X_i^{-1}); (4) The error-evaluating polynomial $\Omega(x)$ is calculated, according to the equation 3; (5) $\Lambda'(x)$, the formal derivative of $\Lambda(x)$, is calculated; (6) the magnitude of the errors Y_i are obtained through the *Forney* method [20] [17, p. 196] [18, p. 247] [19, p. 262], according to equation 4; and (7) the received code is corrected at the positions given by X_i, by the sum of Y_i with the corrupted byte.

$$\Omega(x) = S(x)\Lambda(x) \quad (mod \;\; x^{2t}) \tag{3}$$

$$Y_i = -\frac{X_i^{1-b}\Omega(X_i^{-1})}{\Lambda'(X_i^{-1})} \tag{4}$$

When the decoder fails (an uncorrectable block are received), it sets the *transport error indicator* bit in the MPEG-2 TS packet, according with the standard. This flags the media player that the packet is invalid.

The decoder keeps a log for the error-correction statistics, such as the number of uncorrectable blocks, the number of blocks and bits processed and corrected. This information can be accessed and seen in real time, helping to assess the receiver performance.

H. MPEG-2 TS regeneration

According to [21, p. 7.29–7.30], the partial reception layer should not transmit the Program Association Table (PAT) due to bandwidth limitation. This makes this MPEG-2 TS stream not fully compliant with the MPEG-2 standard, since the latter establishes that the transmission of PAT is mandatory [22, p. 8, 110].

The lack of PAT causes reproduction incompatibility. The media players *VLC* and *Mplayer* where found to follow the MPEG-2 standard strictly, refusing to play the stream. A third player tested, *ffplay*, was found being more relaxed, playing the stream well most of the time.

To improve this situation, a block called *MPEG-2 TS regenerator* was implemented. This block creates the PAT table from data obtained from the Program Map Table (PMT) and inserts it periodically in the MPEG-2 TS stream. This makes the stream standard-compliant and allows media players that previously refused to play the stream be able to reproduce it correctly.

IV. Results

Figure 6 shows the receiver in operation, tuned to a public television channel.

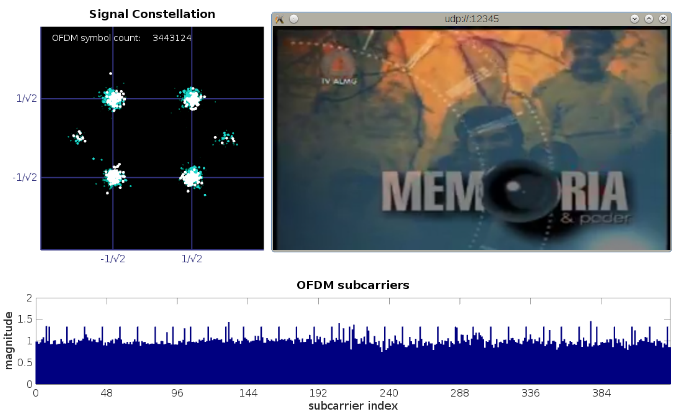

Fig. 6. Receiver in operation. Top right picture shows the media player (ffplay) window; Two other pictures shows a companion tool that displays in real time the constellation and the magnitude plot for the 432 FFT bins (mode 3), considering the signal after the equalizer.

Some processing blocks are important for performance of the receiver: (1) frequency and symbol synchronizer; (2) channel estimator and equalizer; (3) Viterbi decoder and (4) Reed-Solomon decoder.

The frequency synchronization is critical for OFDM systems, where errors of few tens of Hz causes severe intercarrier interference (ICI). Added to this, the RTL-SDR, being low cost, comes with a crystal oscillator clock with coarse frequency stability, that drifts somewhat with temperature. Despite this, the frequency and symbol-start estimation and correction blocks proved to be reliable enough, based on the inspection of the signal's constellation before the equalizer, and on receiver testing subjected to temperature changes.

Although simple, the linear interpolation used in the equalizer worked well, even with internal antenna where significant fading can be seen in frequency domain before the equalizer. This could be assessed qualitatively through the signal's constellation and spectrum inspection, after the equalizer. Nevertheless, more advanced equalization techniques can be employed for even better performance.

As shown in Figure 5, the performance of the Viterbi decoder were virtually identical with the reference implementation (Matlab).

In extensive simulations using random error patterns, the Reed-Solomon decoder behaved as expected, correcting errors of up to 8 bytes per block.

Regarding the speed margin for real-time processing, on a laptop with Intel Core i7 3635QM 3.2 GHz processor (4 cores, 8 threads), the receiver uses about 20% of the time from one of the 8 processor threads to decode the 1-seg signal in real time. The table I shows the detail of CPU usage by each block of the receiver. The memory usage was 7 704 KB.

The dominance of CPU usage for the Viterbi decode can be explained mainly by the computation of 128 metrics $(2^{k-1} \cdot n_o)$ for each input bit [1]. The Viterbi input bit-rate R (including the de-punctured bits) for the one segment receiver is about 950 kbits/s, therefore, the Viterbi decoder needs to compute $128 \cdot R \approx 122$ milion metrics each second. There are room for speed improvement of this algorithm, such as vectorization by use of processor's Single Instruction Multiple Data (SIMD) instructions.

TABLE I
RELATIVE CPU USAGE FOR EACH BLOCK

Block	relative % of CPU time
Viterbi decoder	82.73%
Frequency corrector	3.46%
Demodulator	2.72%
De-interleavers	2.07%
Equalizer	1.39%
Reed-Solomon decoder	0.97%
FFT	0.94%
Energy inverse-dispersor	0.23%
MPEG-2 TS regenerator	0.06%
TMCC decoder	0.05%
Other functions	5.37%
TOTAL	100.00%

V. CONCLUSIONS

The overall outline of the receiver was presented and results were shown, proving that inexpensive SDR hardware with modest attributes like RTL-SDR can be used to develop this kind of application.

A low cost SDR system can be a key factor in places with tight financial resources, as it can enable learning, teaching and research of a complex digital receiver. In this sense, each student in a class could have access to a SDR hardware, so that they can use it even at home for research and learning.

ACKNOWLEDGMENTS

Thanks for Electrical Engineering School of the Federal University of Uberlândia by the support for this work.

[1] k: constraint length of the code, equal to 7 for The ISDB-T; n_o: number of convolutional encoder outputs, equal to 2 for ISDB-T.

REFERENCES

[1] RTL-SDR.COM. About RTL-SDR. [Online]. Available: http://www.rtl-sdr.com/about-rtl-sdr

[2] *Transmission System for Digital Terrestrial Television Broadcasting*, ARIB Std. B31, mar. 2014.

[3] *Televisão digital terrestre - Sistema de transmissão*, ABNT Std. NBR 15 601, nov. 2007.

[4] REALTEK. RTL2832U. [Online]. Available: http://www.realtek.com.tw/products/productsView.aspx?Langid=1&PFid=35&Level=4&Conn=3&ProdID=257

[5] "R820T datasheet," DatasheetsPDF.com, Rafael Microelectronics. [Online]. Available: http://www.datasheetspdf.com/datasheet/R820T.html

[6] OSMOCOM.ORG. (2014) RTL-SDR. [Online]. Available: http://sdr.osmocom.org/trac/wiki/rtl-sdr

[7] S. Cass, "A $40 software-defined radio," *IEEE Spectrum*, vol. 50, no. 7, pp. 22–23, jul 2013. [Online]. Available: http://dx.doi.org/10.1109/MSPEC.2013.6545114

[8] RTLSDR.ORG. History and discovery of rtlsdr. [Online]. Available: http://www.rtlsdr.org/#history_and_discovery_of_rtlsdr

[9] J. Van de Beek, M. Sandell, and P. Börjesson, "ML estimation of time and frequency offset in OFDM systems," *IEEE Transactions on Signal Processing*, vol. 45, no. 7, Jul. 1997.

[10] R. Townsend and E. Weldon, "Self-orthogonal quasi-cyclic codes," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 183–195, apr 1967. [Online]. Available: https://doi.org/10.1109%2Ftit.1967.1053974

[11] O. Yamada, "Development of an error-correction method for data packet multiplexed with TV signals," *IEEE Transactions on Communications*, vol. 35, no. 1, pp. 21–31, jan 1987. [Online]. Available: https://doi.org/10.1109%2Ftcom.1987.1096669

[12] J. Proakis and M. Salehi, *Digital communications*, 5th ed. Boston: McGraw-Hill, 2008.

[13] S. Haykin, *Digital communication systems*. Hoboken, NJ: J. Wiley & Sons, 2014.

[14] I. S. Reed and G. Solomon, "Polynomial codes over certain finite fields," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 2, pp. 300–304, 1960.

[15] E. R. Berlekamp, *Algebraic Coding Theory*. World Scientific Publishing Co, 2015, revised Edition.

[16] J. Massey, "Shift-register synthesis and BCH decoding," *IEEE Transactions on Information Theory*, vol. 15, no. 1, pp. 122–127, jan 1969. [Online]. Available: https://doi.org/10.1109%2Ftit.1969.1054260

[17] R. E. Blahut, *Algebraic Codes for Data Transmission*, 2nd ed. Cambridge University Press, 2003.

[18] S. Lin and D. J. Costello, *Error Control Coding*, 2nd ed. Pearson, 2004.

[19] T. K. Moon, *Error Correction Coding: Mathematical Methods and Algorithms*. Wiley-Interscience, 2005.

[20] G. Forney, "On decoding BCH codes," *IEEE Transactions on Information Theory*, vol. 11, no. 4, pp. 549–557, oct 1965. [Online]. Available: https://doi.org/10.1109%2Ftit.1965.1053825

[21] "Provisions for carrier operations," ARIB, Tech. Rep. B14, 2015, fascicle 5, Vol. 7, Revision 6.0-E1.

[22] *Information technology - Generic coding of moving pictures and associated audio information: Systems*, ISO/IEC Std. 13 818-1:2000 (E), 2000.