The design and implementation of TPC encoder and decoder

L J Xiang[1], Z B Wang[2], J B Yuan[1] and L H Zheng[1]
[1] School of Electronic Sciences and Engineering, the National University of Defense Technology, Hunan Chang-sha, 410073, China
[2] National Key Laboratory of Science and Technology on Information System Security, Beijing, 100101, China

Abstract. Based on the analysis and simulation of TPC codec principle and iteration decoding algorithm based on Chase 2, the design and implementation methods of TPC encoder and decoder are presented in this paper. In particular, circuit design and implementation flow of soft-input soft-output modified method in decoder algorithm and iteration decoding are analyzed. Aiming at (64,57,4) TPC, when the iteration times is 3 and the uncertain position number is 3, simulation and implementation results show that at least 6.8dB encoding gain can be obtained under the condition of BER=10^{-6}.

1. Introduction

Turbo Production Code (TPC) [1] comes from the Turbo Convolution Code (TCC) [2]. Block Turbo Code (BTC) was carried out by Pyndiah in 1994, and by using the BTC as sub-code, He designed the Chase decoding algorithm which can achieve SISO iteration[3]. This kind of iteration was applied in TPC[4]. In 1998, a modified Chase decoding algorithm was brought out by Pyndiah. Numerous experiments have proved that, compared with the Turbo code with sub-code of convolution code which generally 6-10 iterations is necessary, TPC with sub-code of blocking code converge faster, and only 4-5 iterations is necessary to approximately achieve the coding gain of the PCCC SISO decoding algorithm. In terms of the decoding, TPC is similar to the TCC. TPC has the performance of higher encoding efficiency and lower decoding complexity and it has lots opportunities in deep space communication, satellite communication and other digital communication systems.

2. Principles of TPC

TPC is a blocking code, generally, two or more blocking codes are coded to two or much bigger dimension coding blocks, and the blocking codes are the sub-code of the TPC, which can be same or difference. Blocking code of Hamming code, Extended Hamming code, BCH code and RS code, etc. are usually adopted as the sub-code.

2.1. Encoding Mechanism of TPC

Suppose two linear block code $C_1(n_1,k_1,d_1)$ and $C_2(n_2,k_2,d_2)$ is used as sub-code, where n is the code length, k is the length of the information, d is the minimum Hamming distance. Two dimensions TPC can be represented by $C_1 \otimes C_2$. The encoding process is as follows.

Firstly, sequentially read the $k_1 \times k_2$ information block as the first k_1 rows and k_2 columns.

Secondly, encode the k_1 row with $C_2(n_2,k_2,d_2)$.

Thirdly, encode the n_2 column with $C_1(n_1,k_1,d_1)$.

The parameter of the TPC is: code length $n = n_1 \times n_2$, information length $k = k_1 \times k_2$, minimum Hamming distance $d = d_1 \times d_2$, and the encoding efficiency $R = R_1 \times R_2$,among which, R_1、R_2 is the encoding efficiency of C_1、C_2 . From the encoding theory, if $t_1 = \left[(d_1 - 1)/2 \right]$, $t_2 = \left[(d_2 - 1)/2 \right]$ random errors can be corrected by C_1 and C_2 , respectively, $t = \left[(d_1 d_2 - 1)/2 \right]$ random errors can be corrected by $C_1 \otimes C_2$. It can also prove that $b \le \max(n_1 t_2, n_2 t_1)$ burst errors can be corrected by $C_1 \otimes C_2$. So, the TPC is kind of code with strong error-correct abilities[5], and its encoding structure is depicted as follows.

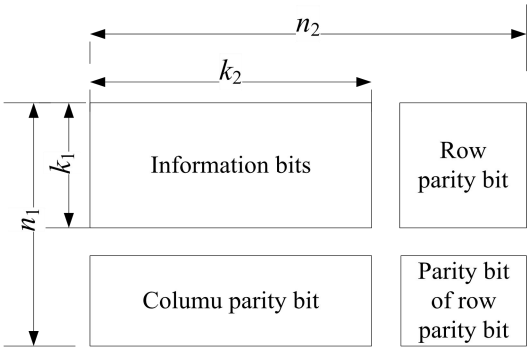

Figure 1. Encoding structure of TPC

2.2. Decoding Mechanism of TPC

2.2.1. TPC decoding structure. TPC is a serial concatenated code, it has good performance with an iterative decoding algorithm, among which, the unit-decoder is the most important part. The TPC unit-decoder is constituted with two parts, one is a soft input hard output decoder which is based on the Chase2 algorithm, and the other part is the calculation of extrinsic information which used to convert the hard input into soft output. Suppose the received code word matrix is r , extrinsic information matrix is $[W(m)]$, where m denotes the m th unit-decoder, and the soft input information is $[R(m)]$, soft output information is $[R'(m)]$. Based on the Chase2 decoding algorithm, the unit-decoder's work can be divided into four steps:

a) Calculate the soft input information $[R(m)]$ with the multipliers and adders.

b) Define the decoding subset Ω based on Chase 2 algorithm, get the decision code C_C , and find its competitive code C_D .

c) Calculate the soft output information $[R'(\mathrm{m})]$ and extrinsic information $W\left[(m+1)\right]$ with C_C and C_D .

d) Transfer the matrix $W[(m+1)]$ to next decoder to update the previous in it and the new extrinsic information $W[(m+1)]$ will be used as prior knowledge in decoding[5].

The soft input matrix of the m th decoder is denoted as $\left[R(m)\right] = r + \alpha(m)\left[W(m)\right]$, among which, $\alpha(m)$ is a factor to adjust the peak value of the m th decoder, and it can be changed flexibly based on different sub-code of TPC and different iteration decoding times. Input the $\left[R(m)\right]$ into the SISO decoder, and then decode it row by row (or column by column) to get the soft output matrix $\left[R'(m)\right]$. Subtract the received code word from $\left[R'(m)\right]$ and $\left[W(m+1)\right] = \left[R'(m)\right] - r$, and the result will be used as next decoder's prior information. Calculate the soft input matrix $\left[R(m+1)\right]$ of the next half iteration with $\left[W(m+1)\right]$, and repeat the process depicted above to decode column by column or row

by row, than an iteration decoding process of a two-dimension TPC is completed. Concatenate multiple unit-decoders can make a concatenated decoding structure which is shows as follows.

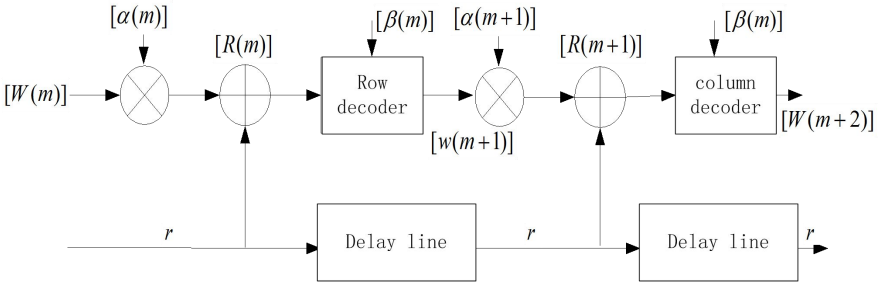

Figure 2. Concatenated iteration decoding structure

2.2.2. Chase2 Decoding Algorithm. The Chase 2 decoding algorithm is the most important and the most complex point of a unit-decoder. When a linear block code C is sent through a binary AWGN channel, suppose the received code word is R, and based on the maximum likelihood principle, the optimal decision word D is carried out by equation (1),

$$D = C^i \quad if \left|R - C^i\right|^2 \le \left|R - C^l\right|^2 \quad \forall l \in \left[1, 2^k\right] \qquad \text{* MERGEFORMAT (1)}$$

Equation (2) shows the Euclid distance:

$$\left|R - C^i\right|^2 = \sum_{l=1}^{n} \left|r_l - c_l^i\right|^2 \qquad \text{* MERGEFORMAT (2)}$$

We need to exhaust all possible code words to make a best decision, of which the complexity is very high. Chase 2 is a sub-optimal algorithm to decode the linear block code. When we decode the received information with the maximum likelihood principle, the produced word will drop into the sphere, whose centre is located at $Y = (y_1, \cdots, y_i, \cdots, y_n)$ and the radius is $(\delta - 1)$ with high probility. Y is the hard decision of the received signal. So, not all the code words should be considered, steps of Chase 2 is depicted as follows,

a) Select the most unreliable positions of Y and its number must be more than $p = \lfloor d/2 \rfloor$, the reliable is given by Equation (3):

$$\Lambda(d_j) = \ln\left(\frac{\Pr\{e_j = +1 / d_j\}}{\Pr\{e_j = -1 / d_j\}}\right) = \left(\frac{2}{\sigma^2}\right) d_j \qquad \text{* MERGEFORMAT (3)}$$

b) Construct the test pattern $T^q, q = 2^p$.

c) Construct test sequence Z^q according to $Z^q = Y \oplus T^q$.

d) Decode Z^q, and the result set is Ω.

e) Calculate the Euclid distance between code word C in Ω and the received word R, the word with the minimum distance is chosen as D.

As the square computation is hard to implement in hardware platform when calculates the Euclid distance. In practice, the Euclid distance calculating is simplified to correlation value calculating that showed in Equation (4). The correlation value calculating have advantages compared with the Euclid distance calculating, such as correlation value calculating only need the adder and subtractor, and the multiplier is not necessary.

$$COR\_V_{i+1} = \begin{cases} COR\_V_i + r_i & if\ z_i = 1 \\ COR\_V_i - r & if\ z_i = 0 \end{cases} \qquad \text{* MERGEFORMAT (4)}$$

After getting the decision code word, in order to implement the iterative decoding, the Chase decoding result should be outputted as soft data, the data with reliability measure. The reliability measurement of the hard decision needs two code words, one is the hard decision result, and the other is the competitive code which has a great effect on the complexity and performance of the decoding

algorithm. For simplification, a factor β is directly used instead of looking for the competitive code. Simulation results shows that more than 1 dB is lost in performance. In this paper, a modified method for looking for the competitive code and calculating the soft output information is proposed, with which the decoding complexity is slightly increased but the decoding performance is greatly enhanced. For example, when a TPC with the structure of （64,57,4）is decoded by using 3 times iteration , 3 unreliable positions, and the code gain is 6.8dB under the condition of BER=10^{-6}.
Equation (5) shows how to find the competitive code, and $c\_se(i) = -1$ represents that the competitive code is not found.

$$c\_se(i) = \begin{cases} -1 & if\ z_i(k) = z_d(k) \\ k & if\ z_i(k) \neq z_d(k)\ \&\ COR\_V(k) > \max(COR\_V) \end{cases} \quad i = 1,\cdots,n;\ \ k = 1,\cdots,q \quad \text{* MERGEFORMAT}$$

In calculating the soft output information, From Equation (6), we can see that when the competitive code can not be found, the result is substituted by a factor β .

$$R' = \begin{cases} COR\_V(d\_se) - COR\_V(c\_se) & if\ c\_se(i) \neq -1\ \&\ if\ z_d(k) = 1 \\ COR\_V(c\_se) - COR\_V(d\_se) & if\ c\_se(i) \neq -1\ \&\ if\ z_d(k) = 0 \\ \beta & if\ c\_se(i) = -1\ \&\ R' > 0 \\ -\beta & if\ c\_se(i) = -1\ \&\ R' < 0 \end{cases} \quad \text{* MERGEFORMAT (6)}$$

$$i = 1,\cdots,n;\ \ k = 1,\cdots,q$$

When decoding a row or a column, if the competitive code can not be found, make a statistic as follows,

$$\begin{aligned} ex\_sum &= ex\_sum + |R' - R| \\ ex\_num &= ex\_num + 1 \end{aligned} \quad \text{* MERGEFORMAT (7)}$$

After decoding a row or a column, Equation (8) is implemented,

$$ex\_sum = ex\_sum / ex\_num \quad \text{* MERGEFORMAT (8)}$$

The soft output is finally achieved by Equation (9),

$$R' = \begin{cases} R' \times ex\_sum & if\ c\_se(i) = -1 \\ R' & if\ c\_se(i) \neq -1 \end{cases} \quad \text{* MERGEFORMAT (9)}$$

3. Performance Analysis of TPC Decoding

In order to simulate the performance of TPC, a two-dimension TPC code with the structure of （64,57,4）is used, and its columns and rows are all use the extended Hamming code.

3.1. Effect of the decision method and the quantization bits

In order to make best use of the information from every decoder, TPC use soft decision, and the output of the decoders is soft information. In AWGN, compared with hard decision, the system performance in terms of errors is enhanced around 2dB[6].

Compared with the unquantized method, the fixed point quantization can lose some code gain. In this work, we simulate the TPC with float point and fixed point quantization, respectively. The results shows that, in the AWGN channels, the code gain lose can be controlled under 0.2dB if more than 8bits is adopted in the linear quantization. But if the quantization bits are increased, the performance is not correspondingly increased.

3.2. Effect of the Iteration

The output of each decoding iteration includes the received signal of the decoder and the previous iteration, and more repeats can lead to a more reliable decision. In the SISO serialized decoding, increase the iteration time can increase the decoding performance, but when the iteration time is increased beyond a certain number, the performance changes little. The decoding complexity and the time delay are all increased along with the increase of the iteration time, and the decoding speed oppositely changes. Usually, 3-5 iterations is used to decode TPC. Simulations with 8bit quantization

and 3 unreliable positions shows that the code gain can reach more than 6.8dB under the condition of BER=10^{-6}, and when the iteration time is increased, the code gain can also be increased but can not be increased in direct proportion.

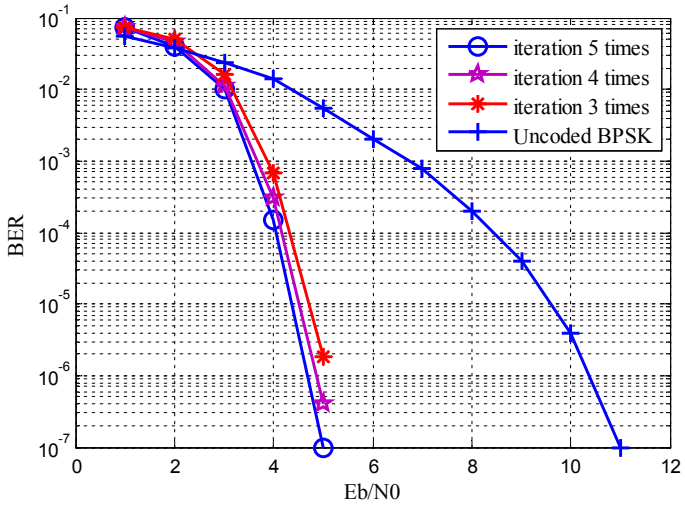

Figure 3. The influence of the iteration times in the TPC Decoding.

3.3. Effect of the Unreliable Positions

In the iterative decoder, in order to maximum likelihood decode the extended Hamming code (n,k,δ), the correlation value between the received code word R and 2^p possible code words should be calculated. So, the decoding complexity is increased exponentially with p. It is hard to physically realize when $p>6$. The following simulation shows the effect of p, when the quantization bits are constrained to 8 and the iteration time is set to 3. Simulation results indicated that p greatly affect the decoding performance, and increasing p can increase the decoding performance while the decoding complexity is also increased. When $p<3$, increasing it can greatly increase the decoding performance, and when $p>3$, increased p can not greatly improve the decoding performance.

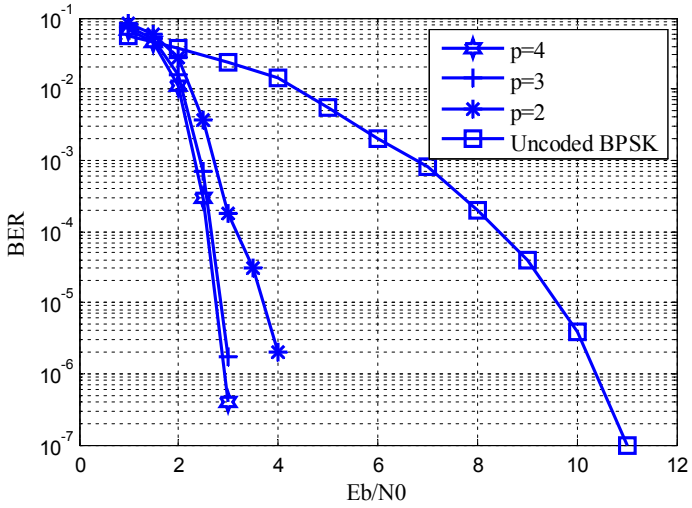

Figure 4. The influence of the numbers of unreliable positions in the TPC decoding.

Figure 4 shows the BER performance of the TPC decoding where 8 quantization bits, 3 iterations and 2 unreliable positions are used. When aiming at (64,57,4) TPC, simulation results confect that the code gain is more than 6.2dB at BER=10^{-6}, and if the unreliable positions is increased to 3 the code

gain can be 6.8dB. The code gain can be improved if the iteration time and unreliable positions are increased, but it is too difficult to implement in engineering.

4. TPC in FPGA

4.1. TPC Encoding in FPGA

Four steps which include the data receiving, buffering and pre-processing, row encoding, column encoding, framing, data rate adjusting and output are needed to implement the TPC encoding in FPGA. The encoding is discussed in many references before and this paper will focus on the decoding.

4.2. TPC Decoding in FPGA

A decoder includes four parts: the data receiving and buffering, iteration parameters and data controlling, SISO decoding and the data buffering and output, among which, the SISO decoding is the most complex part and consumes the most resources. Ping-Pong technique is adopted between data receiving and buffering and data buffering and output, which can makes these two parts parallel the SISO decoding.

In the decoding process, the pre-decoding data sent to the decoder is 8 bits quantized in frame. Firstly, the received data is Ping-Pong buffered by the decoder. Secondly, once a frame of data is filled, the decoder begins to decode this frame. Thirdly, when the iterative decoding is finished, hard decision is implemented, and the data is Ping-Pong outputted. The first frame is decoded completely. In the process of the row decoding and the column decoding, the main flow is almost similar, the difference is only exist in some parameters.

4.2.1. SISO Decoding Module. SISO is the key part of the TPC decoder, and it is used to decode the receiving data with soft-input and soft-output and to calculate and out the extrinsic information $[W(m)]$. A whole SISO is depicted in figure 5. The purpose of decomposing the SISO decoder into several modules is to increase the decoding efficiency by using the pipeline technologies.

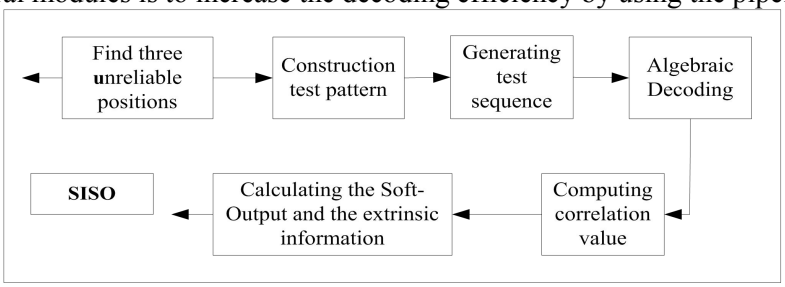

Figure 5. The process of the SISO

a) Module of Finding the Unreliable Positions. This module is used to find *p* data with minimum absolute values in a serially inputted row of soft data while output the current row of data after a hard decision. In order to saving clocks, *p* unreliable positions should be found in a circle. Finding the *p* unreliable positions in a row is the basis of generating the testing sequence and algebraic decoding.

b) Module of Constructing the Test Pattern and Generating the Test Sequence. Construct *q* test patterns based on the *p* unreliable positions produced in the previous module. The way of construct the test pattern is depicted as follows. Firstly, make the *p* unreliable positions 0 and 1 respectively, and then exclusive or with the hard decision code word. The test sequence should be saved until the decision code C_C and the competitive code C_D is get and the middle variable related to the soft output and the extrinsic information.

c) Module of Algebraic Decoding

Based on the algebraic decoding rule of the extended Hamming codes, this module finds the error bit according to the value of the check bit. Then, the error bit is reversed.

d) Module of Computing Correlation Value

Hardware resources, such as multipliers, are used to calculate the Euclid distance of different test sequences and received sequences. The calculating of Euclid distance is simplified to calculating of

correlation value. Both the cycle accumulation and the binary trees adder can be used to calculate the correlation value, the binary trees adder saves the resource of clock but lots of middle value will be imported and then consumes lots of hardware resources. Thus, cycle accumulation is used to calculate the correlation value in our work.

e) Calculating of the C_C *and* C_D. The sequence corresponding to the maximum correlation value is used as the C_C, and then the C_D is achieved. The correlation values C_C and C_D will be used in the subsequent calculating of the soft output and extrinsic information. Looking for the C_D is the hardest part in hardware implementation since it consumes lots of hardware resources due to the pipeline technique. Figure 6 shows the pipeline simulation.

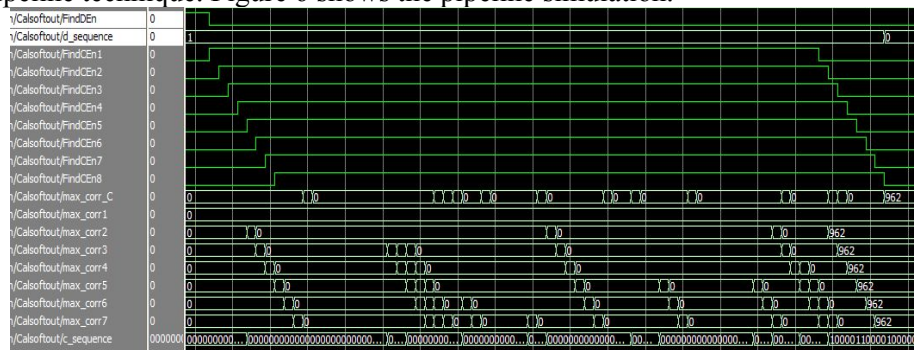

Figure 6. Pipeline simulation figure of the competitive code calculating.

f) Module of Calculating the Soft-Output and the Extrinsic Information. While calculating the soft-out, it must firstly look up the competitive code corresponding to each soft-out. If the competitive code exists, then compute the soft-out based on the correlation value, extrinsic information value, and if the competitive code doesn't exist, substitute the soft-out with the reliable factor β.

4.2.2. Iteration Control. Iteration Control, which is used to implement the data transfer and iteration parameter control, is the most important part of the TPC decoder. In decoding, the α is decided by the iteration time, and get the $r + \alpha[W(m)]$ by the internal signal composition module, and the composited signal is sent to the SISO module. Because there are row SISO and column SISO in decoding, in the next half iteration, the data should be read according to the row address and the column address. When the row decoding is finished, the memory should be written in row while the memory should be read in column in column decoding, and vice versa.

The module of iteration control is an opened interface in the program, where the iteration time can be easily modified by changing a few parameters. The iteration is a serialized process, and multi-iteration is realized by multiple calling of the same module in different times. So, in theory, changing the iteration time will not change the resources used by the program, and only the input data rate will be affected. Generally, the less iteration time the much higher decoding rate, and the less coding gain. If the iteration time is reduced, the pre-decoding data rate will be reduced accordingly, and the code gain will be enhanced.

5. Conclusions

Soft-input and soft-output is adopted by the TPC, the code gain can achieve 6.8dB when the BER=10^{-6}, and it can be implemented in satellite communications and deep space communications. The Ping-Pong technique and pipeline technique in this work can greatly alleviate the time delay and improve the decoding rate. In practice, a decoding rate of 20Mbps can be achieved by using a Xinlinx XC5VLX85-3FF676 FPGA chip. In summary, the TPC has a good prospect in engineering.

6. References

[1] X M Wang, G Z Xiao 2001 *Error Correction Code--Principle and method [M]*. (Xian：Xidian University Publisher House)

[2] D H Liu, G M Liang 2011 *Turbo Code Design and Application [M]*. (Beijing: Electronic industry Publisher House)
[3] S Lin, M Fossorier 1998 *Soft input soft output decoding of linear block codes based on ordered statistics [J]*.(Global Telecom. Conference,2828-2833)
[4] X F Wd, A N Akansu 2001 *On parallel iterative decoding of product code [J]*. (VIC 2001. IEEE VTS 54 7-11, Oct.2001)
[5] F P Wen. 2010 *Research and Application of Turbo Product Code Decoding Algorithm Based on Chase Algorithm [D]*, (Chengdu：University of Electronic Science and Technology of China 2010.5)
[6] X Luo. 2007 *Turbo Product Code Technology Research and its FPGA Implementation [D]*. (Xian：Northwestern Polytechnical University 2007.3)