PERANCANGAN DAN IMPLEMENTASI TURBO ENCODER PADA LTE BERBASIS FPGA

DESIGN AND IMPLEMENTATION OF TURBO ENCODER FOR LTE BASED ON FPGA

Ni'matul 'Abdah A.F.[1], Rina Pudji Astuti[2], Denny Darlis[3]
[1,2]Fakultas Teknik Elektro, Universitas Telkom, Bandung
[3]Fakultas Ilmu Terapan, Universitas Telkom, Bandung
[1]afakhriy@gmail.com, [2] rinapudjiastuti@telkomuniversity.ac.id , [3]denny.darlis@tass.telkomuniversity.ac.id

Abstrak

Teknik pengkodean kanal merupakan salah satu cara meningkatkan *Quality of Service* (QoS) dengan mengkodekan informasi yang dikirimkan, sehingga dapat mengurangi tingkat kesalahan informasi yang diterima di penerima. Pada transmisi jumlah data yang besar, teknik pengkodean kanal yang mendekati sempurna (mendekati teorema Shannon limit), selain *Low Density Parity Code* (LDPC) adalah *turbo code*. Berdasarkan penelitian sebelumnya, *turbo code* dirancang dengan *code rate* $^1/_3$ tanpa memenuhi proses *Transport Channel* (TrCHs) sesuai standar *3rd Generation Partnership Project* (3GPP).

Pada tugas akhir ini dilakukan perancangan *prototype Turbo Encoder* yang digunakan pada *Transport Channel* (TrCHs) pada teknologi *3rd Generation Partnership Project Long Term Evolution* (3GPP LTE) menggunakan bahasa pengkodean *VHSIC Hardware Description Language* (VHDL). Model *Turbo Encoder* yang dirancang dan diimplementasikan memiliki 40 bit *input* (*input* ditentukan dengan minimum *code block* = Z) yang sudah dijumlahkan dengan bit *Cyclic Redudancy Check* (CRC) yang akan diproses pada dua bagian *Recursive Systematical Convolutional* (RSC) dengan *code rate* $^1/_3$ dan menggunakan *interleaver* berjenis *Quadratic Permutation Polynomial* (QPP). Kemudian digabungkan dengan blok *rate matching* yang terdiri dari *sub- block interleaver, bit collection*, serta blok *bit selection and pruning*. Kemudian desain tersebut diimplementasikan pada *board Field Programmable Gate Array* (FPGA).

Hasil implementasi menunjukkan menunjukkan bahwa perancangan *prototype Turbo Encoder* dapat dilakukan pada *board* FPGA. Hasil sintesis mengenai penggunaan *resource* pada FPGA untuk *Turbo Encoder* adalah *slice registers* 1 % , *slice* LUTs 2%, *occupied slices* 6%, *bonded* IOBs 25%, BUFG/BUFGMUXs 43%, MUXCYs 1%, RAMB16BWERs 5%, RAMB8BWERs 1%, *fully used LUT-FF pairs 39%,* dan BSCANs 25%.

Kata kunci : Turbo Code, LTE, 3GPP, FPGA, VHDL

Abstract

To improve the Quality of Service (QoS) can use channel coding technique which encoding the information that will be transmitted, so it can reduce the error rate of the information received in the receiver. In the large amount of transmission data, channel coding technique that nearly perfect (approaching the Shannon limit theorem) other than Low Density Parity Code (LDPC) is turbo code. Based on previous research, the turbo code with code rate $^1/_3$ designed without complying with the Transport Channel (TrCHs) at 3rd Generation Partnership Project (3GPP) standart.

In the final project is to design a prototype Turbo Encoder which used in the process of Transport Channel on 3GPP LTE technology using VHSIC Hardware Description Language (VHDL) coding language. Design and implementation of Turbo Encoder is assumed have 40-bit input (input is determined by the minimum code block = Z) that have been added by the Cyclic Redudancy Check (CRC) bit which will be processed in two parts Recursive Systematical Convolutional (RSC) with code rate $^1/_3$ and use Quadratic Permutation Polynomial (QPP) interleaver. Then combined with rate matching block which consists of a sub-block interleaver, bit collection, and bit selection and pruning. And then the design is implemented on Field Programmable Gate Array (FPGA) board.

The result of the implementation showed that the design of a prototype Turbo Encoder can be performed on the FPGA board. The synthesis results of the resource usage in the FPGA for Turbo Encoder are 1 % slice registers , 2% slice LUTs , 6% occupied slices, 25% bonded IOBs, 43% BUFG/BUFGMUXs, 1% MUXCYs, 5% RAMB16BWERs, 1% RAMB8BWERs, 39% fully used LUT-FF pairs, and 25% BSCANs.

Keywords: Turbo Code, LTE, 3GPP, FPGA, VHDL

1 Pendahuluan

Peningkatkan *Quality of Service* (QoS) diperlukan dalam perkembangan teknologi saat ini. Perkembangan teknologi saat ini seperti *Long Term Evolution* (LTE) membutuhkan *bandwidth* yang cukup besar, tetapi *resource bandwidth* yang ada sangat terbatas, sehingga diperlukan penanganan *bandwidth* secara efisien. Untuk meningkatkan *Quality of Service* (QoS) salah satunya adalah dengan mengurangi tingkat kesalahan yang diterima pada *receiver*. Salah satu bentuk rekayasa yang dilakukan untuk mengurangi kesalahan bit pada penerima adalah dengan mengkodekan informasi yang akan dikirimkan atau dikenal dengan sebutan teknik pengkodean kanal (*channel coding*). Teknik *channel coding* yang diterapkan pada *Long Term*

Evolution (LTE) adalah *convolutional coding* dan *turbo coding* yang terletak pada *Transport Channels* (TrCHs). Teknologi sebelumnya yaitu *Global System for Mobile* (GSM) hanya menggunakan *convolutional coding* pada *Traffic Channels* (TCHs) dan *Control Channels* (CCHs) yang lebih rendah *data rate*-nya dibandingkan dengan LTE[2]. Perbandingan antara *convolutional coding* dan *turbo coding* terletak pada *reliability* dan *efficiency* pada proses transmisi di teknologi LTE [1]. Teknik pengkodean *turbo code* pada LTE dapat diterapkan pada perangkat di sisi *transmitter* maupun *receiver*. Pada sisi *transmitter*, perangkat ini disebut dengan *turbo encoder*.

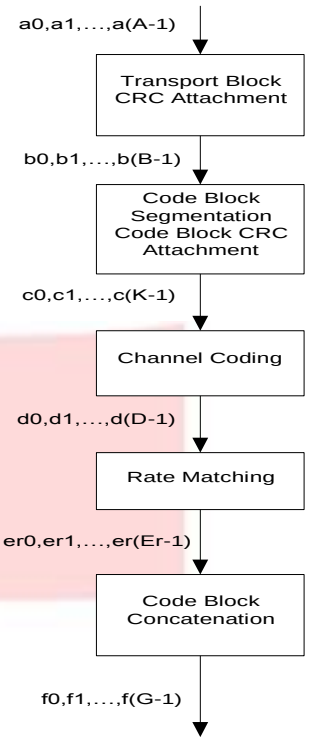
Dari penjelasan di atas, dilakukan sebuah perancangan *Turbo Encoder* yang mengacu pada standar *3rd Generation Partnership Project* (3GPP) [2] yaitu pada *Downlink Share Channel* (DL-SCH), *Paging Channel* (PCH) dan *Multicast Channel* (MCH). Pada penelitian sebelumnya yang dilakukan oleh Gooru Santosh, DR. S. Rajaram[6] telah berhasil mendesain dan mengimplementasikan *Turbo Coder for LTE on FPGA*. Pada penelitian lain yang telah dilakukan oleh K. Kalyani, A. Skahti Amutha Vardhini, S. Rajaram[5] juga telah berhasil mendesain dan mengimplementasikan *Turbo Decoder* untuk standar LTE. Akan tetapi kedua penelitian tersebut belum memenuhi proses *Transport Channel* di LTE.

Berdasarkan pemaparan di atas, maka pada tugas akhir ini diusulkan untuk dirancang sebuah *prototype Turbo Encoder* sesuai dengan proses *Transport Channel* di LTE sehingga diperoleh desain elektronik digital yang berbasis *software* atau *Very Large Scale Integration* (VLSI) *chip design*. *Software* yang digunakan dalam perancangan tugas akhir ini adalah Xilinx ISE Design Suite 14.5 dan menggunakan bahasa pengkodean VHSIC *Hardware Description Language* (VHDL). Setelah *prototype* sistem berhasil dirancang, selanjutnya ditanamkan pada *board* FPGA ATLYS Spartan-6 XC6SLX45 CSG324C.

Pada *paper* ini, bagian pertama dijelaskan tentang latar belakang masalah tugas akhir ini dibuat. Lalu bagian kedua dijelaskan tentang teknik pengkodean kanal di LTE. Kemudian pada bagian ketiga memaparkan tentang model perancangan *Turbo Encoder* yang akan diimplementasikan pada FPGA. Lalu pada bagian ke-empat akan dijelaskan mengenai hasil dari implemetasi *Turbo Encoder* di FPGA. Bagian akhir yaitu ke-lima menyatakan tentang kesimpulan dari tugas akhir ini.

2 Teknik Pengkodean Kanal pada LTE

Teknik pengkodean kanal pada LTE terjadi pada *downlink* dan *uplink* transmisi data, transmisi *paging*, serta transmisi *broadcast multicast*. Pada *transport channel* di sisi *downlink*, *turbo code* digunakan pada *Downlink Share Channel* (DL-SCH), *Multicast Channel* (MCH), dan *Paginng Channel* (PCH) [7]. Berikut ini gambar proses *transport channel* pada DL-SCH, MCH,dan PCH :



Gambar 1. *Transport Channel and Control Informasi diadaptasi dari [7]*

Pada proses *input transport blok CRC attachment*, data *a* diberi *parity* menggunakan *cyclic generator polynomial* (L=24)[3] :

$$g_{CRC24A}(D)=[D^{24}+D^{23}+D^{18}+D^{17}+D^{14}+D^{11}+D^{10}+D^{7}+D^{6}+D^{5}+D^{4}+D^{3}+D+1] \quad (1)$$

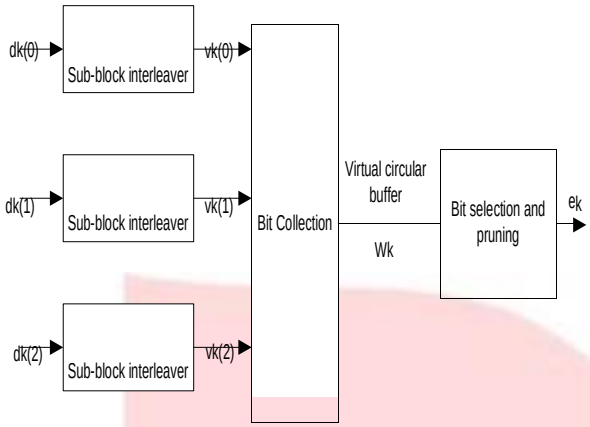
Sehingga menghasilkan bit B = A+L. Ukuran maksimal *code block* adalah Z = 6144. Apabila lebih dari ukuran maksimal *code block*, maka perlu disegmen tiap per *code block* menggunakan *cyclic generator polynomial* (L=24)[3] :

$$g_{CRC24B}(D)=[D^{24}+D^{23}+D^{6}+D^{5}+ D+1] \quad (2)$$

Sehingga menghasilkan bit *c* sebagai masukan *channel coding*. Kemudian *bit* yang melewati *channel coding* di-*generate* menjadi bit *d* . Pada skema *channel coding transport channel* di LTE terdapat 2 jenis yaitu *turbo code* dan *tail biting convolutional coding* dengan *code rate* 1/3 [3].

Pada proses *rate matching*, hasil *channel coding* didefinisikan ke dalam *sub-block interleaver*, kemudian hasilnya diproses dalam blok *circular buffer* yang dikumpulkan, diseleksi, dan di-*pruning* menghasilkan bit e_k. Setelah itu bit e_k diproses dalam blok *code block concatenation* menghasilkan *codeword* f_k yang akan diproses di bagian *physical channel* sebelum ditransmisikan ke radio frekuensi[3][7].

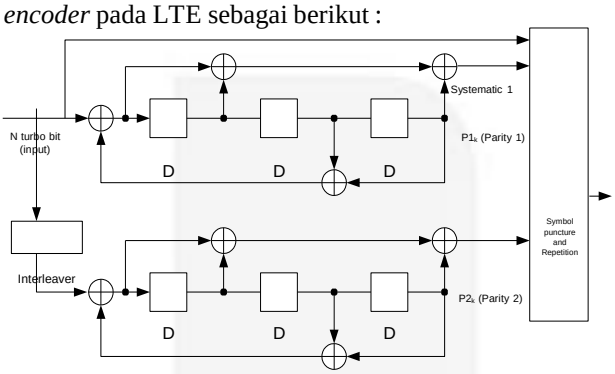
Berikut ini gambaran proses *rate matching* :



Gambar 2. *Rate Matching for transport channel diadaptasi dari [3]*

2.1. Turbo Encoder

Skema *turbo code* pada standar LTE merupakan *Parallel Concatenated Convolutional Code* (PCCC) dengan 2 komponen *encoder* 8-*state* dan sebuah *quadratic permutation polynomial* (QPP) *interleaver*. *Coding rate* dari *turbo encoder* adalah $^1/_3$ [3]. Struktur dari *rate* $^1/_3$ *turbo encoder* pada LTE sebagai berikut :



Gambar 3. Struktur *turbo encoder* dengan *rate* $^1/_3$ diadaptasi dari [3]

Pada dasarnya *coding rate* 1/3 dapat diartikan sebagai data N bit yang akan dikodekan menjadi data 3N bit. Fungsi transfer dari komponen 8-*state* dari PCCC adalah sebagai berikut[3] :

———

(3)

(4)

(5)

Nilai inisial dari *shift register* dari komponen *encoder* 8-*state* harus semua nol, ketika memulai untuk mengkodekan bit masukan. Terminasi trellis dilakukan dengan mengambil *tail bits* dari umpan balik *shift register* setelah semua bit informasi dikodekan [3].

2.1.1. Internal Interleaver

Internal interleaver memiliki 2 kandidat pada LTE yaitu *Almost Regular Permutation* (ARP) dan *Quadrature Permutation polynomial* (QPP), dimana keduanya sangat mirip. Namun QPP dipilih LTE, karena menawarkan lebih *paralelisme* untuk *turbo coding* dan untuk menghindari konflik saat *run time* [8]. Berikut persamaan QPP *interleaver* sesuai dengan [3] yaitu:

$$\Pi(i) = (f_1.i + f_2.\, i^2\,).\mathrm{mod}\ K \qquad (6)$$

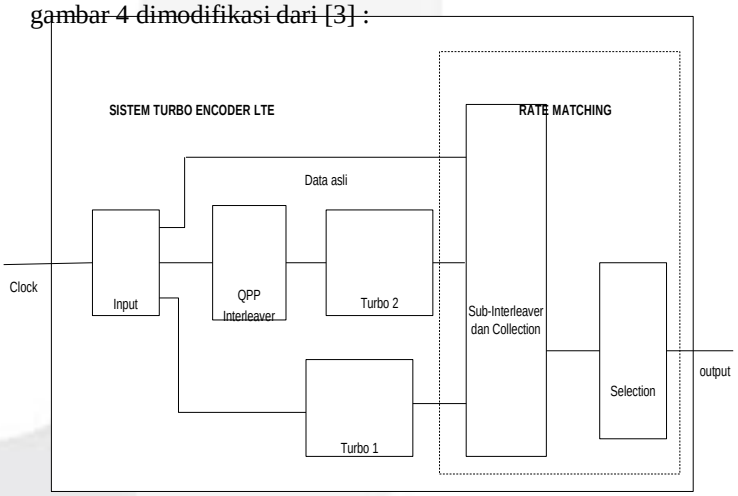
3 Perancangan *Turbo Encoder* Pada LTE

Proses perancangan *Turbo Encoder* diawali dengan menentukan spesifikasi pada standar LTE 3GPP [3].

Tabel 1 Spesifikasi Turbo Encoder LTE[3][7][8]

Spesifikasi	Keterangan
Code rate	1/3
Transport Channel	DL-SCH,MCH,PCH
Interleaver	QPP Interleaver K=40
Input + CRC	40 bit (minimum *code block*)
Pengelompokkan bit	Penggunaan *Circular Buffer*
Output Turbo Encoder	54 bit

Model sistem *Turbo Encoder* yang diimplementasikan pada tugas akhir ini digambarkan seperti gambar 4 dimodifikasi dari [3] :



Gambar 4. Struktur *Turbo Encoder* dispesifikasikan oleh 3GPP diadaptasi dari [3]

Pada gambar 4, data masukan ditentukan dengan minimum *code block* (Z) = 40 bit (yang sudah dijumlahkan dengan CRC gCRC24A) yaitu "10101010101010111011 11100110010011000011", kemudian diproses dalam tiga *stream* sesuai *code rate* 1/3 (setiap satu bit direpresentasikan ke dalam tiga bit).

3.1. Pemodelan QPP *Inteleaver*

Berikut ini permutasi QPP *Interleaver* dengan K=40 sesuai dengan [3] :

Tabel 2. Permutasi QPP *Interleaver* dengan K=40

Urutan Bit Awal	Urutan Bit Setelah Permutasi QPP
<0,1,2,3,4,5,6,7,8,9,10,11,12 ,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>	<0,13,6,19,12,25,18,31,24,3 7,30,3,36,9,2,15,8,21,14,27, 20,33,26,39,32,5,38,11,4,17, 10,23,16,29,22,35,28,1,34,7 >

3.2. Pemodelan Rate Matching

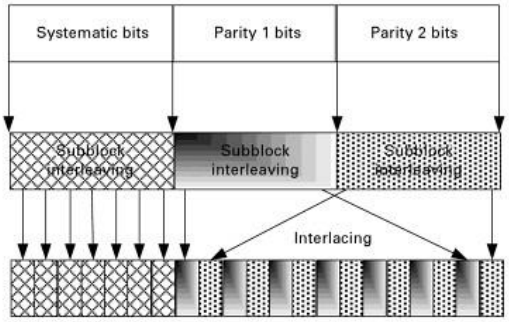
3.2.1. Sub-Interleaver and Bit Collection

Pada tabel 3. Menunjukkan permutasi pada sub-interleaver dalam 32 kolom (berlaku kelipatan bernilai sama) sesuai dengan [3] :

Tabel 3. Permutasi *Sub-Interleaver*

Number of Columns	*Inter-Columns Permutation Pattern* <P(0),P(1),…,P
32	<0,16,8,24,4,20,12,28,2,18,10,26,6,22,14,30, 1,17,9,25,5,21,13,29,3,19,11,27,7,23,15,31>

Kemudian hasil permutasi dibaca per baris sesuai dengan urutan kolom[3] dan dikumpulkan dengan metode *circular buffer* sesuai dengan gambar 5.



Gambar 5. *Sub-block Interlacing* diadaptasi dari [3]

3.2.2. *Bit Selection*

Masukan blok *bit selection* merupakan hasil dari konsep *virtual circular buffer* menggunakan *redundancy versions* (RVs) pada implementasi HARQ *Rate Matching* LTE, dijelaskan pada gambar 6.

Gambar 6. Komposisi dari *circular buffer* pada LTE *turbo code rate matching*. Sel berwarna putih, merah, dan biru berurutan digunakan untuk aliran *Systematic*, *Parity* 0, dan *Parity* 1 dibaca per baris. Untuk sel warna hijau ditandai untuk *starting points* dari RVs pada RM. Indeks RVs terdiri dari 0,1,2 atau 3. Hal ini dilakukan apabila ditransmisikan gagal, maka akan dilakukan pengiriman ulang per RVs [3].

Kemudian hasil keluaran *bit selection* pada *rate matching* mengikuti aturan 3GPP [3] yaitu e_k yang diseleksi dari $W_{(k0+j) \bmod Ncb} \neq$ <NULL>, maka

$$e_k = W_{(ko+j) \bmod Ncb} \quad (7)$$

dimana :

ko = _____ .rv_{idx}+2) (8)

Ncb= min(___ (9)

NIR= _____ =

_____

C= 1 (dalam perancangan hanya 1 *number of code block*)

Dalam perancangan *virtual circular buffer* pada blok *Selection* didapatkan :

ko = 2(2. ___ = 4, untuk indeks RVs= 0

ko = 2(2. ___ = 52, untuk indeks RVs= 1

ko = 2(2. ___ = 100, untuk indeks RVs= 2

ko = 2(2. ___ = 148, untuk indeks RVs= 3 .

Sehingga hasil blok *Selection* sebagai berikut :

$e_0 = W_{(4+8) \bmod 192} = W_{(12)}$
$e_1 = W_{(4+9) \bmod 192} = W_{(13)}$
$e_2 = W_{(4+10) \bmod 192} = W_{(14)}$

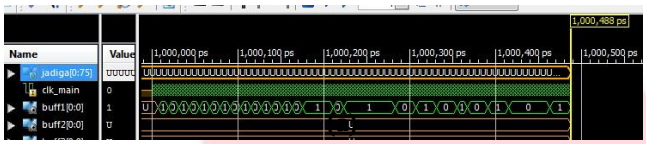
$e_{12} = W_{(4+20) \bmod 192} = W_{(24)}$
$e_{18} = W_{(4+32) \bmod 192} = W_{(36)}$
$e_{19} = W_{(4+34) \bmod 192} = W_{(38)}$
$e_{20} = W_{(4+36) \bmod 192} = W_{(40)}$
$e_{21} = W_{(4+38) \bmod 192} = W_{(42)}$
$e_{22} = W_{(4+40) \bmod 192} = W_{(44)}$
$e_{23} = W_{(4+42) \bmod 192} = W_{(46)}$
$e_{24} = W_{(4+44) \bmod 192} = W_{(48)}$
$e_{25} = W_{(4+46) \bmod 192} = W_{(50)}$
$e_{26} = W_{(4+48) \bmod 192} = W_{(52)}$
$e_{27} = W_{(4+50) \bmod 192} = W_{(54)}$
$e_{28} = W_{(4+52) \bmod 192} = W_{(56)}$
$e_{29} = W_{(4+54) \bmod 192} = W_{(58)}$
$e_{30} = W_{(52+56) \bmod 192} = W_{(108)}$
$e_{31} = W_{(52+57) \bmod 192} = W_{(109)}$

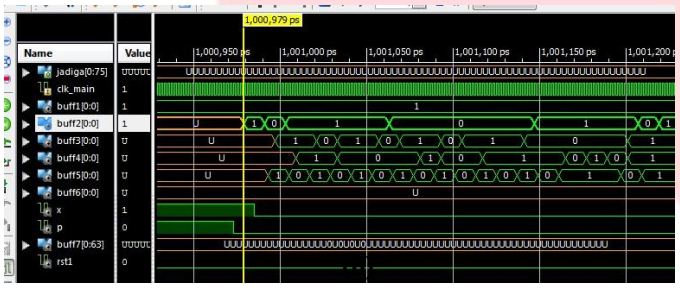
$e_{35} = W_{(52+61) \bmod 192} = W_{(113)}$
$e_{36} = W_{(52+64) \bmod 192} = W_{(116)}$

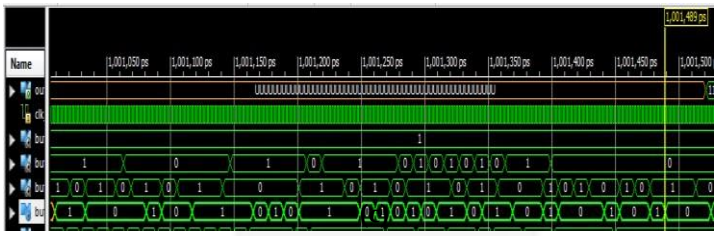
$e_{13} = W_{(4+22) \bmod 192} = W_{(26)}$
$e_{14} = W_{(4+24) \bmod 192} = W_{(28)}$
$e_{15} = W_{(4+26) \bmod 192} = W_{(30)}$
$e_{16} = W_{(4+28) \bmod 192} = W_{(32)}$
$e_{17} = W_{(4+30) \bmod 192} = W_{(34)}$
$e_{37} = W_{(52+65) \bmod 192} = W_{(117)}$
$e_{38} = W_{(52+68) \bmod 192} = W_{(120)}$
$e_{39} = W_{(52+69) \bmod 192} = W_{(121)}$
$e_{40} = W_{(52+72) \bmod 192} = W_{(124)}$
$e_{41} = W_{(52+73) \bmod 192} = W_{(125)}$
$e_{42} = W_{(52+76) \bmod 192} = W_{(128)}$
$e_{43} = W_{(52+77) \bmod 192} = W_{(129)}$
$e_{44} = W_{(52+80) \bmod 192} = W_{(132)}$
$e_{45} = W_{(52+81) \bmod 192} = W_{(133)}$
$e_{46} = W_{(52+84) \bmod 192} = W_{(136)}$
$e_{47} = W_{(52+85) \bmod 192} = W_{(137)}$
$e_{48} = W_{(52+88) \bmod 192} = W_{(140)}$
$e_{49} = W_{(52+89) \bmod 192} = W_{(141)}$
$e_{50} = W_{(52+92) \bmod 192} = W_{(144)}$
$e_{51} = W_{(52+93) \bmod 192} = W_{(145)}$
$e_{52} = W_{(52+96) \bmod 192} = W_{(148)}$
$e_{53} = W_{(52+97) \bmod 192} = W_{(149)}$

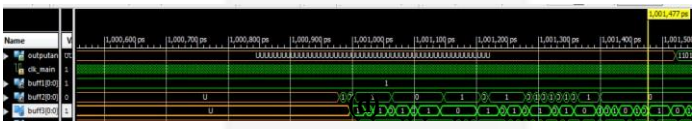
4 Pengujian dan Analisis *Turbo Encoder*

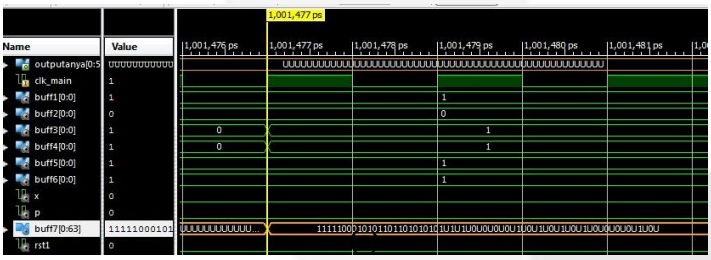
Dari hasil pengujian sistem *turbo encoder* dengan Xilinx 14.5 didapatkan hasil simulasi sebagai berikut :

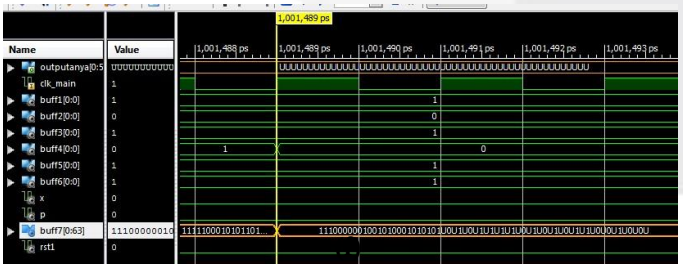


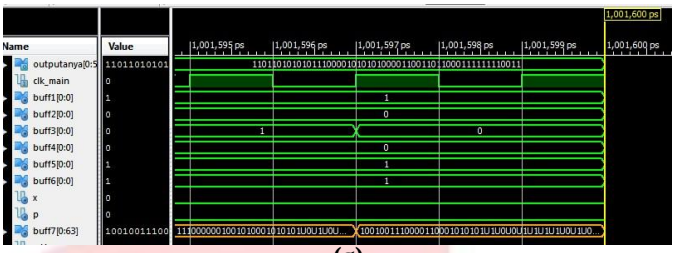




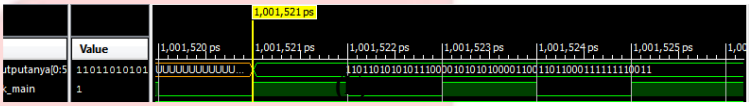


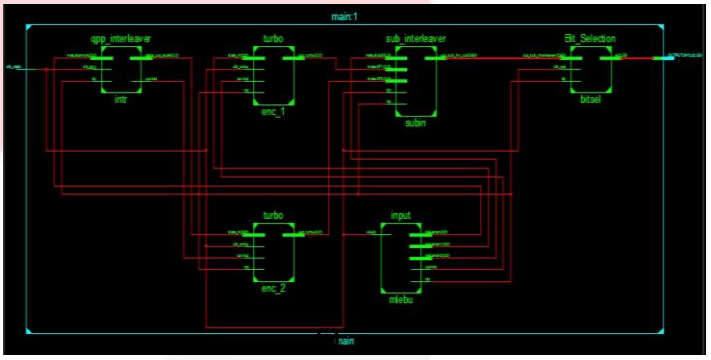


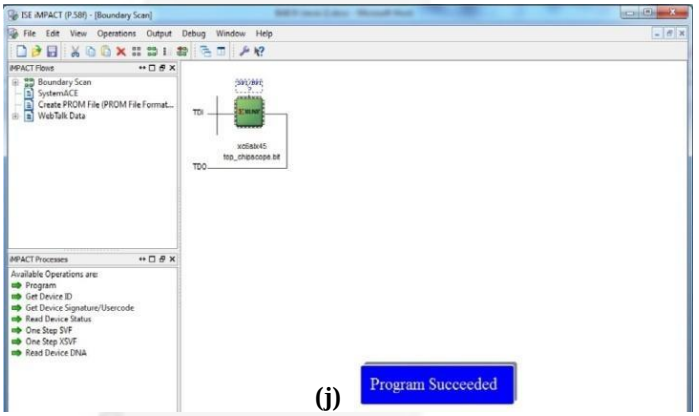




(g)







(j)

Gambar 6 Hasil Simulasi pada *ISim* (a) *Input* QPP *Interleaver* (b) *Output* QPP *Interleaver* (c) *Output* Turbo1 (d)*Output* Turbo2 (e)(f)(g) *Output Sub-Interleaver* (h) Hasil *Turbo Encoder* sebelum dimodulasikan (i) Hasil sintesis (j)Program sukses ditanamkan di FPGA

buff1 [per bit] – *input* QPP *Interleaver*
buff2 [per bit] – *output* QPP Interleaver
buff3 [per bi] – output blok *turbo* 2
buff4 [per bit] – *output* blok *turbo* 1
buff7 [per 64 bit] – *output* blok *sub-interleaver*
outputannya [54 bit] – output pemodelan *turbo encoder* pada LTE sesuai [3].

Dari gambar 6a dan 6b didapatkan hasil QPP *interleaver* pada tabel 4, sedangkan gambar 6c dan 6d menunjukkan *input output Turbo* pada tabel 5 yang sesuai dengan teori [3] yaitu :

Tabel 4. *Input* QPP dan Perbandingan *Output* QPP *Interleaver* pada *Generator Code* dengan *output* QPP *Interleaver* pada *ISim*

Input	***Output Generator Code* QPP**	***Output ISim* QPP**
10101010101010111011 11100110010011000011	10111110000000111110 11111010101011100000	10111110000000111110 11111010101011100000

Tabel 5. Perbandingan *Output* pada *Generator Code* blok *Turbo* 1 dan *Turbo* 2 dengan *Output* blok *Turbo* 1 dan *Turbo* 2 pada ISim

Blok	***Input***	***Output Generator Code***	***Output ISim***
Turbo 1	10101010101 010111011 11100110010 011000011	11000010011110 101111 01010110110010 001001	11000010011 110101111 01010110110 010001001
Turbo 2	10111110000 000111110 11111010101 011100000	11011011011100 000111 01101110110001 010010	11011011011 100000111 01101110110 001010010

Dari gambar 6e, 6f, dan 6g dihasilkan keluaran sub-interleaver sesuai teori [3] dan ditunjukkan pada tabel 6 sebagai berikut.

Tabel 6. *Input* Blok *Sub-Interleaver* dan Perbandingan *Output Generator Code* pada blok *Sub-Interleaver* dengan *Output* blok *Sub-Interleaver* pada ISim

Input *Sub-Interleaver*	**Output *Sub-Interleaver* pada *Generator Code***	**Output *Sub-Interleaver* pada ISim**
10101010101010111011 11100110010011000011	1111100010101101 101010101u1u1u0u 0u0u0u1u0u1u0u1u 0u1u0u0u0u0u1u0u	1111100010101101 101010101u1u1u0u 0u0u0u1u0u1u0u1u 0u1u0u0u0u0u1u0u
11000010011110101111 01010110110010001001	1110000001001011 001010101u0u1u0u 1u1u1u1u0u1u0u1u 0u1u1u0u0u1u0u0u	1110000001001011 001010101u0u1u0u 1u1u1u1u0u1u0u1u 0u1u1u0u0u1u0u0u
11011011011100000111 01101110110001010010	1001001110000110 001010101u1u0u0u 1u1u1u1u0u1u0u1u 1u1u1u0u1u0u0u0u	1001001110000110 001010101u1u0u0u 1u1u1u1u0u1u0u1u 1u1u1u0u1u0u0u0u

Dari gambar 6h dihasilkan keluaran blok bit selection sesuai dengan tabel 7.

Tabel 7. Perbandingan *Output Generator Code* dengan *Output* ISim pada blok *Bit Selection* yang siap ditransmisikan

Output Bit Selection* pada *Generator Code	11011010101011100001010101000011001101100011111111110011
***Output Bit Selection* pada ISim**	11011010101011100001010101000011001101100011111111110011

Pada gambar 6i menunjukkan bahwa sistem *turbo encoder* secara keseluruhan, dimana rangkaian terdiri dari 6 blok utama, yaitu *input*, QPP *Interleaver*, *Turbo*1, *Turbo*2, *Sub_Interleaver*, dan *Bit_Selection*

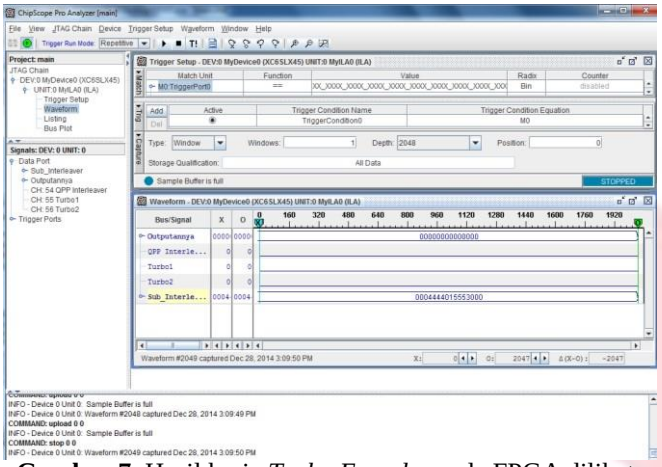
Berikut hasil sintesis sistem setelah implementasi pada Xilinx 14.5 :

Tabel 8. Hasil Penggunaan *Resource* FPGA

Slice Registers	1%
Slice LUTs	2%
occupied Slices	6%
MUXCYs used	1%
fully used LUT-FF pairs	39%
bonded IOBs	25%
RAMB16BWERs	5%
RAMB8BWERs	1%
BUFG/BUFGMUXs	43%
BSCANs	25%

Dari tabel 8 diperoleh penggunaan *Slice register* 1% dari jumlah *resource* yang tersedia. Setiap *slice* tersiri dari 2 LUT dan 2 flip- flop. Pada tabel 8 penggunaan *slice* LUTs 2 % lebih kecil dibandingkan dengan *fully used* LUT-FF 39%. Hal ini berarti desain yang telah dirancang lebih dianggap sebagai *sequential logic* daripada *combinational logic*.

Setelah dilakukan pengujian pada ISim, kemudian dilakukan pengujian pada *software Chipscope* secara *real time* yaitu ditunjukkan pada gambar 7.



Gambar 7. Hasil kerja *Turbo Encoder* pada FPGA dilihat melalui *software Chipscope*

Dari gambar 7 terlihat bahwa hasil keluaran *turbo encoder* (*Outputannya*) dalam bentuk heksadesimal dan berjumlah 54 bit sesuai dengan gambar 6h , yaitu jumlah bit keluaran sama ketika dilakukan simulasi dengan *software Isim* pada *Xilinx* 14.5, serta perbandingan keluaran antara *generator code* yang telah dijelaskan pada teori secara *manual calculation* dengan keluaran ISim sama, sehingga dapat disimpulkan bahwa *prototype Turbo Encoder* yang telah dirancang dapat diimplementasikan pada perangkat FPGA ATLYS Spartan-6 XC6SLX45 CSG324C dengan baik, namun belum sesuai dengan fungsinya. Pada pengujian ini menggunakan bantuan *clock divider* yang berfungsi untuk memperkecil *clock* sistem yang akan memperlambat laju data sehingga akan mempermudah dalam melakukan analisis, serta mendapatkan hasil yang sesuai.

5 Kesimpulan dan Saran

5.1 Kesimpulan

Kesimpulan yang diambil dari hasil perancangan, realisasi dan pengukuran pada penelitian ini adalah sebagai berikut:

1. Perancangan *prototype Turbo Encoder* berdasarkan spesifikasi LTE yaitu *code rate* 1/3 dan *Chanel Coding* yang digunakan *Turbo Coding* telah berhasil dilakukan dan diimplementasikan pada *board* ATLYS Spartan-6 XC6SLX45 CSG324C dengan membandingkan hasil implementasi dengan *generator code*- nya.
2. Hasil perancangan berhasil disintesiskan dan menghasilkan *resources Turbo Encoder* pada FPGA adalah *Number of slice registers* 1 % , *Number of slice* LUTs 2%, *Number of occupied slices* 6%, *Number of bonded* IOBs 25%, *Number of* BUFG/BUFGMUXs 43%, *Number of* MUXCYs 1%, *Number of* RAMB16BWERs 5%, *Number of* RAMB8BWERs 1%, *Number of fully used LUT-FF pairs 39%* dan *Number* BSCANs 25%. Dengan presentase *Slice LUT-FF* yang lebih besar daripada *Slice LUTs* maka *protoype* yang dirancang lebih sebagai *sequential logic* daripada *combinational logic*.

5.2 Saran

Untuk mendapatkan performansi yang lebih baik, saran untuk penelitian berikutnya antara lain

1. Diharapkan pada penelitian berikutnya dapat digabungkan antara *turbo encoder* dengan *turbo decoder* melalui dua FPGA
2. Dilakukan penelitian dengan pendekatan bentuk input yang pasti pada *Transport Channel*
3. Gunakan FPGA dengan spesifikasi yang lebih rendah, agar tidak banyak *source* yang tidak digunakan
4. Pembuatan pengkodean lebih diperhatikan dan disesuaikan dengan clock pada FPGA yang digunakan
5. Penelitian selanjutnya diharapkan harus menguasai *software* dan perangkat yang digunakan.

DAFTAR PUSTAKA

[1] Ardianto, Ahmad Eko. 2014. *Perancangan dan Implementasi Turbo Decoder Pada Teknologi LTE berbasis FPGA.*Universitas Telkom: Tidak diterbitkan.

[2] Cheng, Jung-Fu(Thomas); Ajit Nimbalker; Yufei Blankenship; Brian Classon; and T. Keith lankenship.*Analysis of Circular Buffer Rate Matching for LTE Turbo Code*.Ericsson Research,RTP,NC,USA: Motorola Labs.Schaumburg,IL.

[3] ETSI TS 136.212 version 8.8.0 Release 8. 2010. *Multiplexing and Channel Coding.*

[4] He, Suqin; Qiaozhi Hu and Haijun Zhang.2013. *Implementation of Rate Matching with Low Latency and Little Memory for LTE Turo Code*. *Journal of Information & Computational Science 10:13 (2013) 4117-4125.* China: Beijing University.

[5] Kalyani,K; Vardhini, A.Sakthi Amutha dan Rajaram,S. 'FPGA Implementation of Turbo Decoder for LTE Standard'. *Journal of Artificial Intelegence*, Vol.6, No.1, pp. 22-32, 2013.

[6] Santosh, Gooru dan DR.S. Rajaram.2013.'Design and Implementation of Turbo Coder for LTE on FPGA'. *International Journal of Electronics Signals and Sistems IJESS),ISSN:2231-5969,ol.3,Iss-2,2013.*

[7] [Online].Tersedia http://www.sharetechnote.com/html/BasicProcedure_LTE.html [25November 2014].

[8] Zarei, Shahram. 2009.Seminar *LTE Der Mobilfunk der Zukunft Channel Coding and link Adaptation*