Research and Realisation of Modified Turbo-Block-Codes (MTBC)

Dr. Kalyan Koora

Siemens Information and Communication Mobile Group

kalyan.koora@siemens.com

1 ABSTRACT

A new class of codes, turbo codes, have been introduced for channel coding in communication systems by French scientists in 1993. Due to the iterative decoding scheme of these codes, it is possible to achieve results near to the Shannon limit. Their outstanding performance compared to other forward error correction schemes attracted the interest of researchers. These codes are considered for the future telecommunications like Universal Mobile Telecommunications System (UMTS) and for the next generation wireless LANs. In order to decode these codes, implementation of a soft input and soft output (SISO) decoder is essential. The realisation of such type of decoders can be done either by using a maximum a posteriori (MAP) symbol estimation or by using a soft output Viterbi algorithm (SOVA). The main objective of this contribution is to present the results of the recent research study of different algorithms and design of a suitable hardware architecture for the introduced modified turbo block decoder.

2 INTRODUCTION

Turbo codes [1] are very interesting for digital data transmission because they allow an iterative method of decoding and correction of the channel errors. Each decoder is in a position to work on the received data along with the soft information provided by the previous iteration and produces an enhanced decoded output bits along with a soft information, which gives us a reliable information. Thus, the complete intelligence of the decoding is at the decoder. Since the function of these codes is equivalent to that of a turbo machine, these codes are named as turbo codes.

This paper starts with an introduction of turbo codes. In many systems mostly block codes are preferred to encode, transmit and decode the frames independently from one and other. Due to the application of the recursive systematic convolutional (RSC) codes the difficulties in terminating the trellis of the decoders are also pointed out. By using tail bits and the polynomial division property of RSC, a new approach to terminate both the decoders is presented, *Modified Turbo Block Codes (MTBC)*. Further, MTBC allows to use *variable* input block lengths and provides an extra information with which the decoders are able to enhance its correction performance by suppressing the *flattening*. In channel coding, interleavers are mainly used to

$$R_p = \frac{k}{k + n_1 + n_2} = \frac{R_{ü1} R_{ü2}}{R_{ü1} + R_{ü2} - R_{ü1} R_{ü2}} . \quad (3.1)$$

k is the no. of information bits. n_i represents the no. of redundancy bits generated by the i^{th} component code. This equation is valid for $R_{ü1}$ und $R_{ü2} \neq 0$. The code rate of the component codes is given by:

$$R_{ü1} = \frac{k}{k + n_1} \quad \text{und} \quad R_{ü2} = \frac{k}{k + n_2}. \quad (3.2)$$

To increase the overall code rate of the data transmission, some of the redundancy is punctured out of the code word. Punctured bits simply refer to the bits that are not transmitted at the output of the encoder. While decoding, these bits are filled either with zero information bits or by unsure information. The overall code rate of a punctured code is given by

$$R_p = \frac{k}{k + \rho_1 n_1 + \rho_2 n_2}, \quad (3.3)$$

where ρ_i represents the puncturing factor at the end of the i^{th} encoder, i.e. the no. of coded bits removed from the redundancy generated by the respective coder.

The correction capacity of turbo codes is dependent on different parameters like no. of iterations, selection of polynomials, selection of interleavers etc.. In general, the convolutional codes with greater minimum distance 'd_{min}' leads to better correction. Since turbo codes also belongs to this group, the increase in d_{min} leads to better correction of errors. One way to increase it is by selecting an optimum interleaver. For UMTS/International Mobile Telecommunication-2000 (IMT2000) a multi stage interleaver is selected where the rows and columns are shuffled like treating them as input of a small block interleaver. The minimum weight of these codes is achieved with the input block containing two '1', i.e. having a weight of 2. The following section presents a fastened systematic search of an optimum interleaver with the input frames of weight 2 and also discusses the achieved simulation results. The section following to it deals with a method of using all possible inputs of a given frame length for an interleaver search.

3.1 Selection of interleavers using input frames of weight 2

In [3] an algorithm is shown to select an optimum interleaver where the input frame weight 'd_{in}' is restricted to 2. If this method is applied to find an interleaver for an extended ATM of size N=448, with N as frame length, then total no. of inputs is equal to 100128. On the other hand, it

has to find an optimum from 448! possible interleavers. Here we describe an efficient method of search. For convenience, we represent 'd_i' as the weight of the redundancy part at the output of the *i*th encoder after puncturing. A desired final weight '$d'_{\min}$' at the output of the turbo coder is given as a start parameter. At first, a weight table for all input frames is generated, where the first and second bit positions represent the *y*- and *x*-axis of the table respectively. The weights 'd_1' of the output blocks after coding once and puncturing are enlisted in their respective positions. The starting state of the coder is set to a known state, e.g. '0' state as shown in [4]. An example for *N*=5 follows:

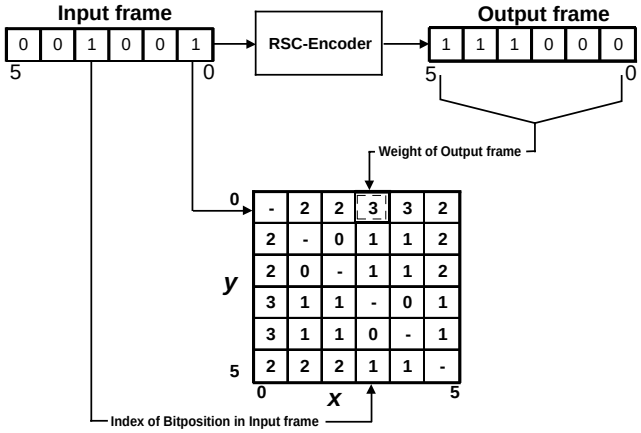


Figure 3.2: *Example for weight table preparation with frame length N=5.*

Before tests are conducted to select an optimum random interleaver, all input frames of weight 2 which leads to $(d_1 + d_{in}) > d'_{\min}$ are discarded, as the interleaver is not having any influence on d_1 and they however give rise to outputs with greater weights, i.e. $d_1 + d_2 + d_{in} > d'_{\min}$. This discarding leads to enormous time saving factor. It made easy to test 6049 inputs of frame length 448 instead of 100128 for a given polynomial. As the positions of the two bits of the frames are of interest, they are grouped together as one tuple. Now those tuples are put together to form a tuple table whose first bit position is same.

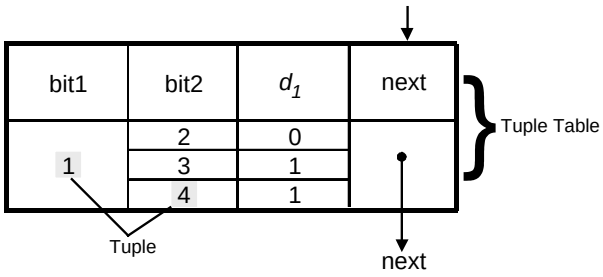


Figure 3.3: *Example of Tuple Table without the weight of the Input frame for d'$_{min}$=1.*

To start the search, either an arbitrary random interleaver or a given interleaver is considered, which is to be improved with respect to the final output weight $d_{\min} (= d_1 + d_2 + d_{in})$. Taking first tuple table, an exchange partner for the first bit position is chosen. Here one has to take into account whether this exchange has already been tested or it is allowed. With the help of the

weight and tuple tables, the resulted distance of the so formed interleaver pattern is computed. Only those parts of the tuple table are effected whose bit position is grouped in the table and the rest are unchanged. If the result $d_{\min} > d'_{\min}$, the new change in the interleaver is made, otherwise this exchange position is discarded. This process is done until all tuple tables are checked with the slowly modified interleaver. Finally, the resulted interleaver is saved as a new random interleaver, if its total weight is greater than $d'_{\min}$. The given minimum weight is then changed to the newly computed weight, i.e. $d'_{\min} = d_{\min}$. To exit the interleaver search one can either use time or maximum $d_{\min}$.

To test this new algorithm, simulations were carried out with turbo block codes using 2 iterations at SNR=3.75dB. The block length was fixed to 448, code rate to ½ and the memory of the RSC encoder to 3. A randomly selected interleaver was given as a starting parameter. With this systematic search for the interleavers, we were able to find 18 good random interleavers out of 10 million patterns which resulted in $d_{\min} = 23$. It was seen that the newly found best optimum random interleaver resulted in BER of 4.37e-6 where as the given interleaver pattern only 1.26e-5. We observed a coding gain of 0.25dB at SNR=3.75dB. It is further notified that not all interleavers which resulted in a greater $d_{\min}$ have good correction capacity. We suspect that not only the consideration of all input frames of size *N* but also the distribution of the redundancy is important factors in design of interleavers [5]. Thus, the selection of interleavers using input frames of weight 2 not only leads to good but also to a bad one with respect to the BER.

3.2 Use of all possible input frames

Actually, minimum distance for block-codes is calculated using all possible input frames. For small *N* one can generate all possible inputs, e.g. if *N*=21 then there are $(2^{21}-1)$, i.e. 2097151 possible inputs excluding the all zero. For *N*=448 it is difficult to use all the frames.

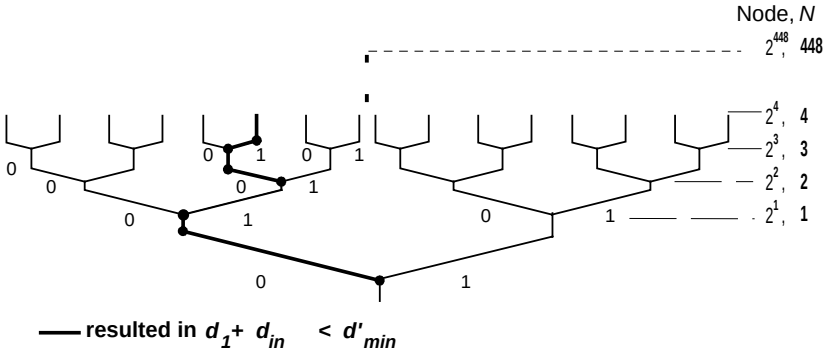


Figure 3.4: *Binary tree structure of search for input frames which leads to* $d_1 + d_{in} < d'_{\min}$.

To solve this problem we made use of binary tree structure of the input frames as illustrated in above figure. This selection algorithm of input frames depends only on the first time encoding and puncturing, i.e. d_1, where the interleaver doesn't have any influence. If and only if the resulted weight is smaller than the desired weight ($d_1 + d_{in} < d'_{\min}$), the path of the input frame is followed. The search of the input frames starts with N=1, i.e. with 2 possibilities '0' & '1'. After that it goes to the next node where there are 4 possible blocks '00', '01', '10' and '11'. As soon as the weight at a given node is greater than the desired weight it is discarded. In this way all the input frames which leads to an output weight less than the desired quality are taken into consideration to design an optimum interleaver.

Result, the first method is restricted only to the inputs of weight 2. Making use of pre computed weight tables and tuple tables one can fasten this search. The simulation results show that this method doesn't lead always to optimum interleavers because all the possible input blocks are not considered. The second method deals with the search of all possible input frames which can be used to design an optimum interleaver. The 'optimum' interleaver found through this systematic search has been considered for the implementation.

3.3 Polynomial Division Property

Here, the coding is achieved with a 'convolution' of input data sequel with generator polynomials. With an increase in the memory depth the no. of polynomial combinations also increase. The selection of an optimum polynomial naturally leads to best correction performance of turbo codes. Due to the application of the 'Recursive Systematic Codes (RSC)' the difficulties in terminating the trellis of the decoders were also pointed out in [6]. With the presentation of the Turbo-Block-Codes in [7] a hurdle of terminating one trellis is overcome. By using tail bits and the polynomial division property of RSC codes, we present a way to terminate both the decoders, which are used in an iteration of a turbo decoder separated by an interleaver. We also enlighten the flexibility of the cell size by a fixed code polynomial. Here we give a concise explanation of this property. All the considered polynomials *G(D)* for the recursion of the encoder have a particular polynomial *P(D)* which satisfy the equation

$$P(D) = f(D) * G(D) = 1 + D^l . \tag{3.4}$$

We define *P(D)* as the reset polynomial and *l* as the length of it. The division of *P(D)* by *G(D)* is aliquot. If the coderate of the encoder is restricted to ½, the selection of good polynomial combinations can be done with the equation

$$G_1(D) + G_2(D) = D * H(D), \tag{3.5}$$

where the grades of $G_1(D)$ and $G_2(D)$ are equal to the memory of the encoder 'm' and that of $H(D)$ is 'm-2'. $G_1(D)$ and $G_2(D)$ differ in the coefficients D and D^{m-1}. All the considered polynomials include the term 1. This class of codes possess a large free distance and does not lead to catastrophic codes [8]. These codes also include some of the well know primitive polynomials. For a primitive polynomial there exists a $P(D)$ with $l = 2^m - 1$.

If $G(D)$ divides $1+D^l$ without any remainder, then it also divides $1+D^{n*l}$ where 'n' is a natural number greater than 0. By encoding a sequence related to $1+D^{n*l}$ through a RSC encoder starting at the zero state, the encoder will be driven back to the zero state after the end of the sequence. This property along with the linearity feature is used to terminate the trellis of the second decoder of an iteration while decoding.

3.4 Modified Turbo-Block-Encoder

In order to make use of the above mentioned property and the tail bits, the encoder is modified as shown in **Figure 3.5**.

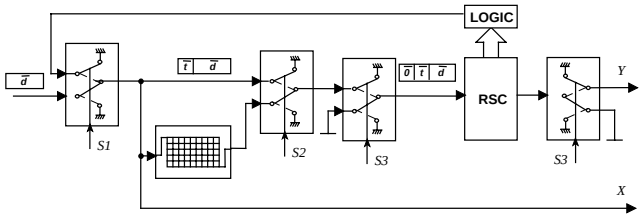


Figure 3.5: *Modified turbo block encoder (MTBC).*

The information data '$\bar{d}$' of N-bits, where N represents the size of a block, is first passed through the RSC encoder and at the same time fed into an interleaver of size equal to $(N + N_t)$ with 'N_t' denoting the number of tail bits, i.e. size of '$\bar{t}$'. The interleaver should assure that each bit $d(t)$ at its input should be equal to the output bit $d(t+x)$, with $x = (N + N_t + n * l)$. A logic circuit inspects the present state of the RSC encoder and generates the respective bit, with which the encoder is driven to the zero state in short time. Note the logic can be replaced with a modulo addition at the input of the RSC coder. After the last information bit has been passed through the coder, the 'S_1' switch selects N_t tail bits which depends on the memory of the encoder. If the sum of the data bits and the tail bits is not a multiple of the grade of reset polynomial, then 'N_0' number of zero bits '$\bar{0}$' are introduced into the data input of the RSC encoder by switching 'S_3'.

$$N_0 = i * l - (N + N_t) \geq 0. \tag{3.6}$$

'*i*' is the smallest integer which satisfies the required condition. During the insertion of zero bits the clocking of the interleaver is stopped. At the same time the output of the encoder is ignored, as these bits are simply used to satisfy the polynomial division property. Once the last zero bit has been passed through the encoder the reading of the interleaver data is done by selecting the 'S_2' switch. Through this modified method the number N is not fixed to a value which is a multiple of the length of the reset polynomial. Since the blocks are to be independent of each other, a continuous encoding is not possible.

For instance, a RSC encoder with the octal representation of the polynomials {13,15} is considered. The recursion polynomial is 13. The grade l of the reset polynomial is 7 and requires N_t=3 tail bits. If N is 440, then the total size of the interleaver would be 443 with 22 rows and 21 columns where the last row consists of only three elements. The interleaver is filled with the data in rows and the reading is done column by column. This makes sure that each input bit of the interleaver is written out after (443+7*n). If a search for an optimal interleaver is performed, then the shuffling of the order of reading is allowed only within a column. In the considered example there is a need of 5 zero bits so that the size of input frames of the encoder is a multiple of its reset polynomial grade 7. Now the systematic part 'X' consists of the data bits and the tail bits. The redundancy part 'Y' is punctured to obtain the required code rate.

4 TURBO DECODER

As already mentioned, two candidates are known to implement a SISO decoder, MAP and SOVA. Since SOVA algorithm is considered for implementation, here we give a synopsis of it. For thorough description of this algorithm please refer to [9]. In its basic form, the Viterbi algorithm accepts either hard or soft values of the received signals to compute the branch metrics for every state, which are added recursively to the old path metric values giving rise to new path metrics. These operations are done for every state and the path with largest metric value is selected as the survivor among the other paths merging in that state. This is done in the Add-Compare-Select (ACS) unit of a Viterbi decoder. The outputs of the ACS unit are then managed in the survivor memory unit. Finally, the state with the largest accumulated metric is selected as the survivor at the given time point, and the output of the decoder is taken from the survivor memory of that particular state. Better results can be achieved if the survivor memory unit is very large. But for practical realisation it is limited. This is also called truncation path length and depends on the memory which is used by the convolutional encoder. Acceptable value is five times the memory length [10]. For the soft output algorithm, not only the hard

values but also the metric difference ('Δ') between the survivor and concurrent paths are delivered by the ACS unit. Along with the hard values, these values are also stored in the soft value memory unit. For a time point '*t*' all the metric differences, which have been calculated till time '*t-1*' are updated by comparing the survivor and concurrent paths of each state in case if the hard output values are different. This update is done by selecting the minimum among the old value and the newly computed Δ. In order to keep the complexity of the computations and comparisons of Δ values acceptable, only codes with code rate = 1/*x* are considered for the practical realisation, where '*x*' represents the number of encoder outputs.

5 STEPS TO IMPLEMENT A SOVA DECODER

In order to find out the input, output and internal results word length without any great loss on the performance, the algorithm was implemented as a bittrue model for a simulation tool**.** **Figure 5.1** shows the results of the simulations, where the algorithm has been tested for a convolutional encoder and using the soft input decoder. With the help of those simulations, the word lengths for the signal processing have been chosen.

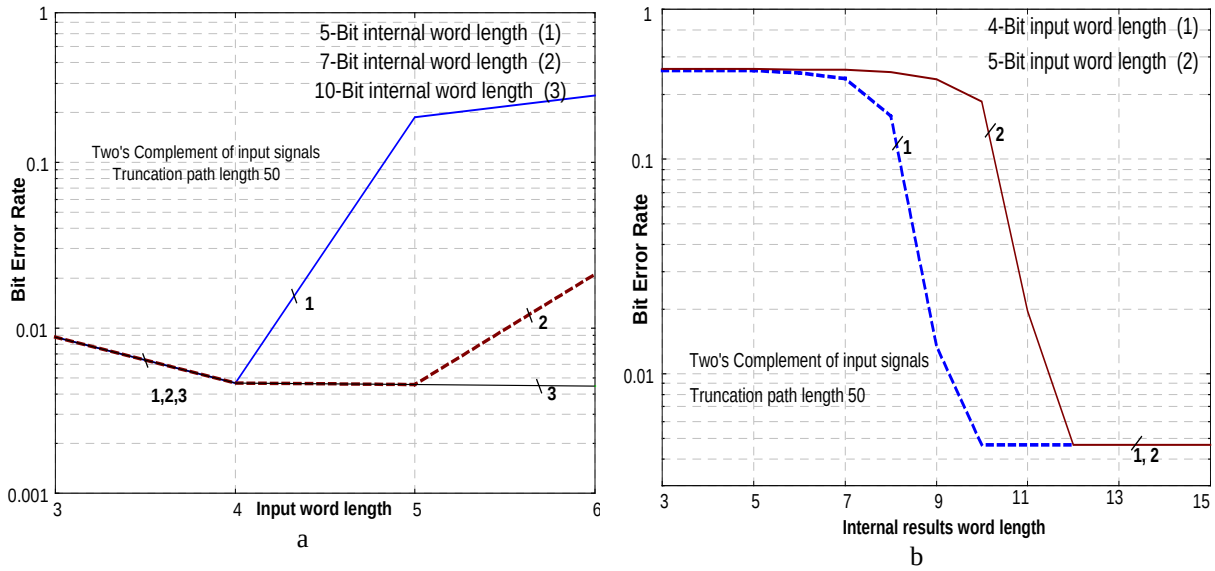


Figure 5.1: *Effect of (a) input and (b) internal results word length on the performance at SNR = 2.5 dB.*

By a fixed internal word length, though the input word length is increased there is degradation in the performance of the algorithm. This is because, by large input word lengths, the internal results reach the maximum limit very quickly and many of the accumulated values are manipulated, for example by clipping. The algorithm, which is used here, selects a path as the survivor, if its accumulated value is greater than that of the other path values. In case of clipping the results, many of the paths have the same maximum value. This increases the possibility for the selection of a wrong path as the survivor. **Figure 5.2** shows the histograms of the soft values, also called extrinsic informations, which are generated and quantised (4- and 5- bit

quantisation) by SOVA decoder, when only 1's are taken as input of the encoder. Due to less number of signal levels at 4-bit quantisation compared to 5-bit, there are many ‘sure’ events. This may degrade the performance of the next decoding stage, as all the events are not sure events at low SNR values.

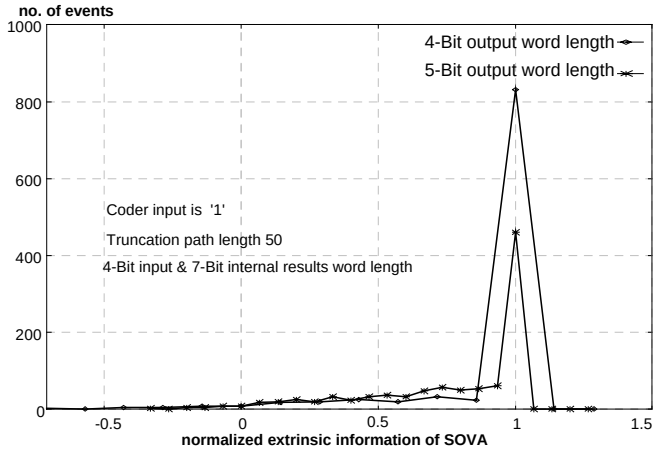


Figure 5.2: *Histogram of extrinsic information for given output word length at SNR = 2.5 dB.*

These simulations help the designer to choose the word lengths for the practical realisation. Through the achieved results we selected 4-bit word length for input and output ports and 7-bit word length for internal results.

6 TESTSYSTEM

Simulation tool aided algorithm test was the first step taken to implement the above described turbo decoder. C program realising a SOVA was written for the simulation tools. After successful implementation and testing of the floating point and bittrue software, word lengths are selected and a VHDL code has been prepared to realise the SOVA-Decoder. In a parallel approach the code was fed into ASIC design software tools. Concentrating on the flexibility of the codec for further tests, e.g. use of different random interleavers, influence of the word lengths in real systems etc., it was finally decided to test the decoder using field programmable gate arrays (FPGA).

Figure 6.1a shows the difference between floating point simulation and hardware measurements. The considered algorithm achieves a SNR loss of ca. 0.5 dB for the same coding conditions due to the quantisation. **Figure 6.1**b depicts the hardware circuit which realises the flexible turbo block decoder. The control of the hardware circuit is taken up by a personal computer. The communication between the PC and the circuit is achieved via parallel port. Since the circuit performs only decoding, the PC also performs rest of the functions like data generation, encoding, puncturing, distortion and quantisation. The circuit supports two modes.

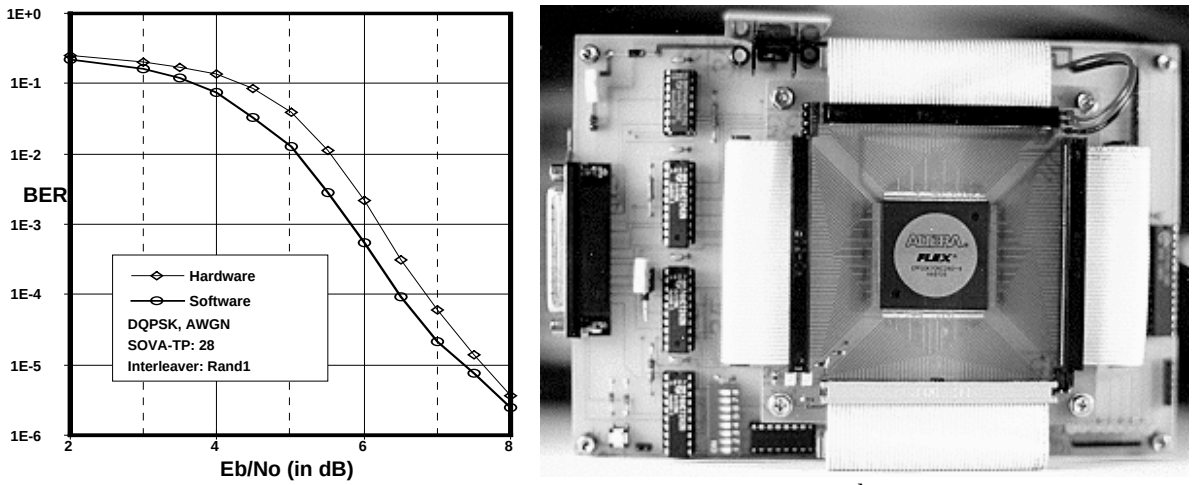


Figure 6.1:*a) Implementation loss b) Hardware circuit of Turbo-Block-Decoder.*

In first and simple mode the circuit takes up the received and extrinsic information and performs one complete iteration and gives back the new extrinsic information. This allows to observe the development of the results after every iteration. As the communication via parallel port is considered to be one of the bottleneck, this mode is very slow compared to the second and fast mode. In the second mode the iteration number 'I' is set as a parameter. After the reception of the information block along with the redundancy information, the desired number of iterations are performed by the hardware circuit and final results are given back to PC. The functional diagram is shown in the following figure:

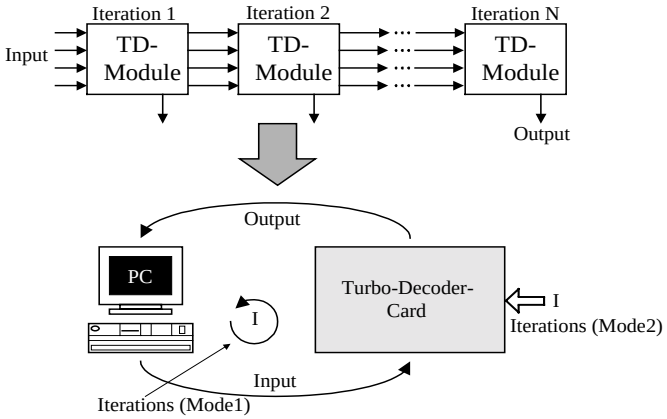


Figure 6.2: *Mutual communication between hardware and software.*

Using the slowest speed grade of the FPGA, a maximum throughput of 14 MBit/s channel data was measured for Mode 1. Communication between PC and hardware is also considered as a bottleneck for fast tests. The system design was further developed to increase the throughput up to 30 MBit/s with extending hardware resources. It is possible to achieve a throughput which is about twice the present rate by using faster speed grades of FPGAs.

7 SIMULATION AND MEASUREMENT RESULTS

To test efficiency of the new modified block decoding of turbo codes, at first simulations were carried out using a software tool under different communication environments. Here we explain one of them where the block length *N* was 440 bits which is equal to an extended ATM cell in MEDIAN. In order to be independent of carrier phase estimation differential QPSK modulation scheme was selected. **Figure 7.1**a and b depicts the comparison between turbo block codes with two decoding iterations with SOVA truncation path length 56 and Reed-Solomon Coding schemes. The code rates are close to '5/7'. TC1[619;440] and TC2[616;440] represent the AWGN channel with termination of all decoders (new method) and termination of only the second decoder (old method) respectively. In the same way TC3[619;440] and TC4[616;440] represent the multipath channel with Line-of-Sight (LOS) environment [11]. Since the new scheme requires more sub-carriers, the number of used sub-carriers by the OFDM was increased from 310 to 312. Synchronisation of the receiver was done on a reference symbol inserted into each TDD frame.

The new kind of block decoding leads to a gain of 0.2dB SNR for AWGN channel and 1.5dB SNR for LOS-MEDIAN modelled channel at a BER of 2e-5.

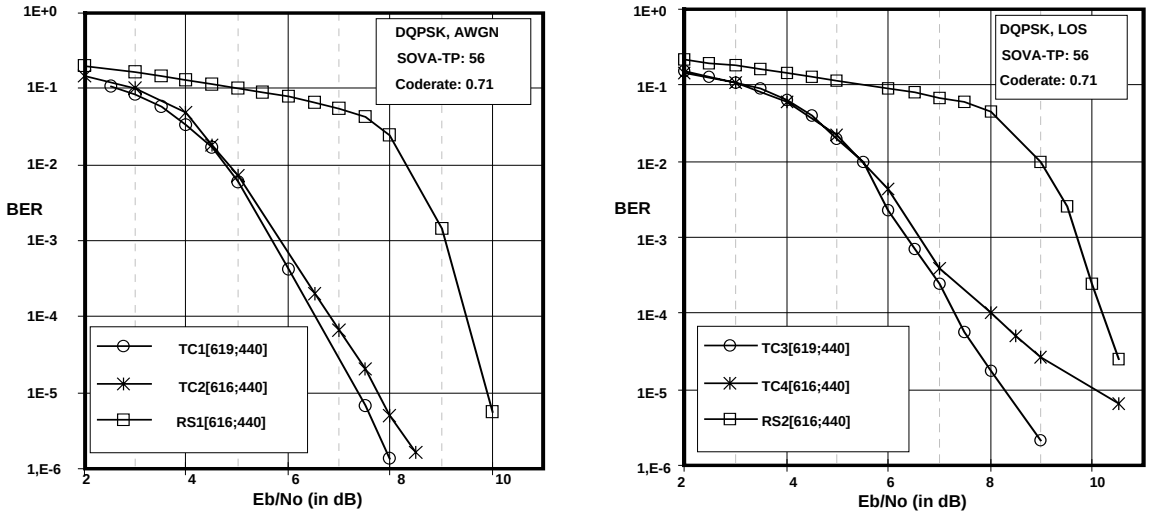
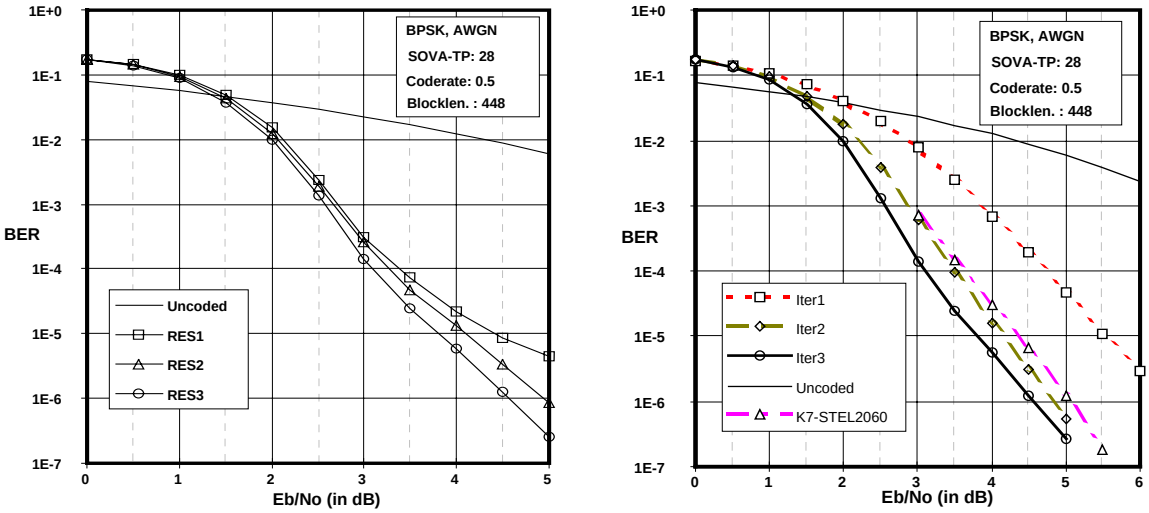


Figure 7.1: *Comparison of Turbo Codes (new and old termination scheme) with Reed-Solomon codes for 310/312 used sub-carrier DQPSK-OFDM system in (a) AWGN and (b) LOS environment.*

Thus, for multipath channel there is an appreciable increase in coding gain especially for small BER (TC3 compared to TC4). Another effect of turbo block decoder, which arises by breaking the decoding scheme at the end of a block without terminating perfectly, is the flattening of the achievable bit error rate even by increasing iteration numbers. Influence of this inconvenience is also decreased using this method of termination. The RS1[616;440] shows the BER for AWGN channel whereas RS2[616;440] denotes for LOS channel model, both are outperformed by the

turbo coding scheme (new and old). At BER of 2e-5, the new turbo block code shows a gain of 2.8dB on SNR for AWGN channel compared with Reed-Solomon codes and 2.5 dB for LOS channel model. The improved coding gain and the substantial decreasing of the flattening level resulting from the new reset method is representative for all code polynomials. Since this algorithm brings a small change in the encoder structure, it can be applied for all types of soft-input-soft-output turbo decoding processors (MAP / SOVA). An other main advantage of this new coding scheme is its inherent possibility to process variable input block lengths for the same coding polynomial. This property can be utilised for systems where the block lengths vary for different applications.

In **Figure 7.2** the measurement curves using the hardware circuit are shown. Coherent BPSK modulation scheme was applied and the truncation path length was set to 28. Selection of the random interleaver was done by a systematic search depending on the weight distribution table of the code words at the output of the encoder after puncturing as mentioned in 3.1.



a b

Figure 7.2: *(a) Development of results after 3 iterations (b) Iteration effect.*

Figure 7.2a presents the development of the results. RES1 stands for the decoding type where the starting state of the first decoder and the end state of the second decoder of an iteration are know. RES2 show the new turbo block decoding scheme. RES3 is same as RES2 but resulted soft values are kept in certain bounds (weighting the soft values). Thus, if RES1 is compared with RES3 a gain of 1dB SNR is noticed at BER of 7e-6. **Figure 7.2**b shows the development of the results after every iteration. Further, the performance is compared with the commercial chip STEL2060 whose constraint length is 7. After three iterations, turbo block decoder is able to achieve a gain of 0.5 dB SNR compared with the commercial chip.

Figure 7.3a and b represent results using the hardware in MEDIAN environment. Modulation of sub-carriers was set to DQPSK, differential to adjacent sub-carriers, and code rate to ½. The MEDIAN system synchronisation and phase noise were set to ideal. In contrast to the above mentioned software simulation results the truncation path length was 28. At BER of 1e-6, which is interesting for ATM transmission, a total coding gain of 4.5dB has been measured in AWGN channel whereas 11dB coding gain has been achieved in LOS channel. In all the cases it has been observed that gain on SNR between iterations 1 and 2 is greater than that between iteration 2 and 3. This natural behaviour is additionally forced due to absence of the optimal weighting of the extrinsic values between decoding iterations and limiting the internal word lengths in order to consume less hardware.

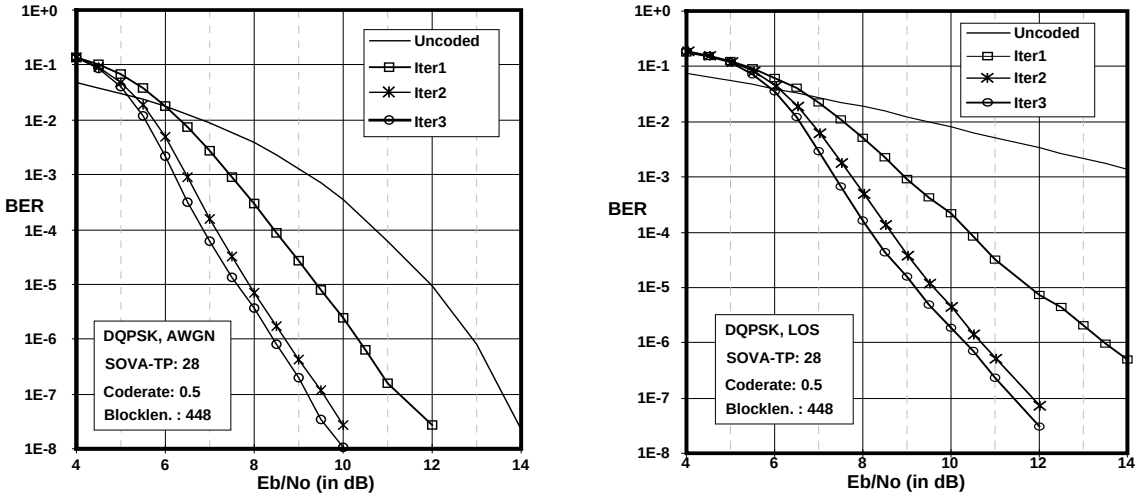


Figure 7.3: *Measured results (a) AWGN and (b) LOS.*

8 COMPARISON OF DIFFERENT SISO DECODERS

In modern communications, the trend is to miniaturise the systems for less power consumption and less weight. Thus, there is a necessity to develop algorithms which are not only optimal in their function but also occupies less hardware and consume less power. This section deals with power consumption and delay factor of the SISO decoders. In [3] it has been shown that the selection of a RSC coder with 8 states gives a good price performance ratio. Therefore, this coder is taken as an example to describe the calculations. It can be extended to other codes. For convenience, following abbreviations are used:

- SOVA: two step soft output Viterbi algorithm, the soft updates are done according to [9] only for the fist half of truncation path. The remaining half is simple update.
- B-SOVA: also a two step SOVA, but the updates are done according to [12]
- S-SML-MAP: slow sliding max-log MAP, where only one add-compare-select (ACS) unit is used to calculate all the forward and backward recursions [13]

- F-SML-MAP: fast SML-MAP, each forward and backward recursion has its own ACS unit

Power Dissipation: The average power dissipation in traditional CMOS circuits can be represented as three components [14, 15].

1) Switching Power Dissipation: This component, also know as dynamic dissipation, is independent of the logical function but results due to the charge of the capacitors at the input and output ports of the gates and wiring of gates (C_{load}~0.05pF). The average power consumption of the nodes can be calculated with the amount of energy (E_{load}) required to charge the nodes to operating voltage (V_{DD}).

$$P_{avg}(d) = \frac{\alpha_t}{T_{sys}} E_{load} = \alpha_t f_{clk} C_{load} V_{DD}^2. \tag{8.1}$$

The factor α_t conveys the number of effective node transitions within a clock cycle (f_{clk}) in a chip. It is taken as 0.2 times the gate count.

2) Short Circuit Power Dissipation: Due to the presence of small capacitors at the outputs of the gates, the fast change of the signals, until they reach a stable stage, lead to form a direct current path between the power supply and the ground. This current component during switching doesn't contribute to the charging of the capacitance of the ports to desired voltage level and therefore leads to short-circuit power.

$$P_{avg}(s) = \frac{1}{12} \alpha_t b \tau f_{clk} (V_{DD} - 2V_T)^3. \tag{8.2}$$

For a simple analysis it is considered that $\tau=\tau_{rise}=\tau_{fall}$=2ns, $b=b_n=b_p$ transistor gain factor ~50µA/V^2, V_{DD} = 5V and $V_T = V_{Tn} = |V_{Tp}|$ threshold voltage= 0.7 V

3) Leakage Power Dissipation: The considered transistors have very small leakage current ($I_{reverse}$). But when the gate count is large, this current is perceptible. It is given by $P_{avg}(l) = V_{DD} I_{reverse}$ and also occurs in the stand-by mode. A typical value for the reverse leakage current at room temperature is 2.5 pA/mm^2. Now the complete average power dissipation can be calculated:

$$P_{avg} = \alpha_t f_{clk} C_{load} V_{DD}^2 + \alpha_t V_{DD} I_{short} + A V_{DD} \frac{2.5pA}{mm^2} \tag{8.3}$$

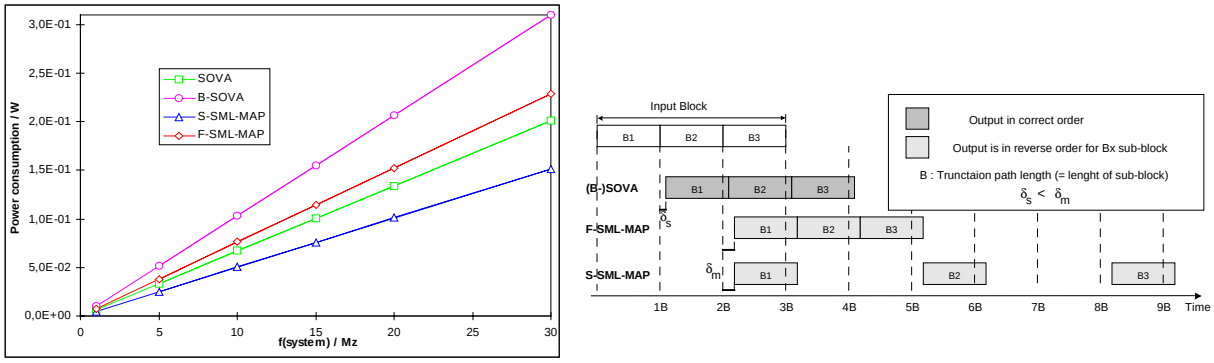
With this equation it is possible to estimate the average power consumption of the decoders. For SML-MAP decoders some delay elements and the control logic are not considered. The SOVA decoders need no RAM at the input and output of the unit, which is essential for SML-MAP decoders. Since most of the modern communication systems have an internal RAM which can also

be used by the decoders, this aspect is also not taken into calculation. For all decoders the observation length (window length) is fixed to 28.

	SOVA	B-SOVA	S-SML-MAP	F-SML-MAP
BMC	811			2433
ACS	15688			47064
Pathmanagement	64620	108860	19375	42975
Softvalue Computation	620			
Total Trans.	81739	125979	36494	93092
Total Gates = (Trans/4)	**20435**	**31495**	**9123**	**23273**
Core Area [mm²]	12.16	18.74	5.43	13.85
α = 0.2*gates	4087	6299	1824.6	4654.6
P(d) = α*C*V^2*f [w]	0.1532625	0.2362125	0.13029375	0.1745475
P(l) = V*I(rev)*Area [w]	1.52E-10	2.3425E-10	6.7875E-11	1.7313E-10
P(s) [w]	0.04767077	0.07347154	0.02128213	0.05429125
Power Consumption [w]	**0.20093327**	**0.30968404**	**0.15157588**	**0.22883875**

Table 1: *Estimation of hardware resources and power consumption.*

Table 1 shows the hardware complexity and the power dissipation of the four considered SISO decoders at a system clock of 30 MHz. For the calculations only the power is taken, which is need to decode one complete input block. Thus, $P_{avg}(d)$ of S-SML-MAP decoder has an extra factor, as the ACS units are used thrice for one sub-block. For the area calculation 0.75 µm technology is used. With new technologies the area will change, but the gate count will be the same. Further, it should be noted that due to change in the parameters only the absolute values are altered, but the difference is unchanged. Therefore, this calculation can be used as an approximation to compare the decoders. As it is depicted in **Figure 8.1**a, power dissipation is directly proportional to the system clock speed. If an input block with 3 sub-blocks is considered, **Figure 8.1**b shows the delay introduced by the decoders between the input and output. Decoder latency depends on the observation length.



a b

Figure 8.1: *a)Power consumption and b) delay comparison of different SISO implementation algorithms.*

SOVA decoder with register exchange path management, which needs around 0.5dB more SNR for same BER, is a ‘straight forward’ solution. Its total latency is equal to the observation length and a small delay δ_S, which depends on the pipe lining architecture and is usually equal to 5 clock cycles. Though the realisation of B-SOVA is same as SOVA, due to the extra update option it requires much hardware resources. The decoder latency of SML-MAP is greater. It can start its calculations only if at least two sub-blocks are stored in memory. Further, the outputs are in reverse order which requires RAM to output decoded data in correct order. Anyhow, the presence of interleavers for turbo codes solves this problem. From the above deliberation it can be concluded that for a high speed or for a continuous SISO decoding SOVA is a well suitable. If the delay is not the bottleneck, then S-SML-MAP is a good choice.

9 CONCLUSIONS

In this paper we have shown two new methods to select the input frames for design of an "optimum" interleaver for turbo block codes. The first method is restricted only to the inputs of weight 2. Making use of pre computed weight tables and tuple tables one can fasten this search. The simulation results show that this method doesn’t lead always to optimum interleavers because all the possible input blocks are not considered. The second method deals with the search of all possible input frames with a help of a binary tree which can be used to design an optimum interleaver. In the introduced modified turbo block codes, the termination of the decoders results in an improvement of the correction capacity of the decoder at high SNR values by suppressing the early flattening effect. This type of decoding allows to keep the block length variable for a given code polynomial. We have also presented the selection of generator polynomials for the coding scheme.

The first design of 30 MBit/s flexible hardware realisation of this turbo decoder for block oriented transmission using FPGA is presented. Theoretical simulation results are verified developing this hardware turbo decoder. In addition, the possibility to investigate bit error rates less than 1e-7 is created. Achieved simulation and measured results using AWGN and time invariant modelled 60GHz indoor wireless LAN channel are presented. An appreciable coding gain is achieved in both cases using the introduced modified turbo encoder for block transmission.

A procedure to compute power consumption is described and the different SISO algorithms are compared. The implementation of SOVA decoder with register exchange path management, which needs around 0.5dB more SNR for same BER, is a fastest and straight forward solution. If the delay is not the drawback, then slow sliding max-log MAP is a good choice.

Acknowledgements

The author would like to thank Prof. A. Finger of Dresden University of Technology, Germany, from department of Electrical Engineering and Information Technology for guiding and supporting the work. Further, the useful discussions with the staff members of the 'Communications Theory' department is gratefully acknowledged. It shall be emphasised that most of the used physical layer concept is a part of the overall MEDIAN consortium.

10 REFERENCES

1. Claude Berrou, Alain Glavieus, Punya Thitimajshima: Near Shannon Limit Error-Correcting Coding and Decoding (TURBO-Codes), International Conference on Communication (ICC), Geneva Switzerland, May 1993, pp. 1064 - 1070
2. Corrado Ciotti, Jörg Borowski: The AC006 MEDIAN Project-Overview and State of the Art, Proc. of ACTS Mobile Communications Summit, Granada, Spain, 27-29 November 1996, Vol. 1, pp. 362-36
3. P. Jung and M. Naßhan: Performance evaluation of turbo codes for short frame transmission systems, Electronics Letters, Vol. 30, No. 2, 20th January 1994, pp. 111-112
4. K. Koora and A. Finger: A New Scheme to Terminate all Trellis of TURBO-Decoder for Variable Block Length, in Proc. of the Int. Symposium on Turbo Codes and Related Topics, Ecole Nationale Supérieure des Télécommunications de Bretagne, France, 3-5 September 1997, pp. 174 - 179
5. A. S. Barbulescu and S. S. Pietrobon: Interleaver design for turbo codes, Electronics Letters, Vol. 30, No. 25, 8th December 1994, pp. 2107-2108
6. Patrick Robertson: Illuminating the Structure of Code and Decoder of Parallel Concatenated Recursive Systematic (Turbo) Codes, Proceedings IEEE GLOBECOM'94, San Francisco, 1994, pp. 1298 - 1303
7. Claude Berrou, Stéphane Evano, Gérard Battail: Turbo-block-codes, Proceedings of TURBO CODING Seminar at Lund University, Sweden, 28-29 August 1996, pp. 1 - 8
8. Farhad Hemmati and Daniel J. Costello, JR.: Asymptotically Catastrophic Convolutional Codes, IEEE Transactions on Information Theory, Vol. IT-26, No. 3, May 1980, pp. 298 - 304
9. Joachim Hagenauer, Peter Hoeher: A Viterbi Algorithm with Soft Decision Outputs and its applications, Proceedings of IEEE GLOBECOM, November 1989, pp. 1680 - 1686

10. Ivan M. Onyszchuk: Truncation Length for Viterbi Decoding, IEEE Transactions on Communications, Vol. 39, No. 7, July 1991, pp. 1023 - 1026
11. J. Hübner, S. Zeisberg, K. Koora, J. Borowski and A. Finger: Simple Channel Model for 60GHz Indoor Wireless LAN Design Based on Complex Wiedeband Measurements, Proceedings of 47th Vehicular Technology Conference (VTC), Phoenix, Arizona, USA, 4-7 May 1997, pp. 1004 - 100
12. Lin, L., Cheng and Roger S.: Improvements In SOVA-Based Decoding For Turbo Codes, Proc. of ICC, Montreal, Canada, June 97, pp. 1473 – 1478
13. P. Robertson, E. Villebrun, and P. Hoeher: A comparison of optimal and sub-optimal decoding algorithms in the log domain, Proc. ICC, (Seattle, WA), June 1995, pp. 1009-1013
14. Weste, Niel H. E and Eshraghian, K.: Principles of CMOS VLSI Design, Addison-Wesley Publishing company, October 94, ISBN: 0-201-53376-6
15. Mlynek, D. and Leblebici, Y.: Design of VLSI Systems, Integrated System Center, Switzerland, http://c3iwww.epfl.ch/teaching/webcourse/