Journal of Engineering and Sustainable Development
Vol. 22, No. 4, July 2018
ISSN 2520-0917
https://doi.org/10.31272/jeasd.2018.4.11

PERFORMANCE COMPARISON OF PROPOSED INTERLEAVER WITH DIFFERENT TYPES FOR PARALLEL TURBO CODE

*Amjad Ali Jassim [1], Dr. Wael A.H. Hadi [2]

1) Ph. D, Electrical Engineering Department, Technology of University, Baghdad, Iraq.
2) Assist. Prof., Communication Engineering Department, Technology of University, Baghdad, Iraq.

Abstract: Interleaver type is important in parallel turbo code system. It's responsible of improve code performance against additive white Gaussian noise AWGN. In this paper, introduce a new interleaver with its mathematical model. Study its performance as comparison with different interleavers, including random interleaver, QPP quadratic primitive polynomial interleaver as standard of LTE, chaotic interleaver and finally 2-D interleaver for burst error environment, Study the performance as simulation. The system takes in consideration also different modulation schemes including QPSK, 16-QAM and 64-QAM. Result comparison between different types summarized in performance BER curves and Tables.

Keywords: *Chaotic Interleaver, 2-D interleaver, Proposed Interleaver, QPP Interleaver, Turbo code*

مقارنة اداء مداخل مقترح مع نوعيات أخرى للمشفر الفائق المتوازي

الخلاصة: ان نوع المداخل المستخدم مع المشفر الفائق المتوازي ذو اهمية كبيرة، فهو ذو مسؤولية عن تحسين اداء المشفر مع تأثير الضوضاء AWGN. في هذه الورقة البحثية، نقدم نوعية جديدة من المداخل مع الموديل الرياضي. دراسة الاداء الخاص به كمقارنة مع نوعيات مختلفة اخرى متضمنة النوع العشوائي و QPPالذي يمثل النوع النموذجي لتقنيات التطوير بعيد المدى، ونوع المشوش chaotic واخيرا النوع ثنائي الابعاد المستخدم مع الاخطاء من نوع burst. ان دراسة الاداء تمت ضمن برمجيات تحليلية، ان النظام المستخدم في الدراسة اخذ بنظر الاعتبار ايضا انواع مختلفة للتضمين وهي QPSK, 16-QAM and 64-QAM. ان نتائج المقارنة بين الأنواع المختلفة تم تلخيصها بمخططات أداء BER والجداول.

1. Introduction

Interleaver one of the most important component that construct Turbo code, it plays major role in enhance overall Turbo code performance against burst errors. in communication applications interleaver design effects system performance, many researches interest in interleaver design and focus on its performance against burst errors, complexity design and time consuming in both interleave and deinterleave data. Interleaver applied in many communication system applications, for examples we could see in wireless half-duplex [1] interleaver gain important for receive diversity schemes of distributed Turbo codes in such subject interleaver design established to be important factor in DTC distributed Turbo code performance [1].

In [2] introduce interleaver design that provide facilities in high throughput Turbo decoding which is required for next generation wireless systems. Interleaver in wireless

* comm_engtec80@yahoo.com

communication [3] applied in division multiple access IDMA with multiple users in wireless communication systems, the chip interleaver here represent the only means of user separation. Interleaver continue take its location in more applications even for OFDM system used to improve BER by using random interleaver [4]. Also for forward error correcting codes FEC the Block interleaver used as main burst error treatment in OFDM based WIMAX (IEEE 802.16d) system [5]. As see, interleaver still present in modern communication system applications and here not about to list all its application but just a brief look where its applied, for coding with FEC and Turbo codes and also brings attention in [6] to estimate Interleaver period for Reed-Muller coded signals .In this paper a study of different interleavers with parallel turbo code system are introduce, paper divided as sections illustrate each used interleaver apart, finally introduce the proposed interleaver in separated section. More information on used system parameters where highlight with considered different modulation schemes. The simulation results listed at the end of paper as comparison between different interleaver types and different modulation schemes ware used in system simulation.

2. Considered System in Simulation

2.1. *System Transmitter*

The considered system encoder takes parallel turbo code system as encoder rate 1/3, with two identical convolutional encoders of maximum free distance of d_{free} = 5, generated polynomials (5, 7) defined in octal system number and constraint length of 3. The interleaver located as shown in Figure 1 in between to convolutional encoder. This interleaver type will changed as case study. Then followed by modulation process takes one of QPSK, 16-QAM or 64-QAM type. The choice of modulation scheme is done also according case study.

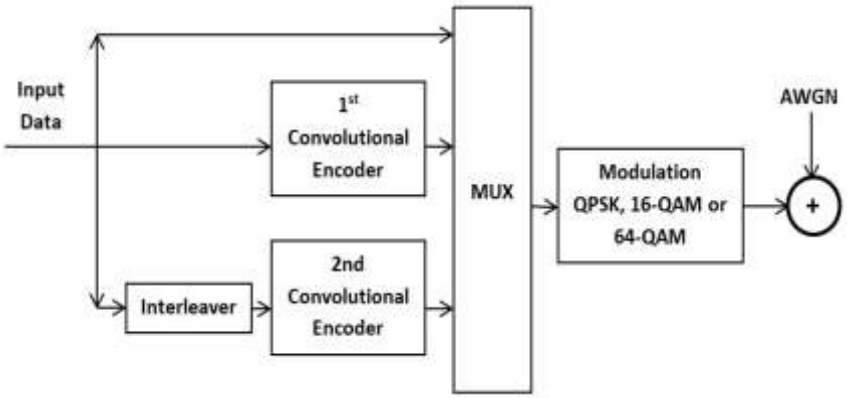


Figure 1. System transmitter.

2.2. *System Receiver*

At receiver side a reciprocal of transmitter operation done. It starts by demodulation process and then iterative decoder of parallel turbo code system with 6 iterations.

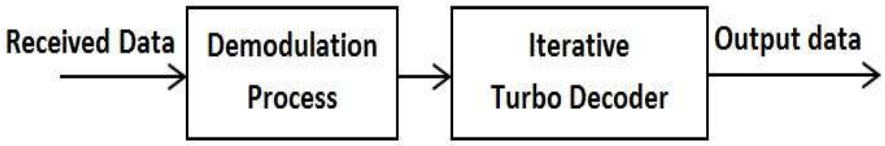


Figure 2. System receiver

3. *Random Interleaver*

It represents the simplest type of interleaver, which permutes the input vector randomly according initial seed. It may give different output sequence at each run time since it randomly rearranges the input elements sequence depends on initial seed. So reciprocal operation means restore the original data sequence. This process should be use the same initial seed used in random interleaving process. The restore data sequence called deinterleaver. So deinterleaver processes actually represent the inverse of interleaver process [7].

4. *QPP Quadratic Primitive Polynomial*

Quadratic permutation polynomial interleaver represents the standard interleaver used with parallel turbo code encoder system for LTE long term evolution. Its permutation operation depends on mathematical formula describes the new sequence of permutated data. The QPP interleaver formula in standard of LTE support frame length of 188 different values, from smallest of 40 to 6144 as largest value. The QPP formula:

$$P(i) = (f_1.i + f_2 \, . \, i^2) \bmod (k) \qquad (1)$$

Where k is input frame length, f_1 and f_2 are standards constants depends on k length summarized in reference [2], "i" is the index or sequence position of original location and finally $P(i)$ gives the new position of permuted data. The deinterleaver process at receiver side do reciprocal operation by tracing permuted data frame and restore it to its original location.

5. *Chaotic Interleaver*

Chaotic interleaver is derived from Baker map [8]. It could be described as method to randomize the bit sequence arranged in two dimensions' square matrix. Chaotic interleaver may be really referred as a concept how to sectaries element in square matrix and rearrange them. its mathematical formula may be illustrated as let $B(n_1, n_2 \ldots . n_k)$ denotes the discretize map. Where the vector $[n_1, n_2 \ldots . n_k]$, represent the sector *key*. S_{key} . Defining N as the number of data items in one row, the secret key is chosen in way such that each integer n_i divides N data items with condition:

$$n_1 + n_2 + \ldots\ldots n_k = N \qquad (2)$$

Let $N_i = n_1 + n_2 + \ldots\ldots + n_{i-1}$ the data items at indices (r, s) is moved to new indices [8]-[13]:

$$B(r,s) = \left[\frac{N}{n_i} (r - N_i) + s \bmod \left(\frac{N}{n_i}\right), \frac{n_i}{N} \left(s - s \bmod \left(\frac{N}{n_i}\right) \right) + N_i \right] \quad (3)$$

$$where\ N_i \leq r < N_i + n_i, 0 \leq S < N, and\ N_1 = 0.$$

To illustrate this equation in steps, the chaotic interleaver process listed in following points [3]:

1. An N×N square matrix is divided into N rectangles of width n_i and number of elements N.
2. The elements in each rectangle are rearranged to a row in the permuted rectangle. Rectangles are taken from left to right beginning with upper rectangles then lower ones.
3. Inside each rectangle, the scan begins from the bottom left corner towards upper elements.

To understand previous steps listed in [8], for example

A simple square matrix of 8×8 which mean 64 elements as shown in Figure (3) below at (a), (b) and (c) . The *key* should be selected by designer to satisfy chaotic interleaver condition, that divide the square matrix to rectangles each of elements equal to N. here in this example N=8. So the selected *key* = {2, 4, 2}. The rectangle borders are shown in heavy bold lines.

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

(a) Original data sequence

Key= {2, 4, 2}

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

(b) Key segments

31	23	15	7	32	24	16	8
63	55	47	39	64	56	48	40
11	3	12	4	13	5	14	6
27	19	28	20	29	21	30	22
43	35	44	36	45	37	46	38
59	51	60	52	61	53	62	54
25	17	9	1	26	18	10	2
57	49	41	33	58	50	42	34

(c) Chaotic interleaver

Figure 3. Chaotic interleaver key segments on 8×8 square matrixes.

In case study an extended to chaotic interleaver including square matrix of 16×16 with elements of 256.

The designed *key* such that satisfy chaotic interleaver conditions is calculated to be *key* = {2, 2, 4, 4, 2, 2} with each inside square matrix a rectangle with number of elements equal to N=16.

See Figure (4).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112
113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144
145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160
161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176
177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192
193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208
209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224
225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240
241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256

Figure 4. Chaotic interleaver key segments on 16×16 square matrixes.

Simulation parameters include input frame length equal to 1024 bit. This leads to design at first a square matrix with dimension of 32×32. Contain input data arranged in such matrix. Next step is to design *key* divide the square matrix with rectangles each of number of elements equal to N=32. The designed *key* = {2, 2, 2, 2, 4, 4, 4, 4, 2, 2, 2, 2}. Then new sequence of input data read row by row to construct chaotic interleaved frame of length 1024 bit.

6. 2D Two Dimensional Prime Interleaver

This type of interleaver works on two-dimension matrix. It really remaps the matrix element sequence in manner depends on mathematical formula set the new positions of interleaved data sequence according calculated row and column vectors.

Row index

Column index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
3	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48
4	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64
5	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80
6	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96
7	97	98	99	100	101	102	103	104	105	106	107	108	109	110	111	112
8	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127	128
9	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144
10	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159	160
11	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175	176
12	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191	192
13	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208
14	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223	224
15	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239	240
16	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255	256

Figure 5. 2D 16×16 Prime interleaver input data matrix before interleaving

The two shaded row and column represent the original location index of each element at the matrix. Recall that 2D prime interleaver calculate new location index [14]. Depends on following mathematical formula:

Let P_{row} and P_{col} two design parameters and n_r, n_c number of elements in row and

column respectively. The new location of column wise and row wise calculated from:

Row-wise	Column-wise
$1 \rightarrow 1$	$1 \rightarrow 1$
$2 \rightarrow (1+P_{row}) \bmod n_r$	$2 \rightarrow (1+P_{col}) \bmod n_c$
$3 \rightarrow (1+2\ P_{row}) \bmod n_r$	$3 \rightarrow (1+ 2\ P_{col}) \bmod n_c$
$4 \rightarrow (1+3\ P_{row}) \bmod n_r$	$4 \rightarrow (1+ 3\ P_{col}) \bmod n_c$
…..	…..
$n_r \rightarrow (1+(n_r-1)\ P_{row}) \bmod n_r$	$n_c \rightarrow (1+ (n_c-1) P_{col}) \bmod n_c$ (4)

So for case 16×16 matrix, let design parameter P_{row} =13, P_{col} = 11. Row-wise and Column-wise calculated as follows:

Row-wise	Column-wise
1→1	1→1
2→(1+1*13) mod 16 = 14	2→(1+1*11) mod 16 = 12
3→(1+2*13) mod 16 = 11	3→(1+2*11) mod 16 = 7
4→(1+3*13) mod 16 = 8	4→(1+3*11) mod 16 = 2
5→(1+4*13) mod 16 = 5	5→(1+4*11) mod 16 = 13
6→(1+5*13) mod 16 = 2	6→(1+5*11) mod 16 = 8
7→(1+6*13) mod 16 = 15	7→(1+6*11) mod 16 = 3
8→(1+7*13) mod 16 = 12	8→(1+7*11) mod 16 = 14
9→(1+8*13) mod 16 = 9	9→(1+8*11) mod 16 = 9
10→(1+9*13) mod 16 = 6	10→(1+9*11) mod 16 = 4
11→(1+10*13) mod 16 = 3	11→(1+10*11) mod 16 = 15
12→(1+11*13) mod 16 = 16	12→(1+11*11) mod 16 = 10
13→(1+12*13) mod 16 = 13	13→(1+12*11) mod 16 = 5
14→(1+13*13) mod 16 = 10	14→(1+13*11) mod 16 = 16
15→(1+14*13) mod 16 = 7	15→(1+14*11) mod 16 = 11
16→(1+15*13) mod 16 = 4	16→(1+15*11) mod 16 = 6

This calculated new index values will be used to read from original data sequence and remapped to interleaved matrix. See the Figure (6) below.

	1	14	11	8	5	2	15	12	9	6	3	16	13	10	7	4
1	1	12	7	2	13	8	3	14	9	4	15	10	5	16	11	6
12	209	220	215	210	221	216	211	222	217	212	223	218	213	224	219	214
7	161	172	167	162	173	168	163	174	169	164	175	170	165	176	171	166
2	113	124	119	114	125	120	115	126	121	116	127	122	117	128	123	118
13	65	76	71	66	77	72	67	78	73	68	79	74	69	80	75	70
8	17	28	23	18	29	24	19	30	25	20	31	26	21	32	27	22
3	225	236	231	226	237	232	227	238	233	228	239	234	229	240	235	230
14	177	188	183	178	189	184	179	190	185	180	191	186	181	192	187	182
9	129	140	135	130	141	136	131	142	137	132	143	138	133	144	139	134
4	81	92	87	82	93	88	83	94	89	84	95	90	85	96	91	86
15	33	44	39	34	45	40	35	46	41	36	47	42	37	48	43	38
10	241	252	247	242	253	248	243	254	249	244	255	250	245	256	251	246
5	193	204	199	194	205	200	195	206	201	196	207	202	197	208	203	198
16	145	156	151	146	157	152	147	158	153	148	159	154	149	160	155	150
11	97	108	103	98	109	104	99	110	105	100	111	106	101	112	107	102
6	49	60	55	50	61	56	51	62	57	52	63	58	53	64	59	54

Figure 6. 2D 16×16 Prime interleaver output data matrix after interleaving.

In simulation parameter, 2D prime interleaver is used with matrix of dimension 32×32 with elements of 1024 bit. $P_{row} = 3$, and $P_{col} = 5$.

7. Proposed Interleaver

The proposed interleaver works according predefined step size from designer. The step size takes as reference in interleaving process. Such to insure at most as possible a step distance between two successive elements. For the interleaver input data frame of length N, the interleaving takes also in consideration the input frame length to calculate new position of each element in input data frame. Let examine a simple case. For input frame length of N=10 elements. And simple designed step size of 1. Then the sequence gets jump by step size equal to one, and continue set the new positions, after reaches the position of N=10. It returns to empty position and record the new sequence and so on. See Figure (7):

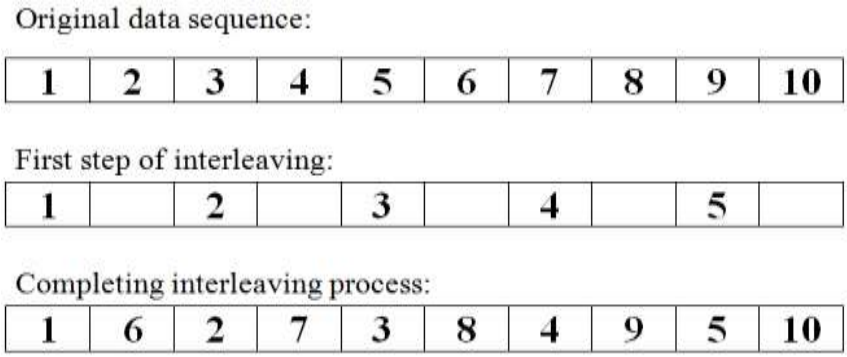


Figure 7. Proposed interleaver with step size of 1. Frame length N = 10.

But what is really happened at different step size? It takes the same steps used for single step size. For other simple example same frame length N=10. And step size of 3. See the illustrated steps shown in Figure (8):

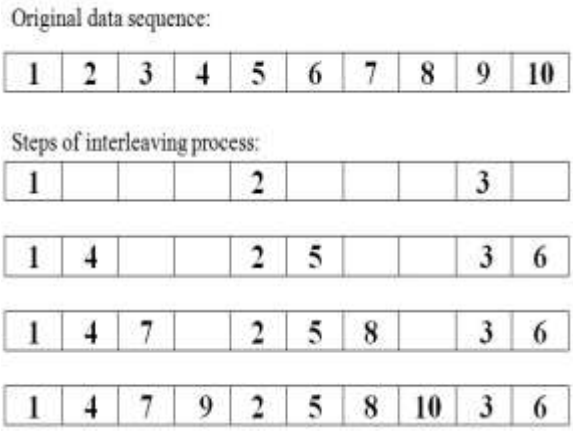


Figure 8. Proposed interleaver with step size of 3. Frame length N = 10.

As mentioned above. The proposed interleaver sensitive to frame length N. to see its effect, let increasing input frame length only by one element, such that N=11. So the steps of proposed interleaver as shown in Figure (9):

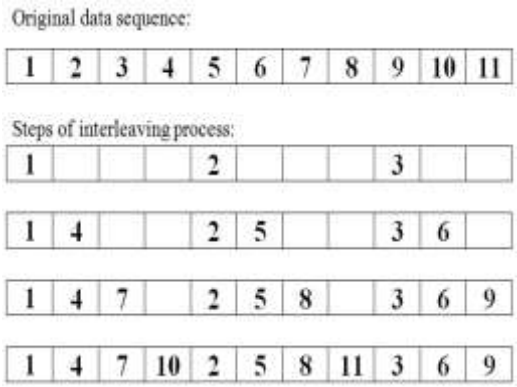
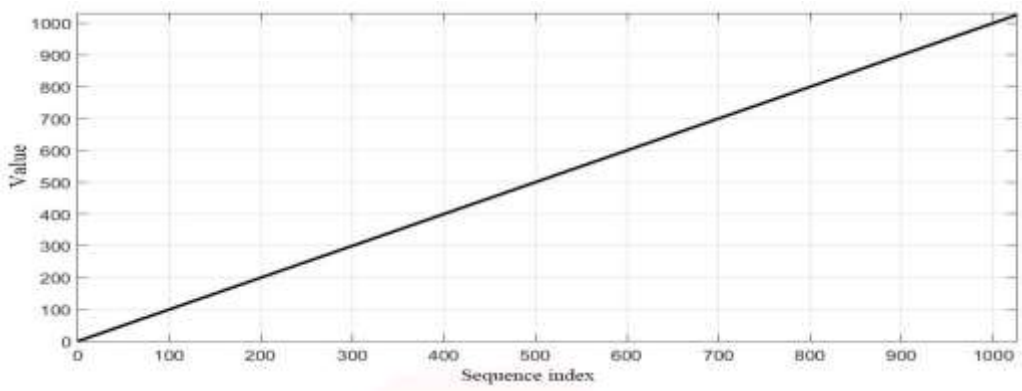


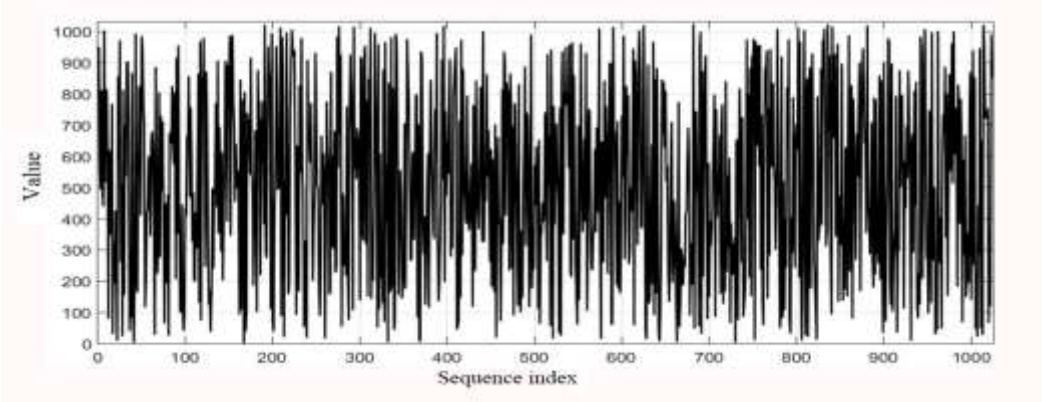
Figure 9. Proposed interleaver with step size of 3. Frame length N = 11.

From previous two examples, it's simple to note that for example element 9 change its position from 4^{th} to 11^{th}, element 10 change its position from 8^{th} to 4^{th}, and new element 11 set at 8^{th} position.

Review for previous presented four types of interleavers represented by Random, QPP, Chaotic and 2D prime interleaver. Look for interleaver index graph. It's clear the difference between previous presented type and proposed interleaver. See figure (10). List graphs of index of each type of interleaver. The graphs take number of elements of 1024 bit. The graphs show the distribution of new sequences after interleaving.

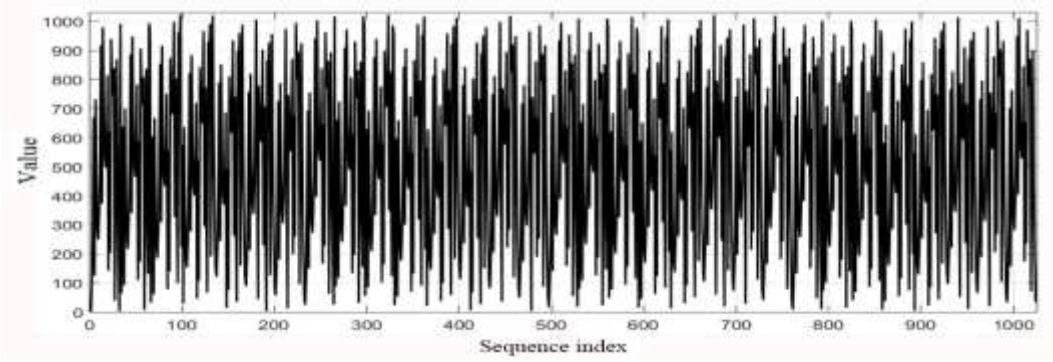


a- Original data sequence from one to 1024.

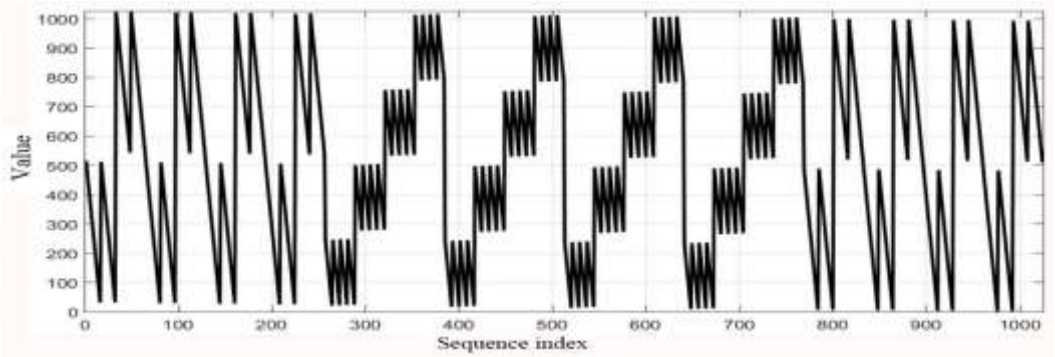


b- Random interleaved data sequence.

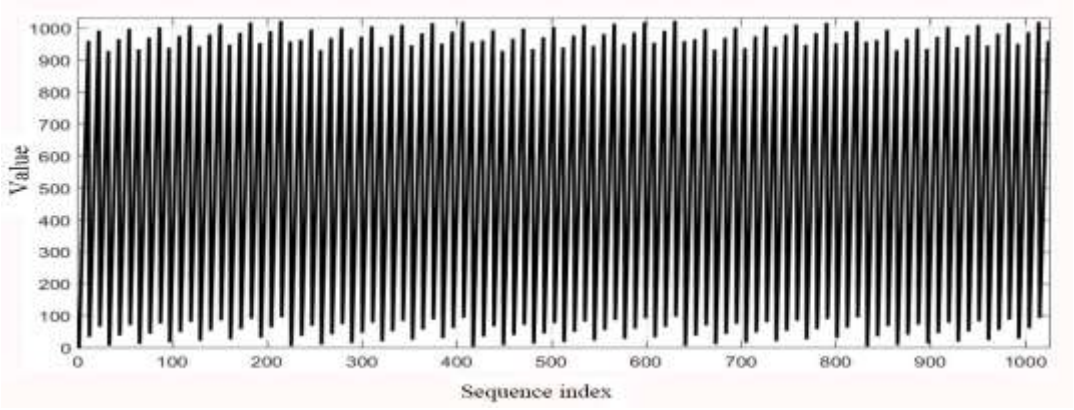
Figure 10. different interleavers' data sequence pattern comparison.



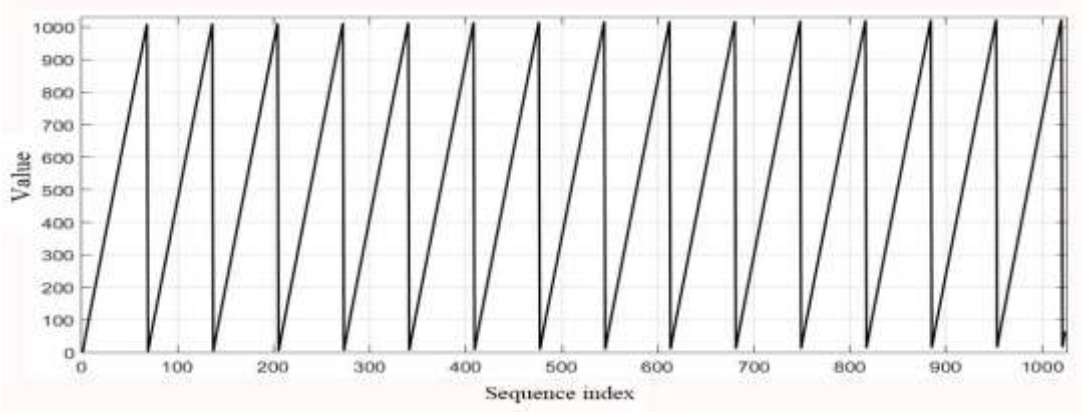
c- QPP interleaved data sequence.



d- Chaotic interleaved data sequence.



e- 2D prime interleaved data sequence.



f- Proposed interleaved data sequence.

Figure 10. Continued

7.1. *Mathematical Model of Proposed Interleaver*

Now express the proposed interleaver *mathematical model*. The behavior of proposed interleaver discussed in worked examples. Let consider the same previous example for frame length N=10, and step size = 3.

The derived mathematical module for proposed interleaver expressed in following equations:

$$P(i) = \begin{cases} P(1) = 1 \\ P_{n+1}(i) = P_n(i) + step + 1 \quad , P_{n+1}(i) =< \text{Frame length} \\ P_{n+1}(i) = P(1) + 1 + \delta_n \quad , P_{n+1}(i) > \text{Frame length} \\ \text{Update } \delta_{n+1} = \delta_n + 1 \, , \; \delta_0 = 0 \text{ initial at zero} \end{cases} \quad (5)$$

Where P(i) is the resulted calculated new index after interleaved. P_{n+1} are the new calculated position. P_n is previous calculated value. Let apply the calculation:

P(1) = 1.
P(2) = P(1) + (step size = 3) + 1 = 1+4 = 5.
P(3) = P(2) + 3+ 1 = P(2) + 4 = 5+4 = 9.
P(4) = P(3) + 3+1 = P(3) + 4 = 9 + 4 = 13 > frame length N
Then:
P(4) = P(1) + 1 + (δ_0 = 0) = 1+1 = 2.
Update $\delta_1 = \delta_0 + 1 = 0 + 1 = 1$.
P(5) = P(4) + 3 + 1 = 2 + 4 = 6.
P(6) = P(5) + 3 + 1 = 6 + 4 = 10.
P(7) = P(6) + 3 + 1 = 10 + 4 = 14 > frame length N
Then:
P(7) = P(1) + 1 + (δ_1 = 1) = 1 + 1 + 1 = 3.
Update $\delta_2 = \delta_1 + 1 = 1 + 1 = 2$.
P(8) = P(7) + 3 + 1 = 3 + 4 = 7.
P(9) = P(8) + 3 + 1 = 7 + 4 = 11 > frame length N
Then:
P(9) = P(1) + 1 + (δ_2=2) = 1 + 1 + 2 = 4.
Update $\delta_3 = \delta_2 + 1 = 2 + 1 = 3$.
P(10) = P(9) + 3 + 1 = 4 + 4 = 8.

The worked example shows the same new index of proposed interleaver. refer to Figure (11)

Original data sequence:

1	2	3	4	5	6	7	8	9	10
1	4	7	9	2	5	8	10	3	6

Figure 11. Worked example results sequences.

8. Simulation Results Discussion

Simulation take parallel turbo code system based on maximum free distance convolutional encoder of generator polynomials in octal defined by (5, 7) and constraint length of 3, free distance equal to 6. Frame length of 1024 bit and total input frames of 1000. The proposed interleaver with a step size of 67. The 2D interleaver takes primes 3 for row and 5 for column respectively. QPP interleaver define f_1 of 31 and f_2 of 64 this according standard tables for frame length 1024 bit. Chaotic interleaver designed for 32×32 matrix that produce 1024 interleaved elements. The simulation studies the system performance with three modulation types QPSK, 16-QAM and 64-QAM. For simulation results see figures (12-14):

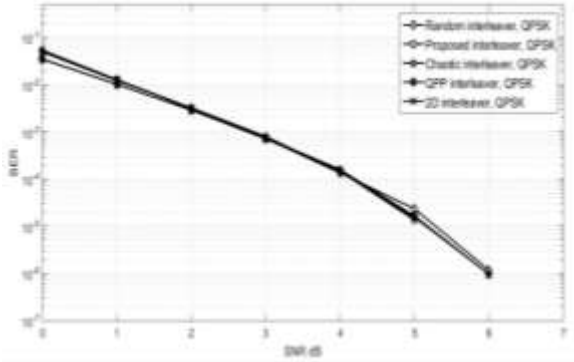


Figure 12. Turbo code system performance, QPSK modulation with different interleaver types.

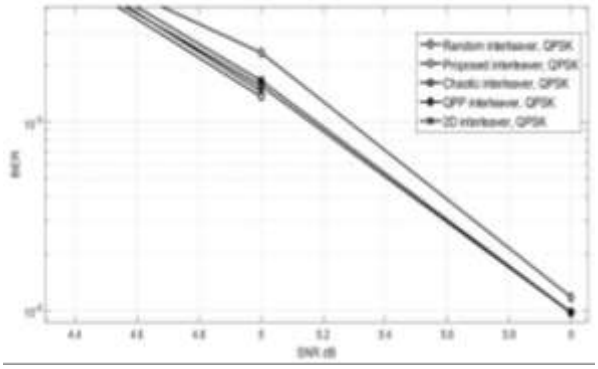


Figure 13. Turbo code system performance, QPSK modulation with different interleaver types enlarged view at curves end.

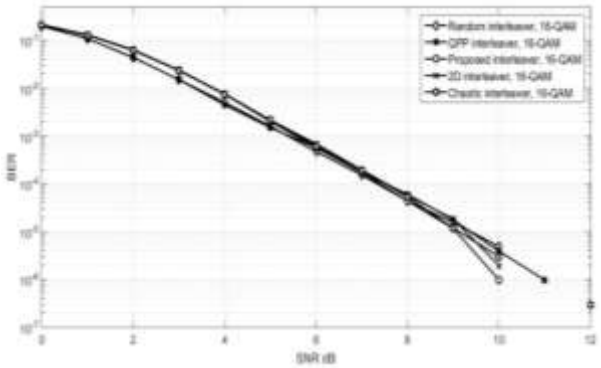


Figure 14. Turbo code system performance, 16-QAM modulation with different interleaver types.

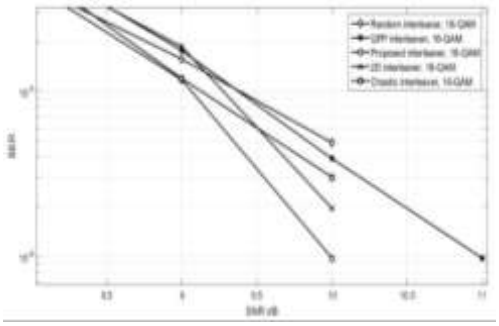


Figure 15. Turbo code system performance, 16-QAM modulation with different interleaver types enlarged view at curves end.

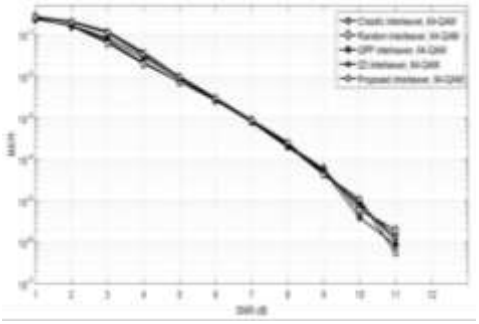


Figure 16. Turbo code system performance, 64-QAM modulation with different interleaver types.

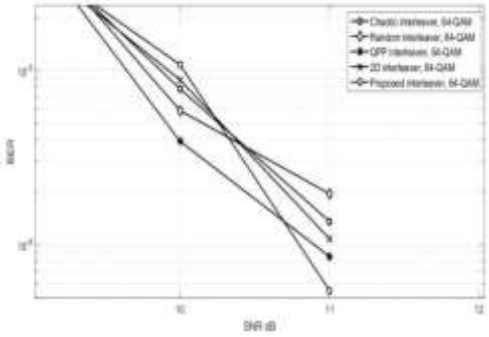


Figure 17. Turbo code system performance, 64-QAM modulation with different interleaver types enlarged view at curves end.

Table 1. Simulation results comparisons

Modulation Type	Interleaver Type	SNR dB	BER
QPSK	Random	6	1.1766×10^{-6}
QPSK	Chaotic	6	9.701×10^{-7}
QPSK	2D	5	1.4648×10^{-5}
QPSK	QPP	6	9.7656×10^{-7}
QPSK	Proposed	5	1.3672×10^{-5}
16-QAM	Random	10	4.8828×10^{-6}
16-QAM	Chaotic	10	3.0156×10^{-6}
16-QAM	2D	10	1.9531×10^{-6}
16-QAM	QPP	11	9.7746×10^{-7}
16-QAM	Proposed	10	9.7656×10^{-7}
64-QAM	Random	11	1.9531×10^{-6}
64-QAM	Chaotic	11	1.3533×10^{-6}
64-QAM	2D	11	1.0766×10^{-6}
64-QAM	QPP	11	8.531×10^{-7}
64-QAM	Proposed	11	5.4332×10^{-7}

9. Conclusions

In this paper, introduce a new type of interleaver used with parallel turbo code depends on predefined step size. Discuss its operation and put a mathematical model to describe its operation in data interleaving. Also recall four types of interleavers, discuss briefly its operation and the way used by each one to interleave data with examples. The simulation also studies the system performance with different modulation including QPSK, 16-QAM and 64-QAM, simulation results show improvement in BER with using proposed interleaver. It's important to note that long term evolution LTE interleaver represented by QPP need standard fixed values of 188 pair define constants of f_1 and f_2, while the proposed interleaver not needs such predefined constants with BER performance listed in Table 1 for each case. This leads to reduce interleaver complexity with accepted BER performance as shown in simulation results.

10. References

1. Tamer H. M. SOLIMAN (2015). "*Interleaver Gains for Receive Diversity Schemes of Distributed Turbo Codes in Wireless Half-Duplex Relay Channel*", RADIOENGINEERING, Vol. 24, No.2, pp. 481-488.
2. Prabhavati D. Bahirgonde (2015)." *BER Analysis of Turbo Code Interleaver",* International Journal of Computer Applications (0975-8887), Vol. 126, No. 14, pp.1-4.
3. Farheen Begum. (2016)." *Implementation of Interleaver Division Multiple Access (IDMA) with Multiple Users in Wireless Communication System*". International Journal of Computer Applications (0975-8887), Vol. 134, No. 15, pp.6-10.
4. Quanitah Shaikh. (2015)." *Improvement in BER Performance of OFDM System using Interleave*r". International Journal of Science, Engineering and Technology Research (IJSETR). Vol. 4, Issue 2, pp. 382-385.
5. Arun Agarwal. (2015)." *Combined Effect of Block Interleaver and FEC on BER Performance of OFDM based WIMAX (IEEE 802.16d) System*". American Journal of Electrical and Electronics Engineering. Vol. 3, No. 1, pp. 4-12.
6. Yeonsoo Jang. (2016) "*Estimation of Interleaving Period for Reed-Muller Coded Signals*". International Journal of Future Computer and Communication, Vol. 5, No.2, pp.116-119.
7. Vineet Chaturvedi, Vivek Kumar Gupta. (2012). "*Performance Analysis for Different Interleavers in Various Modulation Schemes with OFDM over an AWGN Channel*" IOSR Journal of Engineering, Vol.2, No.4, pp: 760-767.
8. Mohsen A. et at "*An Efficient Chaotic Interleaver for Image transmission over IEEE 802.15.4 Zigbee Networ*k" journal of telecommunications and information technology. pp 67-73.
9. A. N. Lemma, J. Aprea, W. Oomen, and L. V. de Kerkhof, (2003). "*A temporal domain audio watermarking technique*", IEEE Trans. Sig. Process. Vol. 51, No. 4, pp. 1088–1097.
10. W. Li, X. Xue, and P. Lu. (2006) "*Localized audio watermarking technique Robust against time-scale modification*", IEEE Trans. Multim. Vol. 8, No. 1, pp. 60–69.

11. G. Voyatzis and I. Pitas. (1998). "*Chaotic watermarks for embedding in the spatial digital image domain*", in *Proc*. IEEE Int. Conf. Image Process. Vol. 2, pp. 432–436.
12. R. Liu and T. Tan. (2002). "*An SVD-based watermarking scheme for protecting rightful ownership*", IEEE Trans. Multim., Vol. 4, No. 1, pp. 121–128.
13. Z. Liu and A. Inoue. (2003). "*Audio watermarking techniques using sinusoidal patterns based on pseudorandom sequences*", IEEE Trans. Circ. Sys. Video Technol., Vol. 13, No. 8, pp. 801–812.
14. P. Hanpinitsak and C. Charoenlarpnop parut . (2013). "*2D Interleaver Design for Image Transmission over Severe Burst-Error Environment*" International Journal of Future Computer and Communication, Vol. 2, No. 4, pp. 308-312.