DVB-T CHANNEL CODING IMPLEMENTATION IN MATLAB

Ondřej Hüttl, Tomáš Kratochvíl

Department of Radio Electronics, Brno University of Technology

Purkyňova 118, 612 00 BRNO

Abstract

The paper deals with the Matlab implementation and simulation of channel coding and modulation of DVB-T (Digital Video Broadcasting – Terrestrial). Transport stream multiplex adaptation and randomization for energy dispersal, outer coding and interleaving, inner coding and interleaving, constellation and mapping blocks of encoder and decoder are implemented. Channel encoder and decoder follow European Standard ETSI EN 300 744 for digital terrestrial television and provide utilization of different convolutional encoder code rates, two inner modulation modes schemes for 2k and 8k OFDM (Orthogonal Frequency Division Multiplexing). The modulation OFDM was not implemented.

1 Introduction

The DVB-T (Digital Video Broadcasting – Terrestrial) is ETSI EN 300 744 standard [1] of European digital television for the terrestrial transmission to fixed, portable and mobile receivers. The DVB-T Standard specifies the framing structure, channel coding and modulation for digital terrestrial broadcasting. The system is fully compatible with MPEG-2 coded TV signals ISO/IEC 13818 [2] and has several similarities of channel coding with DVB-S (Digital Video Broadcasting – Satellite) and DVB-C (Digital Video Broadcasting – Cable) standards.

2 Channel Coding and Decoding structure

The system is composed of functional blocks performing the adaptation of the baseband TV signals from the output of the MPEG-2 transport multiplexer to the terrestrial channel characteristics. The system input data stream is organized in fixed length 188 bytes MPEG-2 packets.

The following processes are applied to the data stream [1]:

transport multiplex adaptation and randomization for energy dispersal;

outer coding (i.e. Reed-Solomon code);

outer interleaving (i.e. convolutional interleaving);

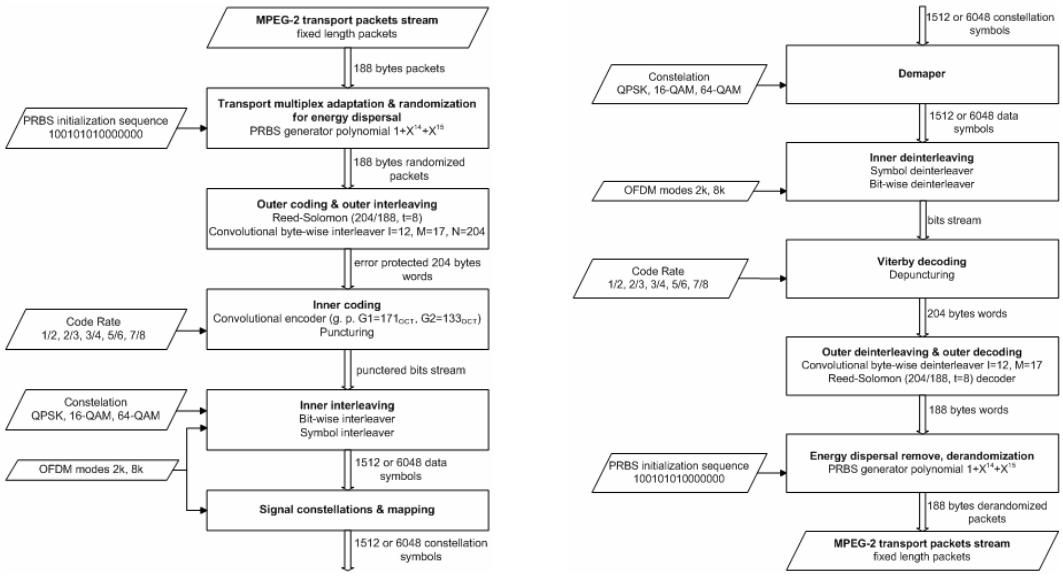
inner coding (i.e. punctured convolutional code);

inner interleaving (either native or in-depth);

mapping and modulation;

Orthogonal Frequency Division Multiplexing (OFDM) transmission.

These processes (except for OFDM) are performed in Channel coder and Channel decoder with the configurations depicted in Figure 1.



a) FEC encoder b) FEC decoder

Figure 1: Flow-chart of channel coding and decoding in DVB-T (implemented in Matlab)

Transport multiplex adaptation and randomization, outer coding and outer interleaving are common with the Satellite DVB-S baseline specification ETSI EN 300 421 [3] and Cable DVB-C baseline specifications ETSI EN 300 429 [4] and the inner coding is common with the Satellite baseline specification. Detailed description of DVB-T channel coder blocks can be found in [1].

3 Channel Coding implementation in Matlab

Following section describes possible implementation of channel coding/modulation and decoding/demodulation for DVB-T in accordance with European Standard ETSI EN 300 744 [1] for digital terrestrial television.

3.1 Transport multiplex adaptation and randomization for energy dispersal

The System input stream is composed by MPEG-2 transport multiplex (MUX) packets with total length 188 bytes. The data of the input MPEG-2 multiplex are randomized to ensure adequate binary transitions. First byte of packet is synchronization word byte (i.e. 47HEX). Synchronization words are not randomized. The polynomial for the Pseudo Random Binary Sequence (PRBS) generator shall is $1 + X^{14} + X^{15}$.

Initialization sequence "100101010000000" is loaded to the PRBS registers at the beginning of every eight transport packets – once per frame composed by 8 packets. The first synchronization byte word of every frame is bit wise inverted to provide an initialization for descrambler. Synchronization byte words are not randomized.

Following code performs randomization or de-randomization of one frame:

```
enable = [zeros(1,8),ones(1,1496)]; %randomize enable signal for one packet
enable = [enable, enable, enable, enable, enable, enable, enable, enable]; %frame

    for b = 1:12032 %188 bytes * 8 bits = 1504 bits, 1504 bits * 8 packets = 12032
         if b < 9    %PRBS generator starts after 1st sync word
             randomized(b) = inp2rand(b);
         elseif b > 8
         PRBSout = xor(PRBS(1,14),PRBS(1,15));
         PRBS = circshift(PRBS, [1,1]);
         PRBS(1,1) = PRBSout;
              %sync words are not randomized due to enable signal
         randomized(b) = xor(inp2rand(b),(and(enable(b),PRBSout)));
         end
    end
```

3.2 Outer coding and outer interleaving

The outer coding and interleaving is performed on the randomized 188 bytes transport packets structure. Reed-Solomon RS (204, 188, t = 8) shortened code is derived from the original systematic RS (255, 239, t = 8) and may be implemented by adding 51 bytes, all set to zero, before the information bytes at the input of an RS (255, 239, t = 8) encoder. These null bytes shall be discarded after the RS encoding procedure, leading to a RS code word of N = 204 bytes.

Following code performs Reed-Solomon RS (204, 188, t = 8) encode:

```
zeros51 = zeros(8,51);          % null bytes filling to 255 bytes words
data239B = [zeros51, data188B]; % 51 null bytes + 188 data bytes

%Reed-Solomon encoding
m = 8;            % Number of bits in each symbol
n = 255; k = 239; % Codeword length, message length - RS(255,239)
msg = gf(data239B,m);   % Represent data using a Galois array

RS255_239 = rsenc(msg,n,k);         % Reed-Solomon(255,239) encoding
RS255_239 = double(RS255_239.x);    % Conversion from Galois array to double
RS204_188 = RS255_239(:,52:255);    % Reed-Solomon(204,188) shortening
```

Convolutional byte-wise interleaving with depth I = 12 is applied to the error protected 204 bytes packets. The interleaver is composed of I = 12 branches, cyclically connected to the input byte-stream by the input switch. Each branch j shall be a First-In, First-Out (FIFO) shift register, with depth $j * M$ cells where $M = 17 = N/I$, N = 204.

Following code performs byte-wise interleaving:

```
% Outer interleaving
M = 17;     % M = 204/12 = 17

% Set delays of 12 shift registers
delay = [0 M (2*M) (3*M) (4*M) (5*M) (6*M) (7*M) (8*M) (9*M) (10*M) (11*M)];

intDelay = length(delay)*max(delay);  % Interleaver delay
intFill = zeros(1,intDelay); % zeros fill for delay compensation
interleaverIn = [RSout,intFill]; % RSout represents RS protected packets

outInterleaved = muxintrlv(interleaverIn,delay); % Outer interleaving
```

3.3 Inner coding

The system allows a range of punctured convolutional codes, based on a mother convolutional code of rate 1/2 with 64 states. The generator polynomials of the mother code are G_1 = 171OCT for X output and G_2 = 133OCT for Y output.

Following code performs mother convolutional encode with rate 1/2:

```
% Convolutional encoding
trel = poly2trellis(7,[171 133]);   % Define trellis

encoded = convenc(msgBits,trel);    % Convolutional encode of msgBits
```

Selection of puncturing code rate allows selection of the most appropriate level of error correction for a given service or data rate. Available code rates CR are 1/2, 2/3, 3/4, 5/6, 7/8.

Following code presents puncturing with code rate CR = 2/3:

```
if CR == '2/3'
    missingPunct = 4 - mod(len,4);  % bits missing to divisibility by 4
    tcode = [encoded,zeros(1,missingPunct)]; % filling with zero bits
    len = length(tcode);            % new length

    reshaped = reshape(tcode,4,((len/4)));     % prepare for puncturing
    punctOut = reshaped;
    punctOut(3,:) = [];                        % remove of X2
    punctOut = reshape(punctOut,1,(len*3/4));  % X1 Y1 Y2
end
```

3.4 Inner interleaving

Inner interleaving block consists of bit-wise interleaving followed by symbol interleaving. Both the bit-wise interleaving and the symbol interleaving processes are block-based. Non-hierarchical mode is described only.

Bit-wise interleaving input is demultiplexed (mapped to output modulation symbols) into v sub-streams, where $v = 2$ for QPSK, $v = 4$ for 16-QAM, and $v = 6$ for 64-QAM. Each sub-stream is then interleaved in the interleaver with own interleaving sequence - permutation function. The bit interleaving block size is 126 bits and is the same for each interleaver. The block interleaving process is repeated exactly twelve times per OFDM symbol in the 2K mode and forty-eight times per symbol in the 8K mode.

Following code presents demultiplexing to 4 substreams for 16-QAM:

```
% correct data length prepare
r = (numel(input)/v);         % m - rows number of reshaped data matrix
a = (reshape(input,v,r))';    % v - columns number of reshaped data matrix,

                        % v - number of sub-streams
repetitions = ceil(r/126); % number of block interleaving repetitions
fill2b = zeros(1,(repetitions*126 - r)); % zeros to be filled at the end of
                                         % multiplexed streams

% demultiplexing for 16-QAM
if v == 4               % demultiplexed 4 data streams for 16QAM
    b0 = (a(:,1))';     % b – demultiplexed sub-streams
    b1 = (a(:,3))';
    b2 = (a(:,2))';
    b3 = (a(:,4))';

    b0 = [b0,fill2b];   % zero bits filling for divisibility by 126
    b1 = [b1,fill2b];
    b2 = [b2,fill2b];   % zero bits filling for divisibility by 126
    b3 = [b3,fill2b];
end
```

Following code performs bit-wise interleaving for 16-QAM:

```
% permutation functions
w = 0:125;

H0 = w;                 % permutation functions for I0:I5 interleavers
H1 = mod((w+63),126);
H2 = mod((w+105),126);
H3 = mod((w+42),126);


%bit-wise interleaving
Aout = [];      %interleaved output matrix preparation
```

```
for c = 1:repetitions       %cycles of 126 bits blocks interleaving
    low = c*126-125;
    high = c*126;

        B0 = b0(1,low:high);    %load of actual blocks
        B1 = b1(1,low:high);

    if  v == 4
        B2 = b2(1,low:high);
        B3 = b3(1,low:high);
    end

% interleaved outputs
    if v == 4
        a0(1,w+1) = B0(1,(H0+1));   % indexed by w+1 and H+1 because Matlab
        a1(1,w+1) = B1(1,(H1+1));   % indexing rules i<0
        a2(1,w+1) = B2(1,(H2+1));
        a3(1,w+1) = B3(1,(H3+1));
        A=[a0;a1;a2;a3];
        Aout = [Aout,A];
    end
end
```

Symbol interleaving is performed at bit-wise interleaved substreams. The purpose of the symbol interleaver is to map *v* bit words onto the 1 512 (in 2K mode) or 6 048 (in 8K mode) active carriers per OFDM symbol. The symbol interleaver acts on blocks of 1 512 (in 2K mode) or 6 048 (in 8K mode) data words. Details about permutation function of symbol interleaver can be found in [1].

Following code computes permutation function for 2K mode:

```
if    mode == '2k'
      K = 1512; % number of active data (sub)carriers (2 * 6) * 126 = 1512
      Nmax = K; % inner symbol interleaver block size
      Mmax = 2^11;      % IFFT length
      Nr = log2(Mmax);  % (Nr - 1) bits length of binary word Rii

      % permutation function inputs prepare
      Rii(1,:) = zeros(1,Nr-1);
      Rii(2,:) = zeros(1,Nr-1);
      Rii(3,:) = [zeros(1,Nr-2),1];

      for i = 4:Mmax
          Rii(i,2:(Nr-1)) = Rii((i-1),1:(Nr-2));
          Rii(i,1)        = xor(Rii((i-1),Nr-1),Rii((i-1),Nr-4));
      end
      Ri =  Rii(:,10), Rii(:,3), Rii(:,5), Rii(:,9), Rii(:,2), Rii(:,8),
            Rii(:,4), Rii(:,1), Rii(:,7), Rii(:,6)];
end


% permutation function H computation
q = 0;
for i = 0:(Mmax-1)
    for j = 0:(Nr-2)
        Rij(i+1,j+1) = Ri(i+1,j+1)*2^j;
    end

    Hq(1,q+1) = (mod(i,2))*2^(Nr-1)+sum(Rij(i+1,:)); % Permutation function

    if  Hq(1,q+1) < Nmax
        q = q+1;
    end
end
```

Following code performs symbol interleaving:

```
Yinp = Yin(:,(s*Nmax-(Nmax-1)):(s*Nmax));       % load of actual symbol data
for q = 1:Nmax
      if    rem(s, 2) == 0
            Yint(:,(Hq(q)+1)) = Yinp(:,q);      % interleaving of words for
                                                % even OFDM symbols
      else
            Yint(:,q) = Yinp(:,(Hq(q)+1));      % interleaving of words for
                                                % odd OFDM symbols
      end
end
```

3.5 Signal constellations and mapping

The system uses Orthogonal Frequency Division Multiplex (OFDM) transmission. All data carriers in one OFDM frame are modulated using QPSK, 16-QAM, 64-QAM, non-uniform 16-QAM or non-uniform 64-QAM constellations.

Following code present simple constellations mapping for QPSK:

```
if strcmp(Modulation, 'QPSK')
      for q = 1:Nmax;
            if Y(q,:)==     [0 0];
            mapped(q)=  +1+1j;
            elseif Y(q,:)== [0 1];
            mapped(q)=  +1-1j;
            elseif Y(q,:)== [1 0];
            mapped(q)=  -1+1j;
            elseif Y(q,:)== [1 1];
            mapped(q)=  -1-1j;
            end
      end
end
```

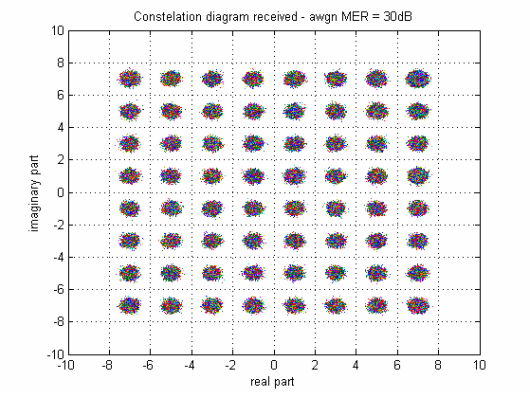
4 Examples of implemented channel coder and decoder functionality

DVB-T channel coder as described above and DVB-T channel decoder have been implemented in Matlab. Picture pout.tif was used as useful data load incoming to channel coder. Data of length 69840 bytes were processed and left coder in form of 64-QAM mapped symbols. AWGN (Additive white Gaussian noise) was added to mapped symbols to simulate *MER* (Modulation Error Rate). Code rate 1/2 and mode 8K were used for simulations. Operations inverse to coder were then performed in decoder with following results. Simulated error rates for different *MER* values are in the Table 1.

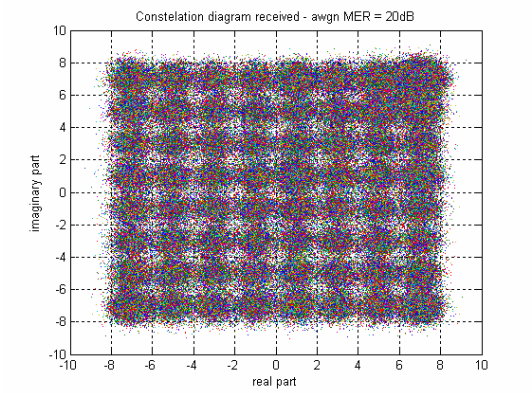
BER_0 in Table 1. represents bit errors after hard decision demapper, BER_1 is error rate after Inner deinterleaver, BER_2 is error rate after Vietrbi (Inner) decoder and BER_3 is error rate after Outer deinterleaving and outer (Reed-Solomon) decoder.

TABLE 1: SIMULATED BER OF DVB-T TRANSMISSION WITH AWGN

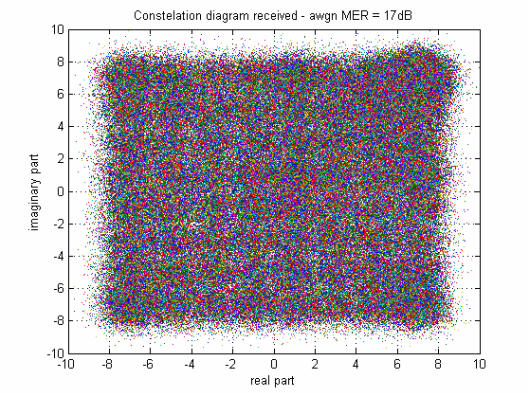
MER	BER_0	BER_1	BER_2	BER_3
30 dB	0	0	0	0
25 dB	$1.09\ 10^{-4}$	$1.13\ 10^{-4}$	0	0
20 dB	$1.25\ 10^{-2}$	$1.27\ 10^{-2}$	$2.45\ 10^{-5}$	0
17 dB	$4.31\ 10^{-2}$	$4.37\ 10^{-2}$	$1.50\ 10^{-3}$	0
16 dB	$5.67\ 10^{-2}$	$5.75\ 10^{-2}$	$6.20\ 10^{-3}$	$5.19\ 10^{-5}$
15 dB	$7.16\ 10^{-2}$	$7.27\ 10^{-2}$	$2.09\ 10^{-2}$	$2.05\ 10^{-2}$
14 dB	$8.74\ 10^{-2}$	$8.88\ 10^{-2}$	$5.70\ 10^{-2}$	$6.35\ 10^{-2}$



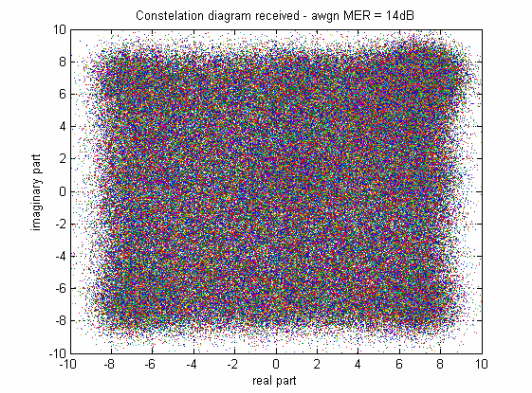
a) 64-QAM symbols with *MER* = 30dB



b) 64-QAM symbols with *MER* = 20dB



c) 64-QAM symbols with *MER* = 17dB

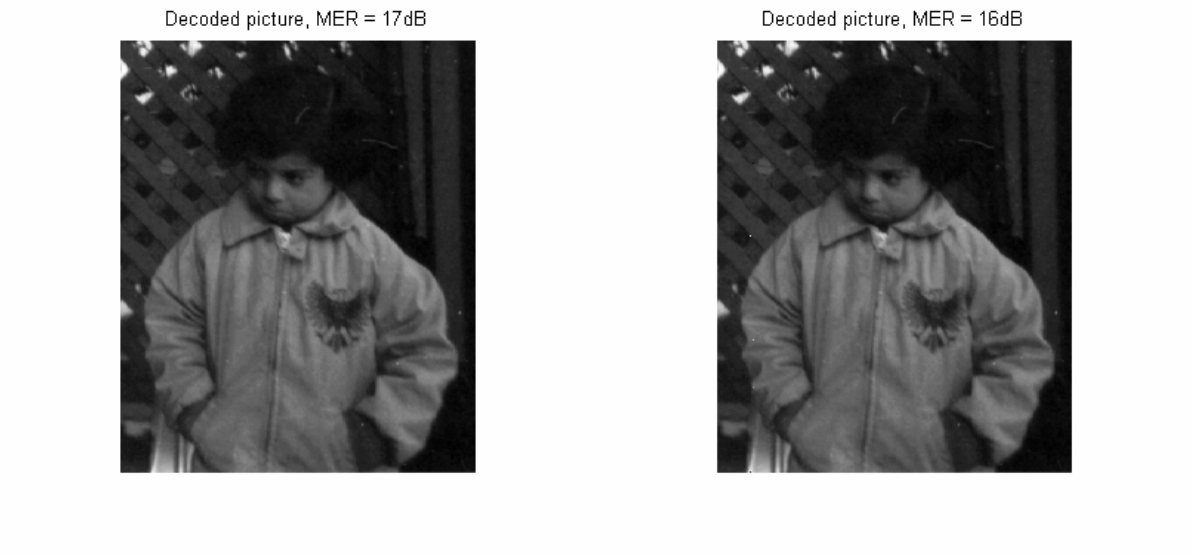


d) 64-QAM symbols with *MER* = 14dB

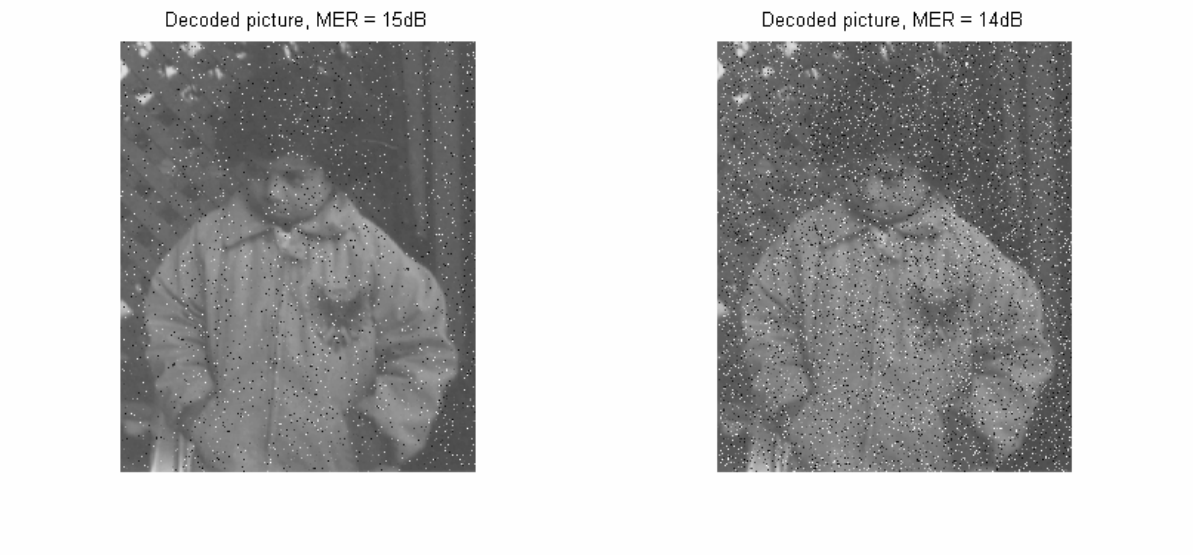
Figure 2: Received 64-QAM symbols with AWGN added

Figure 2 shows received symbols of 64-QAM modulation with added AWGN with decreasing *MER*. Although the received constellation with *MER* = 17dB is significantly distorted, data are decoded with *BER* = 0 and restored picture is without errors.

Errors are detected in pictures with levels lower than *MER* = 16, as is shown in Figure 3.



a) Decoded picture with *MER* = 17dB b) Decoded picture with *MER* = 16dB



c) Decoded picture with *MER* = 15dB d) Decoded picture with *MER* = 14dB

Figure 3: Pictures decoded from received symbols with AWGN added

5 Conclusion

Presented implementation of DVB-T channel coder and decoder and results of its simulation in Matlab are in general in accordance with assumptions of *MER* relation and error rates after corresponding error correction. However, simulated error rates are influenced by finite data length in opposite to continuous data stream in real digital television broadcast.

This work will continue with implementation of DVB-H extensions of cannel coding and following OFDM modulation and different transmissions channels and environments for broadcasting to fixed, portable and mobile receivers.

References

[1] ETSI EN 300 744 V1.5.1 (2004-11). Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for digital terrestrial television. ETSI, 11/2004. [Online] Available: http://pda.etsi.org/pda/queryform.asp

[2] ISO/IEC 13818 (Parts 1 to 3): Information technology - Generic coding of moving pictures and associated audio information. ISO/IEC. [Online] Available: http://neuron2.net/library/mpeg2/
[3] ETSI EN 300 421 V1.1.2 (1997-08). Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12 GHz satellite services. ETSI, 08/1997. [Online] Available: http://pda.etsi.org/pda/queryform.asp
[4] ETSI EN 300 429 V1.2.1 (1998-04). Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for cable systems. ETSI, 04/1998. [Online] Available: http://pda.etsi.org/pda/queryform.asp

---

Ondřej Hüttl, Ing.
Department of Radio Electronics, Brno Univ. of Technology, Purkyňova 118, 612 00 BRNO
E-mail: xhuttl00@stud.feec.vutbr.cz

Tomáš Kratochvíl, Doc. Ing. Ph.D.
Department of Radio Electronics, Brno Univ. of Technology, Purkyňova 118, 612 00 BRNO
E-mail: kratot@feec.vutbr.cz, Tel: +420 541 149 113, Fax: +420 541 149 244