TA/SEKJUR/TE/2017/005

IMPLEMENTASI ALGORITMA VITERBI DENGAN MENGGUNAKAN *SOFTWARE* LABVIEW

TUGAS AKHIR

Diajukan Sebagai Salah Satu Syarat

Untuk Memperoleh Gelar Sarjana Teknik Elektro

Oleh :

Nama : Henik Setyaningsih

No.Mahasiswa : 12524118

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ISLAM INDONESIA

2016

LEMBAR PENGESAHAN PEMBIMBING

TUGAS AKHIR

IMPLEMENTASI ALGORITMA VITERBI DENGAN MENGGUNAKAN SOFTWARE LABVIEW

Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Strata-1
Pada Jurusan Teknik Elektro Fakultas Teknologi Industri

Disusun oleh:

NAMA : HENIK SETYANINGSIH
NIM : 12 524 118

Sleman, Desember 2016
Menyetujui,

Pembimbing I	Pembimbing II
Dr,Eng.Hendra Setiawan, S.T., M.T.	Dwi Ana Ratna Wati, S.T., M.Eng.

PERNYATAAN KEASLIAN

Saya yang bertanda tangan di bawah ini.

Nama : Henik Setyaningsih

No.Mahasiswa: 12 524 118

Menyatakan bahwa Tugas Akhir ini adalah hasil pekerjaan saya sendiri, dan sepanjang sepengetahuan saya, tidak berisi materi yang ditulis oleh orang lain sebagai persyaratan penyelesaian studi di Universitas Islam Indonesia atau perguruan tinggi lain, kecuali bagian-bagian tertentu yang saya ambil sebagai acuan dengan mengikuti tata cara dan etika penulisan karya ilmiah yang lazim. Jika ternyata terbukti pernyataan ini tidak benar, sepenuhnya menjadi tanggung jawab saya. Maka selanjutnya saya siap untuk ditarik kembali ijazah yang telah saya terima dari Universitas Islam Indonesia.

Yogyakarta, Desember 2016

Henik Setyaningsih

LEMBAR PENGESAHAN PENGUJI

TUGAS AKHIR

IMPLEMENTASI ALGORITMA VITERBI DENGAN MENGGUNAKAN *SOFTWARE* LABVIEW

Telah dipertahankan didepan sidang penguji sebagai salah satu syarat untuk memperoleh gelar Sarjana Strata-1 Teknik Elektro

Disusun oleh:

NAMA : HENIK SETYANINGSIH
NIM : 12 524 118

Yogyakarta, Desember 2016

Tim Penguji,

Dr.,Eng.Hendra Setiawan, S.T., M.T.
Ketua

RM. Sisdarmanto Adinandra, ST., M.Sc., Ph.D.
Anggota I

Elvira Sukma Wahyuni, S.Pd., M.Eng.
Anggota II

Mengetahui,
Ketua Program Studi Teknik Elektro
Universitas Islam Indonesia

Dr.,Eng.Hendra Setiawan, S.T., M.T.

HALAMAN PERSEMBAHAN

Yang Utama Dari Segalanya…

Sembah sujud serta syukur kepada Allah SWT. Taburan cinta dan kasih sayang-Mu yang telah memberikanku kekuatan, dan membekali dengan ilmu. Atas karunia dan kemudahan yang Engkau berikan akhirnya skripsi yang sederhana ini dapat terselesaikan. Sholawat serta salam selalu terlimpahkan keharibaan Rasulullah Muhammad SAW.

Kupersembahkan karya sederhana ini kepada orang-orang yang sangat kukasihi dan kusayangi.

Sebagai tanda bakti, hormat dan rasa terimakasih yang tiada terhingga kupersembahkan karya kecil ini kepada kedua orang tuaku, Ibu Susiati dan Bapak Tas'an yang telah memberikan kasih sayang, segala dukungan, dan cinta kasih yang tiada terhingga yang tak mungkin dapat kubalas hanya dengan selembar kertas bertuliskan kata cinta dan persembahan. Semoga ini menjadi langkah awal untuk membuat Ibu dan Bapak bahagia. Untuk Ibu dan Bapak yang selalu membuatku termotivasi dan selalu menyirami kasih sayang, selalu mendoakan, dan selalu menasihatiku menjadi lebih baik. Terimakasih bu, Terimakasih Pak..

MOTTO

“Sesungguhnya bersama kesukaran pasti ada kemudaha. Dan bersama kesukaran pasti ada kemudahan. Karena itu bila selesai tugas, mulailah dengan yang lain dengan sungguh-sunguh. Hanya kepada Tuhanmu hendaknya kau berharap”

(QS. Al- Insyirah: 5-8)

Allah tidak membebani seseorang melainkan sesuai dengan kesanggupannya

(QS. Al-Baqoroh: 286)

KATA PENGANTAR

بِسْمِ اللَّهِ الرَّحْمَٰنِ الرَّحِيمِ

Assalamu'alaikum warahmatullahi wabarakatuh,

Alhamdulillahirabbil'alamin, segala syukur penulis ucapkan kehadirat Allah SWT yang telah melimpahkan segala rahmat dan karunia-Nya, sehingga penulis dapat menyeleseikan laporan tugas akhir ini, yang disusun sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Elektro Universitas Islam Indonesia. Sholawat dan salam semoga tercurah kepada junjungan kita Rasulullah Muhammad SAW beserta keluarga, dan sahabatnya yang setia hingga akhir zaman.

Suatu kebahagian, akhirnya laporan tugas akhir ini dapat terseleseikan sesuai rencana dengan begitu banyaknya kemudahan serta kekuatan yang Allah berikan melalui banyak pihak yang membantu memberikan pencerahan, dukungan dan do'a. Untuk itu penulis ingin mengucapkan banyak terima kasih kepada:

1. Kedua Orang Tua penulis Bapak Tas'an dan Ibu Susiati yang senantiasa memberikan do'a, nasihat, semangat serta ridhonya kepada penulis yang tidak pernah terputus.
2. Dr. Eng Hendra Setiawan, S.T., M.T selaku ketua Jurusan Teknik Elektro Universitas Islam Indonesia dan juga Pembimbing I tugas akhir yang telah meluangkan waktu, memberikan masukan, bimbingan, saran dan nasihat sampai terselesaikannya tugas akhir ini.

3. Ibu Dwi Ana Ratna Wati, S.T., M.Eng selaku dosen pembimbing II tugas akhir yang telah memberikan bimbingan, masukan dan arahan dalam penyusunan laporan tugas akhir.
4. Ibu Hj.Budi Astuti, Bapak Firdaus, Bapak Sisdarmanto A, Bapak Tito Yuwono, Bapak Yusuf.A.Amrulloh, Bapak Wahyudi B.P, Ibu Ida, serta seluruh dosen pengajar Program Studi Teknik Elektro UII yang telah memberikan ilmu yang bermanfaat.
5. Mba Puji, Mas Hery, Mas Dian, Mas Agus selaku staf Teknik Elektro UII yang memberikan bimbingannya dalam menempuh studi ini.
6. Beasiswa Santri Unggulan dari BPKLN yang telah membiayai penulis selama menempuh pendidikan Strata I di Universitas Islam Indonesia.
7. Sahabat serta teman penulis Bulat, Sisca, ifah, Rida, Ida, Qia, Farida, yang telah memberi bantuan, dukungan dan semangat.
8. Kawan-kawan konsentrasi Telkom yang telah memberi bantuan, dukungan dan semangat.
9. Kawan-kawan seperjuangan penulis di Teknik Elektro 2012.
10. Serta semua pihak yang telah membantu, yang tidak dapat penulis sebutkan satu persatu.

Penulis menyadari bahwa penulisan laporan tugas akhir ini masih jauh dari kesempurnaan. Oleh karena itu penulis sangat mengharapkan saran dan kritik yang membangun. Semoga tugas akhir ini dapat bermanfaat dilingkungan yang membutuhkan.

Wassalamu'alaikum warahmatullahi wabarakatuh

Yogyakarta, Desember 2016

Penulis

ABSTRAK

Implementasi algoritma Viterbi merupakan Forward error correction (FEC) yang digunakan pada transmisi data untuk mengoreksi kesalahan pada data yang dikirim kepada receiver. Pemodelan dan simulasi implementasi algoritma Viterbi merupakan hal yang perlu dilakukan karena dapat mendeteksi kesalahan data dari gangguan noise, sehingga data yang sampai ke receiver akurat. Pada Tugas Akhir ini, telah dilakukan pemodelan dan simulasi implementasi algoritma Viterbi menggunakan software LabVIEW. Hasil pengujian menunjukkan bahwa pengoreksi data menggunakan algoritma Viterbi dengan metode radix 2-butterfly mencapai hasil BER 0 pada SNR 8,6 dB dengan menggunakan model kanal AWGN.

Kata Kunci: *Forward error correction* (FEC), Algoritma Viterbi, LabVIEW.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR ISTILAH

Branch Matric	Jarak Hamming (*Hamming distance*) kode yang diterima dengan kemungkinan kode yang seharusnya.
Channel	Media transmisi telekomunikasi *wireless.*
Code rate	Perbandingan antara n bit output terhadap k bit input dalam satu waktu.
Constraint Length	Banyaknya *state* yang mempengaruhi bit keluaran pada encoder konvolusi.
Decoder	Transformasi bit-bit yang diterima menjadi rangkaian biner terduga. Strategi *decoding* bergantung pada aturan *encoding* yang digunakan.
Doppler Spread	Pergerakan relatif antara pemancar dan penerima akibat dari pergerakan objek-objek pada kanal.
Encoder	Transformasi bit-bit informasi dengan metode tertentu menjadi bit-bit terkodekan sehingga lebih tahan terhadap *noise* (derau) selama proses transmisi maupun perekaman.
Fading	Melemahnya sinyal transmisi data dalam suatu *channel*.
Interferensi	Terganggunya transmisi data oleh sinyal transmits data lain yang tidak diinginkan.
Modulasi	Proses modifikasi sinyal pembawa (gelombang elektromagnetik) berdasarkan karakteristik bit-bit informasi yang akan dikirim.

Multipath	Memantulnya sinyal transmisi ke suatu objek sebelum sampai ke *receiver*.
Path metric	Jalur yang telah dilalui *state.*
Traceback	Proses untuk memilih node *state* yang memiliki *state* terkecil dan melakukan penelusuran kembali jalur yang telah ditempuhnya.

BAB I

PENDAHULUAN

1.1. Latar Belakang Permasalahan

Pengiriman informasi pada perangkat telekomunikasi digital yang digunakan dan dimiliki setiap orang saat ini memiliki kemampuan berkomunikasi tanpa kabel atau *wireless* ataupun WLAN (*Wireless Local Area Network*) yang menjadi primadona dalam dunia telekomunikasi. Adapun perangkat telekomunikasi dengan sistem pengiriman informasi menggunakan *wireless*, terdapat kekurangan dalam waktu pengiriman data, yaitu adanya *noise* yang ada di perangkat elektronik maupun di udara. Hal ini bisa mengganggu kevalidan data pada penerima informasi sehingga perlu adanya mendeteksi dan mengoreksi data.

Pendeteksi dan pengoreksi data informasi digital dalam dunia telekomunikasi disebut dengan *forward error correction* (FEC) yang merupakan salah satu cara mendeteksi *error* yang memungkinkan penerima memperbaiki *error* secara otomatis tanpa permintaan transmisi ulang. Sandi konvolusi merupakan salah satu teknik FEC yang bisa direalisasikan. Sandi konvolusi ini, akan menghasilkan keluaran yang nilainya bergantung pada masukan informasi sebelumnya sehingga dalam implementasinya memperlukan memori.

Salah satu algoritma yang bisa menguraikan sandi konvolusi adalah algoritma Viterbi, dimana dalam tekniknya menggunakan data sebelumnya dan mengkalkulasi jarak konstelasi antar data yang berurutan (mencari jarak minimum yang memiliki nilai akumulasi *error matrix* yang terkecil) dan memperkirakan data yang paling mungkin diterima bit yang salah dapat dideteksi dan diperbaiki

sehingga tidak terjadi data *error* atau data tidak valid pada penerima informasi data.

Pada implementasi algoritma Viterbi dapat di rancang menggunakan bahasa pemprograman *tekt based* seperti Matlab, bahasa C dan model *based* seperti Simulink dan LabVIEW. Kelemahannya pada model *tekt based* yaitu *coding* berbentuk *script* yang rumit yang tidak semua orang mengetahui maksudnya, pada pemrograman model *based* sangat mudah dipahami karena berbentuk blok-blok yang mudah dimengerti dengan melihatnya. Algoritma Viterbi dapat diimplementasikan dengan *standart* 3GPP, IEEE 802.11n dan LTE (*Long Term Evolution)*. Untuk lebih memahami kinerja FEC menggunakan algoritma Viterbi, maka sangat diperlukan simulasi rancangan jenis FEC tersebut menggunakan *software* LabVIEW yang merupakan model *based*.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan di atas, maka dapat diambil rumusan masalah sebagai berikut:

a. Bagaimana pemodelan dan simulasi FEC algoritma Viterbi dengan menggunakan *software* LabVIEW ?
b. Bagaimana performansi yang dihasilkan dengan algoritma Viterbi menggunakan *software* LabVIEW ?

1.3. Tujuan Penelitian

a. Pemodelan dan simulasi FEC menggunakan algoritma Viterbi dengan menggunakan *software* LabVIEW.
b. Mengetahui performansi yang dihasilkan FEC dengan algoritma Viterbi menggunakan *software* LabVIEW.

1.4. Batasan Masalah

a. Penelitian sampai tahap simulasi.

b. Menggunakan *standart wireless IEEE 802.11.a.*

c. Model kanal menggunakan AWGN (*Addivite White Gaussian Noise*) dan model kanal Flat Fading Rayleigh.

d. Penelitian tidak melibatkan pemodelan blok lainnya pada *standart wireless IEEE 802.11a.*

e. Algoritma Viterbi menggunakan *hard decision.*

f. Menggunakan modulasi QAM (*Quadrature Amplitudo Modulation*).

1.5. Manfaat Penelitian

Manfaat yang didapatkan dari penelitian ini adalah dapat mempermudah pemahaman implementasi algoritma Viterbi dalam sistem *wireless.*

1.6. Sistematika Penulisan Laporan

Sistematika penulisan dan pembahasan laporan tugas akhir ini yaitu sebagai berikut:

BAB I PENDAHULUAN

Pada bab ini diuraikan tentang judul, latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, metodologi penelitian dan sistematika penulisan laporan.

BAB II TINJAUAN PUSTAKA

Pada bab ini diuraikan hasil penelitian sebelumnya yang telah dilakukan dan membahas mengenai teori FEC, konvolusi kode, dan algoritma Viterbi yang mendukung dalam pelaksanaan simulasi.

BAB III PERANCANGAN SISTEM

Pada bab ini diuraikan tentang perancangan implementasi algoritma Viterbi yang sesuai dengan *standart wireless IEEE 801.11a.*

BAB IV HASIL PERANCANGAN dan SIMULASI

Pada bab ini berisi tentang hasil perancangan simulasi dengan implementasi algoritma Viterbi. Hasil dari proses simulasi menggunakan *noise* dengan pengaturan yang bervariasi dengan model AWGN dan Flat Fading Rayleigh. Hasilnya akan diperoleh nilai perhitungan serta perbandingan pada SNR dan BER. Pada bab ini juga, membandingkan nilai BER dengan *traceback* pada masing-masing model kanal yang digunakan. Perhitungan dan perbandingan tersebut digunakan untuk memperoleh performansi dari sistem implementasi algoritma Viterbi.

BAB V PENUTUP

Pada bab ini dimuat kesimpulan dari penelitian yang telah terlaksana serta saran-saran yang disampaikan berdasarkan hasil simulasi yang telah dibuat.

BAB II

TINJAUAN PUSTAKA

2.1. Penelitian Sebelumnya

Penelitian tentang *forward error correction* (FEC) menggunakan algoritma Viterbi telah berhasil dilakukan pada penelitian sebelumnya seperti pada penelitian yang dikerjakan oleh Supradi[1], yaitu analisis FEC pada algoritma Viterbi menggunakan *standart IEEE 802.11a* dengan *software* Simulasi Network Simulator 2.27 (NS2.27). Kinerja utama yang diukur dalam penelitian ini adalah tingkat kemampuan metode konvolusi dalam mengoreksi kesalahan data selama *transmisi* dengan menghitung banyaknya bit yang mampu dikoreksi dari bit-bit yang salah (dari 0 menjadi 1 dan sebaliknya) dengan pembangkitan data secara acak. Hasil penelitian yaitu besar nilai BER (*Bit Error Rate*) sangat dipengaruhi oleh besarnya nilai Eb/No (*Energy Bit per Noise*) dan perbandingan jaringan pada *transmisi* penggunakan konvolusi lebih tinggi dibandingkan tanpa konvolusi. Kelemahan dalam penelitian ini adalah belum tercapainya simulasi untuk kerapatan data tinggi (hingga batas maksimum bit *rate* nya) dan belum tercapainya simulasi *channel* yang lebih kompleks dengan memperhitungkan multipath dan *fading effect*, *interferensi* dan faktor – faktor propagasi radio.

Peneliti sebelumnya juga pernah dilakukan oleh Mahyadi[2], yaitu visualisasi kinerja pengkodean dengan menggunakan bahasa C dengan Open GL dan menggunakan metode *hard decision*. Hasilnya, proses *decoding* memiliki dua mekanisme berbeda yaitu pada saat proses dari kedalaman *trellis* 0 sampai 2 dan *decoding* proses kedalaman 3 sampai seterusnya, diperoleh bahwa struktur dari

convolutional encoder sangat berpengaruh dalam menentukan proses dan pola pengkodean kembali oleh algoritma Viterbi.

Penelitian sebelumnya juga dilakukan oleh Nanang dan Yoedy[3], yaitu perbandingan kode *rate* konvolusi pada aplikasi CDMA. Bagaimana data sebelum dicoba mengalami proses *spreading* dengan menggunakan kode konvolusi pada kode *rate* 1/2 dan 1/3. Hasil penelitian yaitu data yang hanya melewati kanal AWGN membutuhkan nilai Eb/No yang lebih kecil dibandingkan jika melewati proses kode konvolusi dengan menggunakan *rate* 1/3 dan menghasilkan nilai BER yang lebih baik di bandingkan jika menggunakan *rate* 1/2, pada penelitian ini hanya perbandingan kinerja algoritma Viterbi.

Pada penelitian sebelumnya juga dilakukan oleh Eko Kuncoro Adiyanto [4], yaitu penggunakan konvolusi kode dan turbo kode. Pengkodean menggunakan turbo kode yaitu pengkodean secara *interative* dan *interleaving* tak seragam, pengkodean ini menggunakan algoritma *maximum aposteriori probability* (MAP). Hasil penelitian berupa kurva grafik perbandingan nilai Eb/No dengan penggunaan modulasi QPSK, 8 PSK, 16 QAM dan 64 QAM. Hasil perbandingan konvolusi kode dan konvolusi turbo kode pada penggunaan *channel coding* di peroleh bahwa konvolusi turbo kode mampu menghemat Eb/No sampai 5-15 dB di banding konvolusi kode, dan penggunaan modulasi QAM memiliki sistem yang mampu mengkonversi bit menjadi simbol secara lebih efektif. Pada penelitian ini hanya membahas perbandingan konvolusi kode dan turbo kode dan belum membahas simulasi *channel* yang lebih kompleks.

Penelitian sebelumnya juga dilakukan oleh Kanchana Katta[5] yaitu membahas konvolusi kode menggunakan Matlab dengan *constraint length* 4 dan

bit *rate* 1/2. Hasil penelitian yaitu grafik antara BER (*Bit Error Rate*) dan SNR dengan perbandingan *soft decisious* dan *hard decisious*, di peroleh bahwa *soft decisious* tidak dapat menggunakan *hamming distance metric* karena resolusi yang terbatas. Pada penelitian ini juga belum melakukan simulasi *channel* yang lebih kompleks.

Penelitian lainnya juga dilakukan oleh G.Sivasankar and L.Thangarani [6], yaitu membahas aplikasi kenerja Viterbi untuk komunikasi *wireless,* dimana desain encoder konvolusi dan decoder Viterbi menggunakan FPGA (*Field Programmable Gate Array*) yang disimulasikan dan di implementasikan dengan menggunakan verilog hdl dan xilinx spartan 3e kit. Pada penelitian ini, menggunakan *constraint length* 3 dan 7 serta kode *rate* ½, hal ini kompatibel dengan banyak *standart* umum seperti 3GPP, IEEE 802.16 dan LTE. Hasil yang diperoleh yaitu daya yang diperlukan pada decoder Viterbi adalah 0.0081 watt dengan Xilinx Xpower Analyzer, dengan analisa bahwa panjang bit menunjukkan semakin meningkatnya *hardware* kompleksitasnya, pada penelitian ini hanya perbandingan performansi power yang digunakan dalam *hadware* antara *contraint length* 3 dan 7 dan tidak memperhitungkan besarnya *noise*.

Dari beberapa pemaparan studi pustaka yang pernah dilakukan sebelumnya diatas, maka implementasi FEC menggunakan algoritma Viterbi dengan *software* LabVIEW merupakan hal yang menarik dan perlu dikembangkan dalam sistem pengiriman data dengan memperhitungkan *noise* dan fading, sehingga data informasi yang diperoleh lebih akurat.

2.2. *Forward Error Correction* (FEC)

Dalam teori telekomunikasi dan informasi dijelaskan bahwa FEC adalah sebuah mekanisme sistem untuk melakukan *error correction* pada data *transmisi*. Dimana pengirim menambahkan data tambahan (*redundant)* kedalam pesan/ paket data tersebut, yang juga dikenal sebagai *error correction code*. Hal ini membolehkan *receiver* melakukan deteksi dan memperbaiki *error* (dengan beberapa batasan) tanpa membutuhkan izin dari pengirim dalam menambahkan datanya. Keuntungan dari FEC[7] adalah *back-channel* tidak dibutuhkan, atau retransmisi data dapat dihindarkan, sehingga tidak menimbulkan biaya yang besar dalam kebutuhan *bandwidth*. Oleh karena itu FEC diterapkan dalam situasi dimana retransmisi sangat relatif terhadap biaya atau tidak dimungkinkan untuk melakukan retransmisi. Khususnya, informasi FEC biasanya ditambahkan pada banyak *massstronge devices* untuk melindungi dari kerusakan data yang tersimpan.

Pada FEC [8], penggunakan metode konvolusi encoder ada beberapa metode untuk menjabarkan proses penyandian konvolusi encoder, diantaranya yaitu diagram koneksi, polinom koneksi, diagram keadaan (*state* diagram), diagram pohon (*tree* diagram) dan diagram *trellis (trellis* diagram).

kode konvolusi secara garis besar dapat didefinisikan sebagai sebuah sistem kode yang memiliki informasi-informasi bit masukan yang akan dikodekan menjadi beberapa bit terkode dengan syarat bit masukan harus lebih kecil daripada keluaran bit terkode dan memiliki memori yang menyebabkan terciptanaya aturan untuk mengkodekan setiap informasi bit masukan menjadi

beberapa bit terkode berdasarkan bit-bit informasi masukan sebelumnya. Kode konvolusi dapat disebut kode dengan struktur kode (n, k, m) yaitu:

n = jumlah keluaran bit

k = jumlah masukan bit

m = jumlah memori (jumlah *shift register*)

Kode konvolusi juga dapat didefinisikan oleh beberapa generator polinomial, ini dikarenakan setiap elemen pada generator polinomial mendefinisikan tiga komponen utama dari konvolusi kode yaitu jumlah *shift register* atau panjang kode konvolusi (*constraint length*), jumlah keluaran bit (jumlah *modulo-2 adder*) dan hubungan koneksi antara *shift register* dan *modulo 2-adder.*

2.3. Parameter kode konvolusi

Parameter utama untuk membangun kode konvolusi yaitu :

1. *Rate*

Rate merupakan ratio antara masukan informasi bit dengan keluaran bit terkode dan mempunyai persamaan sebagai berikut:

$$R = k / n \tag{2.1}$$

Dimana:

R = laju kode konvolusi

k = jumlah bit masukan kode konvolusi

n = jumlah bit keluaran kode konvolusi

2. *Constraint Length Rate*

Constraint length adalah jumlah *delay* elemen dalam kode konvolusi yaitu memori dengan masukan bit sekarang pada kode konvolusi atau dapat disebut

juga panjang kode dari konvolusi kode. *Constraint length* dapat didefinisikan sebagai berikut:

$$K = m + 1 \tag{2.2}$$

Dimana:

K = *constraint length*

m = memori

3. Generator polinomial

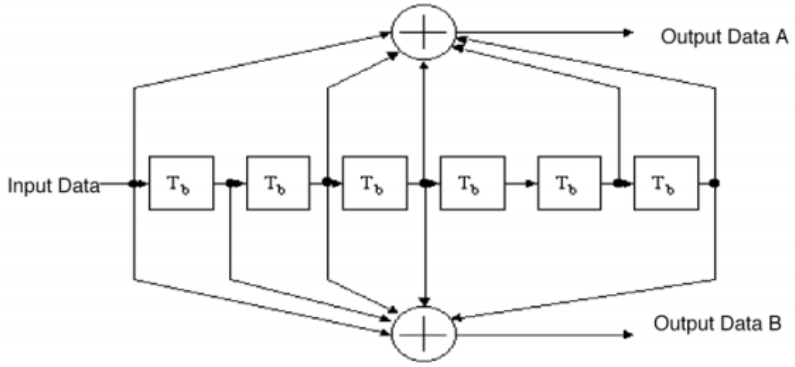
Generator polinomial sangat dibutuhkan untuk merangkai suatu kode konvolusi berdasarkan jumlah memori yang digunakan dalam suatu konvolusi kode selain itu setiap elemen pada generator polinomial serta jumlah dari fungsi generator polinomial mempengaruhi:

a. Jumlah output (jumlah *modulo-2 adder*)

b. Panjang konvolusi kode (jumlah *shift register* + input)

c. Hubungan koneksi antara *shift register* dan *modulo-2 adder*

Pada kode konvolusi banyak menggunakan kode *rate* [9] yaitu 1/2, 2/3, atau 3/4 sesuai data *rate* yang diinginkan. Kode konvolusi akan menggunakan generator polinomial dengan *standart* industri yang diterapkan. Pada *standart wireless IEEE 802.11a*, menggunaakan kode *rate* 1/2 dengan *standart* industri generator polynomial g0= 133_8, g1= 171_8, dilihat pada Gambar 2.1.



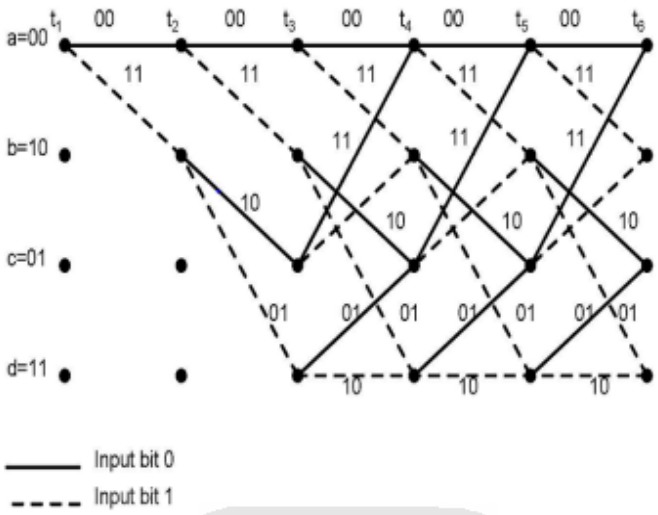
Gambar 2.1 *Convolusional encoder* (2,1,7) [9].

Pada Gambar 2.1, memperlihatkan sebuah konvolusi encoder (2,1,7) sederhana dengan n=2, k=1 dan m=7. Setiap kali sebuah bit data dimasukkan ke register pertama pada encoder, dua buah bit kode akan dihasilkan sebagai output secara berurutan, output data A akan menjadi output decoder sebelum output data B. jika masukan encoder lebih besar maka ada data bit yang dihilangkan sehingga mengurangi jumlah bit yang ditransmisikan dan meningkatkan tingkat *coding*. Untuk itu menambahkan 0 pada decoder pada sisi penerima pada bit yang dihilangkan tersebut. Salah satu algoritma yang dikenal secara luas untuk implementasi pada konvolusi kode adalah algoritma Viterbi.

2.4. Algoritma Viterbi

Algoritma Viterbi sangat cocok digunakan pada metode FEC pada kode konvolusi yang sudah dibuktikan pada penelitian sebelumnya dan mampu mengoreksi *error* pada sistem telekomunikasi digital.

Pada tahun 1967, Andrew J Viterbi memperkenalkan sebuah algoritma *decoding* untuk *convolutional encoder* yang saat ini dikenal dengan algoritma Viterbi[10]. Pada diagram *state* pada konvolusi encoder akan menentukan *state* pada algoritma Viterbi dan menentukan *state* selanjutnya. Algoritma Viterbi biasanya meggunakan *trellis* diagram untuk penggambarannya seperti pada Gambar 2.2.



Gambar 2.2 Decoder *trellis* diagram (rate 1/2 , K = 3) [5].

Untuk mendapatkan nilai akumulasi *error matrix* yang terkecil dari output encoder, kode konvolusi diperlukan data tentang state output dari encoder kode konvolusi tersebut. Bagaimana menggambarkan tentang hubungan *state* selanjutnya dan output pada enkoder kode konvolusi jika diberi input bernilai 1 atau jika input bernilai 0. Pada gambar tersebut jika input bernilai 1 maka akan digambarkan dengan garis putus-putus, jika input bernilai 0 maka digambarkan dengan garis lurus. Proses *trellis decoding* dimulai dari bentuk *state* awal 00 dimana ada 2 kemungkinan input yaitu 0 dan 1, keadaan *state* akan mengikuti masukan input selanjutnya yang membentuk dua buah *path* untuk tiap node. Dua *Path metric* pada *trellis* diagram mempunyai *hamming distance* yaitu jumlah perbedaan antara dua buah deretan bit yang mempunyai ukuran sama. *Hamming distance* antara 2 keadaan akan dibandingkan mana yang akumulari *error* paling sedikit terhadap keluaran kode konvolusi. *Path* yang dipilih akan membentuk

sebuah *trellis*, *trellis* akan di *traceback* atau proses penelusuran kembali jalur yang telah dilalui sehingga data informasi yang terkirim akurat.

2.5. *Hard Decision Decoding*

Setiap *path* yang tidak terputus pada *path matric* merupakan *survivor path* yang digunakan untuk menentukan *decoding path* dalam *trellis* diagram. Dalam metode Viterbi algoritma *hard decision decoding*, prinsip *maximum likehood* atau mencari kemungkinan bit yang mirip adalah mencari *path* dengan nilai *hamming distance* yang paling kecil yang akan menjadi kandidat penerus *survivor path* sebelumnya dalam proses *decoding* pada *trellis* diagram.

2.6. Model Kanal AWGN (*Additive White Gaussian Noise*)

AWGN[11], merupakan suatu proses stokastik yang terjadi pada kanal dengan karakteristik memiliki rapat daya *spectral noise* merata di sepanjang *range* frekuensi. AWGN mempunyai karakteristik respon frekuensi yang sama disepanjang frekuensi dan variannya sama dengan satu. Pada kanal *transmisi* selalu terdapat penambahan derau yang timbul karena akumulasi derau termal dari perangkat pemancar, kanal transmisi, dan perangkat penerima. Derau yang menyertai sinyal pada sisi penerima dapat didekati dengan model matematis statistik AWGN. Derau AWGN merupakan gangguan yang bersifat *Additive* atau ditambahkan terhadap sinyal transmisi, dimodelkan dalam pola distribusi acak Gaussian dengan mean (m) = 0, *standar deviasi* (σ) = 1, *power spectral density* (pdf) = No/2 (W/Hz), dan mempunyai rapat spektral daya yang tersebar merata pada lebar pita frekuensi tak berhingga. AWGN memiliki distribusi Gaussian, yang juga disebut distribusi Normal. Distribusi ini memiliki kepadatan

probabilitas yang simetris dan berbentuk seperti lonceng, dan fungsi kepadatan dinyatakan dengan:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2}\left[\frac{x-\mu}{\sigma} \right]^2 \right) \qquad (2.3)$$

Dimana: μ = rata-rata x x = nilai data

σ = standart deviasi π = 3,14

2.7. Model Flat Fading Rayleigh

Pada lingkungan NLOS (*non line of sight*), distribusi Rayleigh sering digunakan untuk menggambarkan statistik variasi sinyal pada kanal flat fading atau pada masing-masing komponen *path* pada lingkungan *multipath*. Kecepatan variasi sinyal tergantung pada *Doppler spread*. Pada fading Rayleigh sinyal melalui jalur yang berbeda-beda dan memberikan sejumlah energi yang sama terhadap sinyal gabungan yang ada pada penerima. Sinyal yang dipengaruhi fading Rayleigh yang sampai pada penerima dapat dipresentasikan dengan persamaan:

$$e(t) = r(t)\cos\left[2\pi ft + \theta(t)\right] \qquad (2.4)$$

Dimana: $r(t)$ =fluktuasi amplitudo sinyal e(t) sebagai fungsi waktu = $|e(t)|$

$\theta(t)$ =fluktuasi fasa sinyal $e(t)$ sebagai fungsi waktu = $< e(t)$

Fluktuasi amplitude gelombang pembawa pada sinyal yang dipengaruhi fading Rayleigh mengikuti distribusi Rayleigh, dengan persamaan:

$$p(r) = \frac{r}{\sigma^2} e^{-\left(\frac{r^2}{2\sigma^2}\right)} dengan(r \geq 0) \qquad (2.5)$$

Dimana: $p(r)$ =fungsi kepadatan probabilitas munculnya r

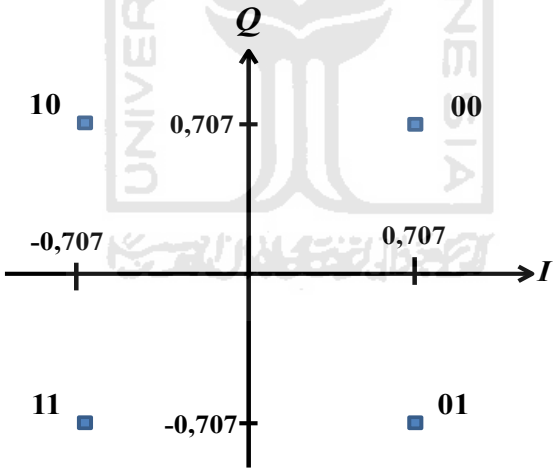
r =amplitude acak

σ^2 =varians pdf

2.8. Konsep Modulasi QAM (*Quadrature Amplitudo Modulation*)

Modulasi adalah suatu proses untuk merubah gelombang pembawa (*carrier)* sebagai fungsi dari sinyal informasi. *Quadrature Amplitude Modulation* (QAM) merupakan salah satu teknik modulasi digital. Pada QAM, informasi yang akan dikirimkan diubah menjadi simbol QAM yang dapat direpresentasikan sebagai sinyal analog pemodulasi. Orde QAM yang sering dinyatakan sebagai Q-ary QAM menunjukkan jumlah simbol QAM yang dapat dihasilkan ($Q = 2^n$), dengan n adalah jumlah bit penyusun satu simbol [12].

Pada Gambar 2.3 di bawah ini adalah salah satu diagram konstelasi dari QAM yaitu 4-QAM.



Gambar 2.3 Diagram konstelasi 4-QAM [12].

Dari diagram konstelasi di atas, maka data di jabarkan pada Tabel.2.1.

Tabel 2.1 Modulasi 4-QAM

Data Bolean	**Simbol Hasil Modulasi 4-QAM**

00	0,707+0,707i
01	0,707-0,707i
10	-0,707+0,707i
11	-0,707-0,707i

Sedangkan demodulasi 4-QAM, prosesnya berkebalikan dari proses modulasi 4-QAM.

2.9. *Software* LabVIEW

LabVIEW adalah sebuah bahasa pemrograman grafis yang menggunakan ikon sebagai pengganti barisan kalimat atau *teks* untuk menciptakan program. Apabila bahasa pemrograman berbasis *teks* seperti Visual Basic menggunakan urutan instruksi dan kode tertentu untuk menentukan jalannya program, maka LabVIEW menggunakan aliran data dalam bentuk gambar yang dapat disusun dari kiri ke kanan [13]. *Software* ini pertama kali dikembangkan oleh perusahaan National Instruments (NI) pada tahun 1986. LabVIEW merupakan kepanjangan dari *Laboratory Virtual Instrument Engineering Workbench* [14].

Pada *software* labVIEW memiliki kelebihan mudah dipahami karena berbentuk grafis dengan instruksi berbentuk ikon-ikon yang dihubungkan dengan garis yang menunjukkan *flowchart*. labVIEW juga didesain sebagai sebuah bahasa program parallel (*multicore*) yang mampu menangani beberapa intruksi sekaligus dalam waktu bersamaan, Hal ini sangat sulit dilakukan dalam bahasa program teks, karena biasanya bahasa program teks mengeksekusi intruksinya secara berurutan perbaris, satu demi satu. LabVIEW juga membuat aplikasi eksekusi menjadi sederhana dari program yang kompleks dan rumit [14]. Dari beberapa

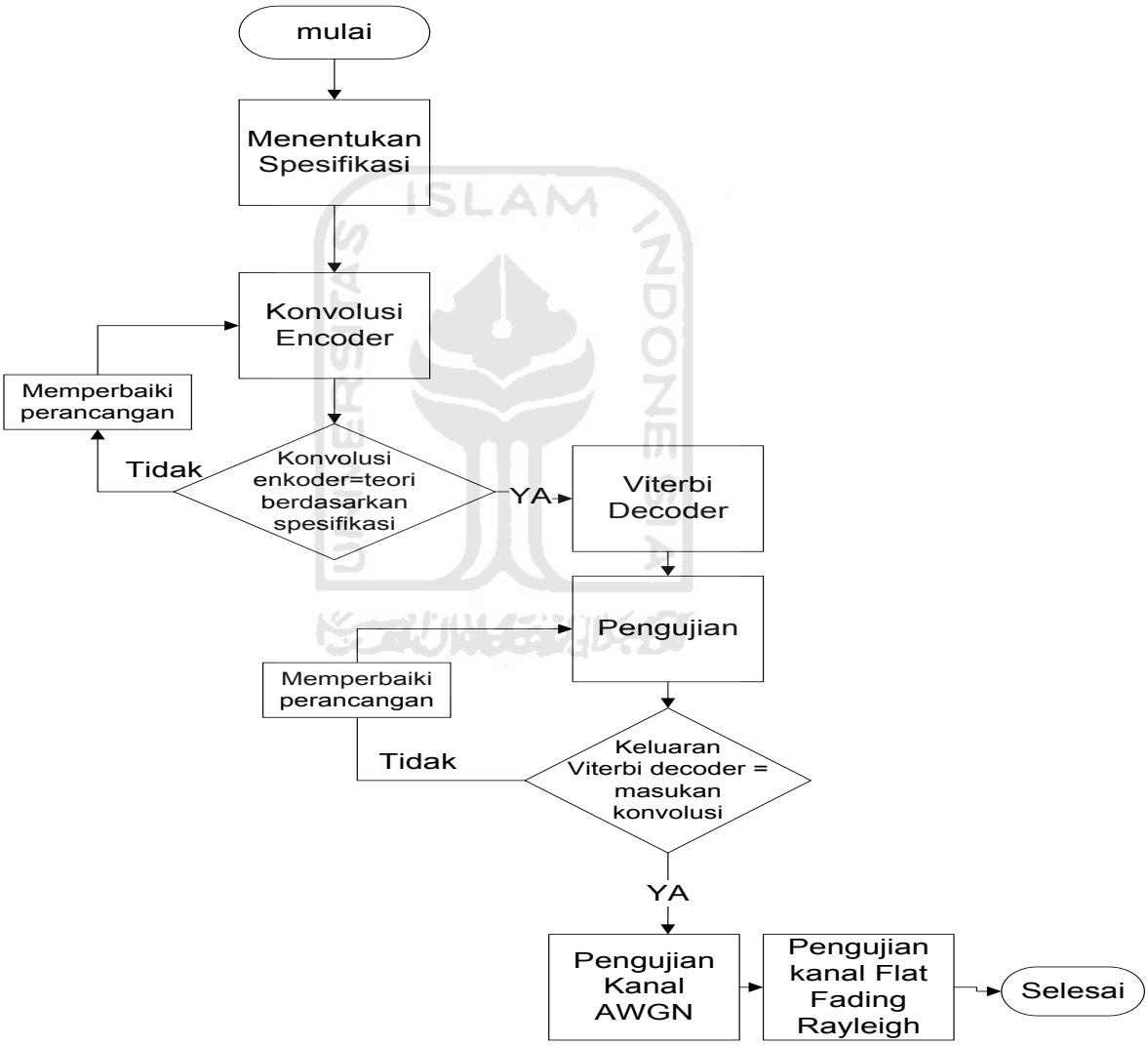
kelebihan *software* LabVIEW tersebut, maka dalam tugas akhir ini penulis menggunakan *software* LabVIEW untuk merancang kinerja FEC menggunakan algoritma Viterbi.

BAB III

PERANCANGAN SISTEM

3.1. Perancangan Sistem

Dalam melakukan perancangan sistem ditampilkan melalui diagram alir pada Gambar 3.1.



Gambar 3.1 Diagram alir penelitian

Pada Gambar 3.1 penelitian di mulai dengan menentukan spesifikasi yang digunakan dalam perancangan yaitu mengacu pada *wireless IEEE 801.11.a* dalam merancang kode konvolusi sebagai *transmitter*. Setelah itu, mengimplementasikan kode konvolusi dalam sebuah simulasi, hasilnya harus sesuai dengan teori, jika tidak sesuai maka memperbaiki rancangan dan jika sesuai maka akan masuk ke *receiver* yaitu implementasi algoritma Viterbi, hasil pada keluaran algoritma Viterbi akan diuji apakah sama seperti pada data yang masuk ke konvolusi kode, jika tidak maka akan diperbaiki lagi. Jika iya maka proses selanjutnya yaitu pengujian menggunakan model kanal AWGN, proses selanjutnya pengujian menggunakan kanal flat fading Raylaigh. Dari proses pengujian model kanal tersebut, maka akan diperoleh hasil performansi implementasi algoritma Viterbi sebagai FEC.

3.2 Spesifikasi yang digunakan dalam Simulasi

Sebelum melakukan simulasi, menentukan parameter dan spesifikasi adalah hal yang sangat penting. Berikut spesifikasi yang digunakan dalam penelitian tugas akhir berdasarkan *standart wireless 801.11a*. di lihat pada Tabel 3.1.

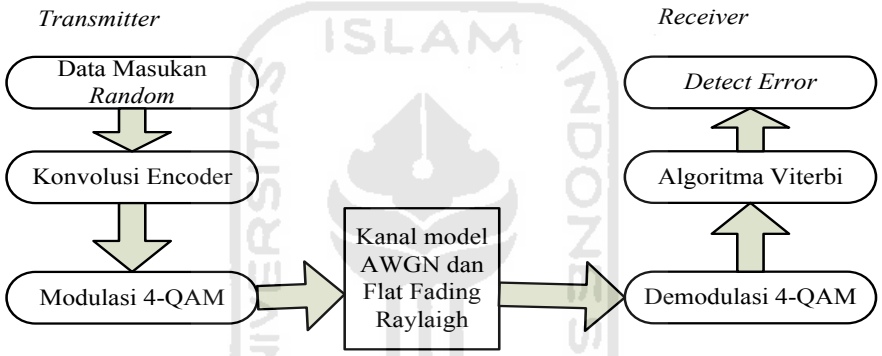
Tabel 3.1 Parameter yang digunakan dalam implementasi algoritma Viterbi

Parameter	**Nilai**
Frekuensi kerja (f*c*)	5 Ghz
Jumlah input	1
Jumlah output	2
Rate	½
Constraint length	7
Generator Polinomial	g0 = 133_8, g1 = 171_8

Parameter	Nilai
Skema Modulasi	4-QAM

3.3 Simulasi menggunakan *Software* LabVIEW

Pada simulasi implementasi algoritma Viterbi ini mengacu pada *standart* yang dikeluarkan dari *wireless IEEE 801.11a*. Berikut ini pada Gambar 3.2 adalah diagram blok perencanaan simulasi implementasi algoritma Viterbi menggunakan *software* LabVIEW.



Gambar 3.2 Perancangan implementasi algoritma Viterbi

Penjelasan dari beberapa blok implementasi algoritma Viterbi yang disimulasikan dengan menggunakan *software* LabVIEW yaitu sebagai berikut:

3.3.1 Blok-blok yang ada pada *Transmitter* (Pengirim):

3.3.1.1 Data masukan *random*

Data masukan berupa data *random.* Data yang dibangkitkan sebanyak 1.000.000 data. Dengan jumlah data yang dibangkitkan tersebut, maka nilai BER (*Bit Error Rate*) akan mencapai 10^{-6}.

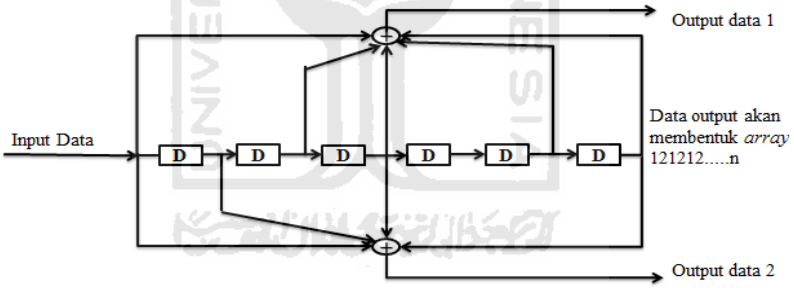
Selain itu, disebabkan data masukan berupa data *random* yang berupa data *numeric* bernilai antara 0 sampai 1. Maka data harus dirubah menjadi bentuk biner (0 atau 1). Caranya yaitu sebagai berikut:

a.) Apabila data *random* nilainya di atas 0,5 maka akan dikonversikan menjadi angka 1 dalam biner.

b.) Sedangkan apabila data *random* nilainya di bawah 0,5 atau sama dengan 0,5 maka akan dikonversikan menjadi angka 0 dalam biner.

3.3.1.2 Konvolusi Encoder

Data *random* dengan masukan 1.000.000 data akan di proses menggunakan kode konvolusi dengan *standart wireless IEEE 802.11a* yaitu dengan parameter industri generator polinomial g0= 133_8, g1= 171_8 , koefisien polinomialnya diwakili oleh nilai bit binary yaitu g0 =1011011, g1=1111001 dan diterapkan dalam blok konvolusi dengan kode *rate* ½ dilihat Gambar 3.3.



Gambar 3.3 *Convolusional encoder.*

Input data satu persatu dari data *random* akan bergeser berdasarkan jumlah *delay* elemen dan akan di XOR menurut generator polinomial seperti pada Gambar 3.3, keluaran data berupa 2 bit biner dengan urutan output bit 1 setelah itu bit 2 yang membentuk sebuah *array*, sehingga data output konvolusi akan berupa 2.000.000 bit biner.

3.3.1.3 Modulasi 4-QAM.

Data keluaran dari konvolusi kode akan masuk ke dalam blok modulasi 4-QAM. Sehingga dua data yang masuk akan dirubah menjadi satu simbol. Untuk lebih jelasnya, dapat dilihat diagram konstelasi serta penjelasan dari modulasi 4-QAM yang merujuk pada Gambar 2.3 dan Tabel 2.1.

3.3.2 *Channel*

3.3.2.1 *Channel Noise*

Dalam *channel noise* digunakan blok *channel* berikut:

3.3.2.1.1 Blok *noise* AWGN (*Additive White Gaussian Noise*).

Blok noise AWGN mengikuti pola distribusi Gaussian atau yang sering disebut dengan distribusi normal. Besarnya simpangan baku atau standar deviasi pada distribusi normal diatur oleh blok pengatur *noise*. Apabila pada pengatur *noise* tersebut dimasukkan bilangan yang kecil, maka standar deviasi dari *noise* AWGN yang dibangkitkan juga kecil, dan *range noise* AWGN yang dihasilkan semakin menyempit. *Range noise* AWGN yang menyempit tersebut, menyebabkan *noise* yang ditambahkan menjadi kecil. Berikut adalah algoritma untuk mendapatkan standar deviasi.

$$\sigma = \sqrt{\frac{\sum_{i=1}^{b} (A_{(i)} - \overline{A})^2}{b}} \qquad (3.1)$$

Dimana :

σ = standar deviasi

b = jumlah data yang dibangkitkan

$A_{(i)}$ = data *random* ke-i

$\overline{A}$ = rata-rata data *random*

3.3.2.1.2 Blok *noise* Flat Fading Rayleigh

Blok ini, sering digunakan untuk menggambarkan *statistic* variasi sinyal pada blok kanal flat fading. Kecepatan variasi sinyal tergantung pada *Doppler spread*, karena pergerakan yang acak komponen amplitude dari variabel komplek adalah Gaussian yang mana pengaturan *noise* berdasarkan blok AWGN. Parameter yang digunakan dalam blok *noise* Rayleigh Fading yaitu *Doppler spread* dengan perhitungan sebagai berikut:

$$f = V . f_c / c \qquad (3.2)$$

Dimana: f_c =Frekuensi *carrier*

V =Kecepatan sumber

c = Kecepatan cahaya (3 x10^8)

Pada penelitian ini menggunakan parameter, f_c = 5 Ghz, dan V=1000. Hasilnya *Doppler Spread* yang digunakan adalah 16.666 Hz.

3.3.2.2 .Penghitungan SNR (*Signal to Noise Ratio*)

Dalam blok penghitung SNR ini, didalamnya terdapat algoritma untuk perhitungan SNR. Algoritma perhitungan tersebut yaitu sebagai berikut:

$$\text{Total daya sinyal (data)} = \sum_{t=0}^{P} y_n^2 \qquad (3.3)$$

$$\text{Total daya sinyal (} noise\text{)} = \sum_{t=0}^{P} x_n^2 \qquad (3.4)$$

$$\text{SNR (dB)} = 10\log \frac{\textbf{Total daya sinyal (data)}}{\textbf{Total daya sinyal (}\boldsymbol{noise}\textbf{)}} \quad (3.5)$$

Dimana:

P = rentang pengamatan/pengukuran

y_n = sinyal data

x_n = sinyal *noise*

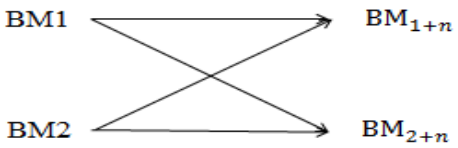
3.3.3 Blok-blok yang ada pada *Receive*r (Penerima):

3.3.3.1 Demodulasi 4-QAM dan Data Keluaran

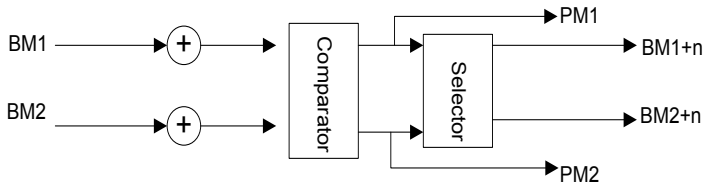
Data symbol mengalami proses demodulasi dengan menggunakan demodulasi 4-QAM. Sehingga data yang awalnya dari satu simbol akan di rubah menjadi dua data dalam bentuk biner. Untuk lebih jelasnya, dapat dilihat diagram konstelasi serta penjelasan dari demodulasi 4-QAM yang merujuk pada Gambar 2.3 dan Tabel 2.1. Data yang telah diproses demodulasi, data keluarannya diubah kembali seperti data masukan. Apabila data tidak sama, itu berarti terdapat data yang mengalami *error*.

3.3.3.2 Implementasi Algoritma Viterbi

Data masuk dibagi beberapa paket data sebagai perbandingan performansi *traceback,* data akan di cari *hamming distance* yang paling kecil dengan menggunakan 64 *output state* konvolusi encoder menggunakan metode *radix 2-Butterfly* (dilihat pada lampiran).



Gambar 3.4 *radix 2-Butterfly*

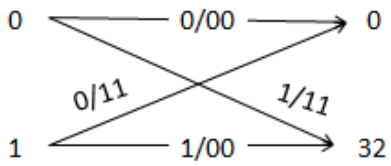


Gambar 3.5 ACS (*add-compare-selector*).

Pada proses algoritma Viterbi mengguakan *radix 2-Butterfly* dilihat pada Gambar 3.4 dapat dijabarkan pada Gambar 3.5 dengan proses ACS yaitu data masuk dalam perancangan akan ditambahkan dengan *branch matric* (BM) berupa *array size* 64x1 sesuai dengan *state* output konvolusi yaitu 64 *state*, dengan awal BM bernilai 0, data akan dibandingkan dan dipilih yang nilainya paling kecil, nilai data minimum akan masuk ke BM selanjutnya yang akan digunakan untuk data berikutnya sampai data selesai dan nilai paling kecil akan menjadi index awal *traceback*, jalur pengambilan nilai terkecil diinisialkan dengan 0 atau 1, jika data berasal dari jalur BM1 diinisialkan dengan 0 dan jalur BM2 diinisialkan dengan 1. Data jalur yang dipilih disimpan dalam sebuah *array size* 64 x n disebut *path metric* (PM). PM yang terakhir akan menjadi awal dari jalur *traceback* berdasarkan *index minimum* dari BM.

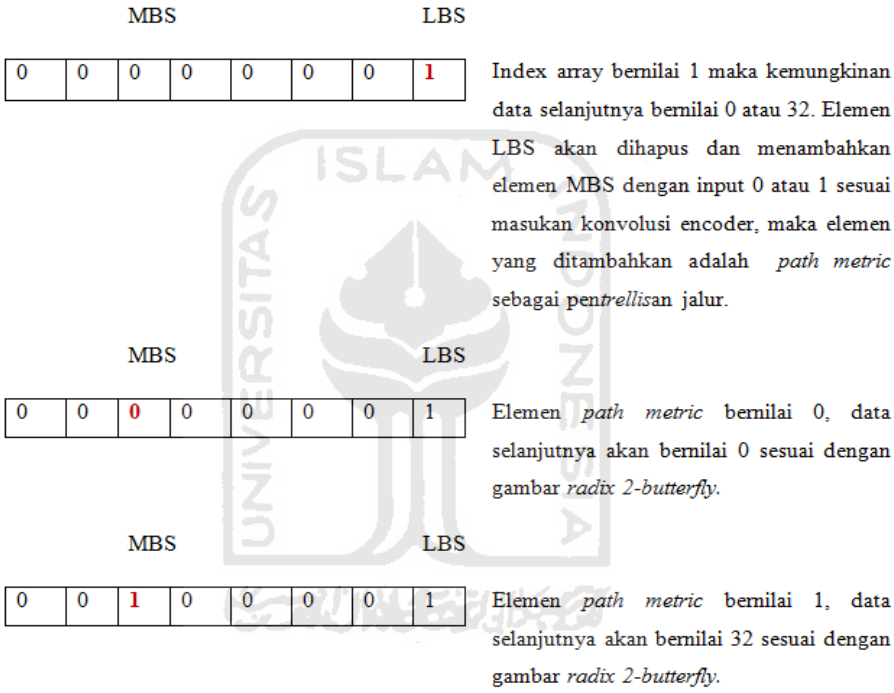
3.3.3.3 *Traceback*

Index minimum BM terakhir akan menjadi nilai awal dari *traceback*. Dimana akan menggunakan rangkaian digital yang sesuai dengan pola dalam *radix 2-butterfly* dengan menghilangkan LSB dan menambahkan MSB dengan elemen PM dalam *binary array*.



Gambar 3.6 Contoh *radix 2-Butterfly 64 state.*

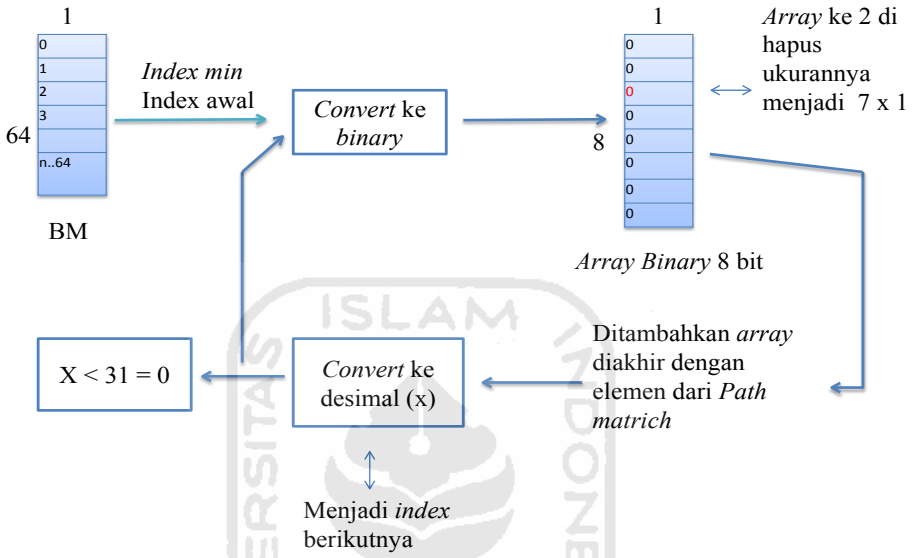
Dalam Gambar 3.6. dapat dipahami pada skema Gambar 3.7 sebagai berikut:



Gambar 3.7 Skema *radix 2-butterfly*

Pada perancangan, data BM berupa angka *numeric* di *convert* ke *binary array size* 8x1, LSB *binary array* akan dihapus pada elemen bit kedua karena *range* BM 0-63, sehingga berjumlah 6 digit dibaca dari elemen terakhir pada implementasi di labVIEW. Data bit akan ditambahkan dengan elemen berdasarkan index minimum BM pada PM dimulai dari *array* terakhir. Data akan di *convert* ke digital yang akan menjadi index berikutnya lihat pada skema *traceback* pada Gambar 3.8.

Data dari index menjadi data keluaran *receiver* dan dibandingkan sesuai dengan pola input konvolusi kode 64 *state* yaitu dalam *range* 0-31 maka data bernilai 0 dan *range* 32-63 data bernilai 1, Sehingga data akan kembali seperti pada data *transmitter*.



Gambar 3.8 Skema *Traceback*.

3.3.3.4 *Detect Error*

Cara untuk mengetahui data terdapat *error* dengan mencocokkan data *transmitter* dan *receiver* yaitu dengan cara *detect error*. Pada *detect error* ini, hasil data keluaran *receiver* di XOR dengan data masukan awal pada proses *transmitter*. Sehingga, apabila data tidak ada yang *error*, maka hasil detect berupa nol, dan apabila data terdapat *error*, maka berupa satu. Dari proses *detect error* akan diperoleh nilai BER (*Bit Error Rate*) dari simulasi tersebut. Algoritma untuk mendapatkan jumlah BER yaitu sebagai berikut:

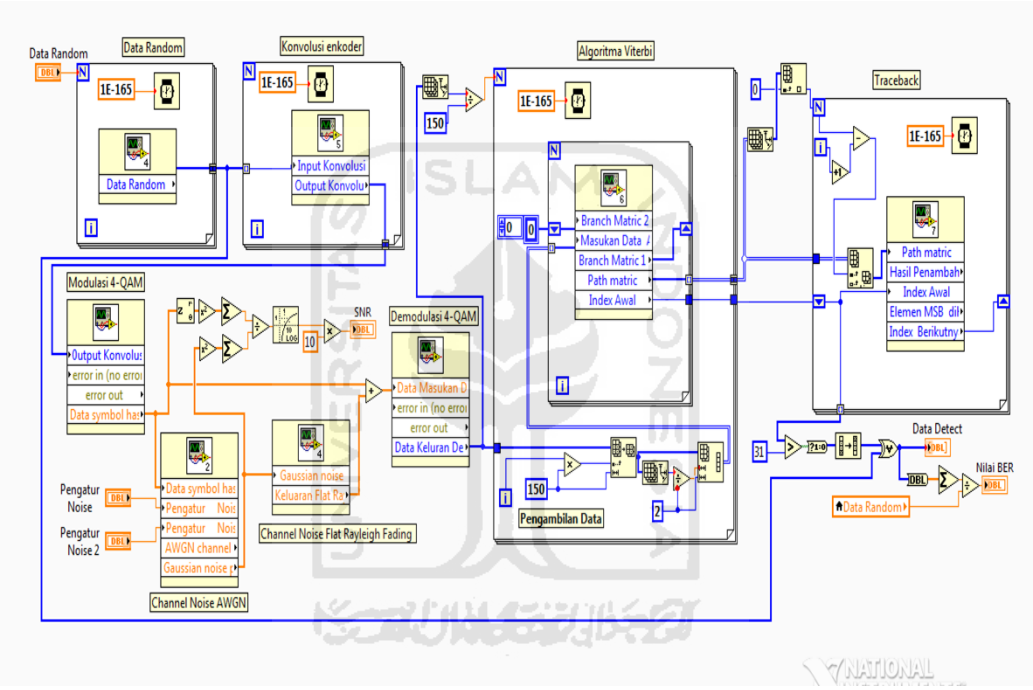
$$\text{BER (bit)} = \frac{\text{Jumlah data yang } error}{\text{Jumlah data keseluruhan}} \qquad (3.6)$$

BAB IV

HASIL PERANCANGAN dan SIMULASI

4.1 Hasil Perancangan

Hasil perancangan dan implementasi sistem yang telah dilakukan dengan menggunakan *software* LabVIEW, ditunjukkan pada Gambar 4.1.



Gambar 4.1 Hasil perancangan implementasi algoritma Viterbi dengan LabVIEW

4.2 Hasil Simulasi

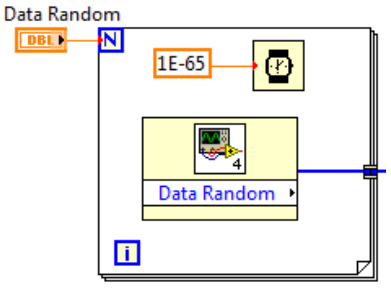
Pada simulasi implementasi algoritma Viterbi menggunakan *software* LabVIEW dengan masukan data random sebanyak 1.000.000 data. Pada hasil simulasi ini, menggunakan pengatur *noise* AWGN atau standar deviasi pada blok AWGN sebesar 0.37 yang besarnya SNR 8.6 dB dan dengan pengambilan data per 150 data dari masukan data pada *receiver*. Selain itu, grafik data yang diperoleh dari masing-masing proses dalam simulasi implementasi algoritma Viterbi diperoleh dengan cara menggunakan fitur *pause* memungkinkan pengguna untuk memahami setiap proses dan mengambil data pada setiap proses tersebut yang tersedia pada *software* LabVIEW.

4.2.1 *Transmitter*

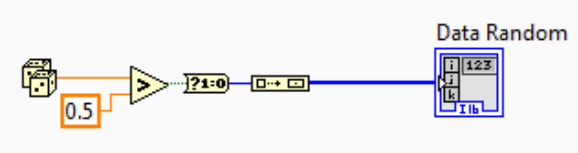
Hasil semua proses yang ada pada sisi pengirim dalam simulasi implementasi algoritma Viterbi dengan menggunakan *software* LabVIEW.

4.2.1.1 Data *Random*

Pada perancangan data *random* menggunakan subVI dengan tampilan pada Gambar 4.2, di dalam subVI perancangan data *random* seperti pada Gambar 4.3.

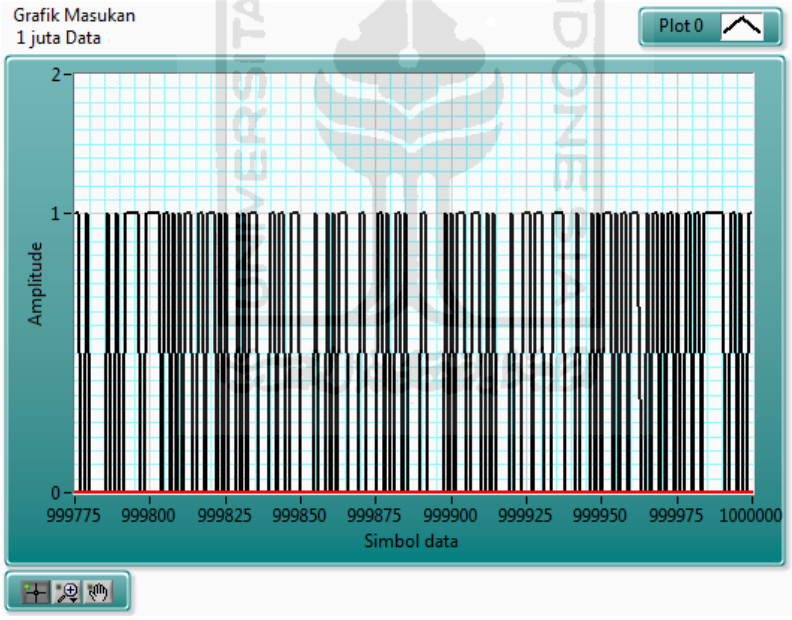


Gambar 4.2 Blok data *random* dalam subVI



Gambar 4.3 Perancangan data *random* di dalam subVI

Blok data *random* dalam *software* LabVIEW adalah blok yang digunakan untuk membangkitkan data *numeric* yang nilainya antara 0 sampai 1 secara acak. Jumlah data *random* yang digunakan dalam tugas akhir ini yaitu sebanyak 1.000.000 data. Pada Gambar 4.4 adalah grafik sinyal masukan 1.000.000 data random dari simulasi implementasi algoritma Viterbi dengan menggunakan *software* LabVIEW.

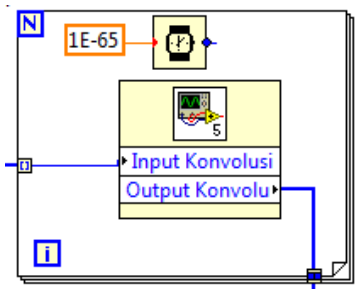


Gambar 4.4 Grafik sinyal masukan 1.000.000 data *random*

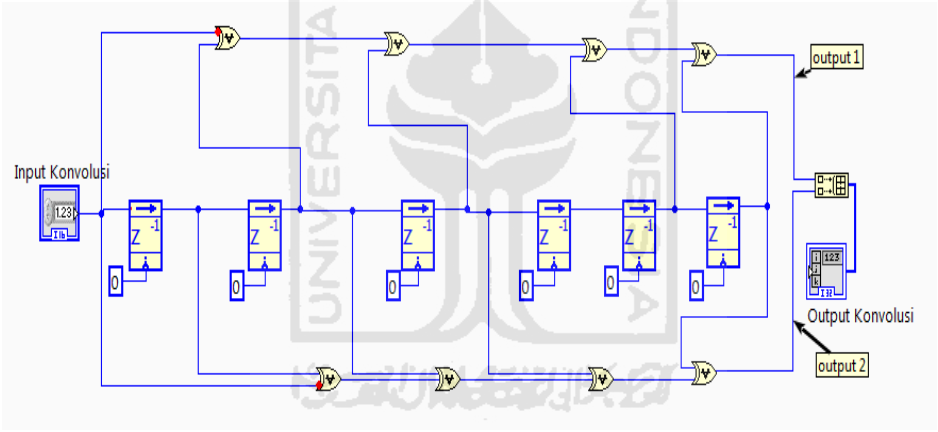
Pada grafik Gambar 4.4, data masukan *random* berjumlah 1.000.000 data. Selain itu, dapat dilihat amplitudo dari setiap data hanya 0 dan 1. Ini menunjukkan bahwa masukan data *random* berupa bilangan biner 0 ataupun 1.

4.2.1.2 Konvolusi Encoder

Pada perancangan konvolusi encoder menggunakan subVI dengan tampilan pada Gambar 4.5, di dalam subVI perancangan konvolusi encoder seperti pada Gambar 4.6.



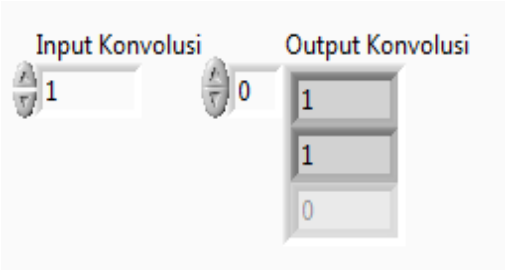
Gambar 4.5 Blok konvolusi encoder di dalam subVI



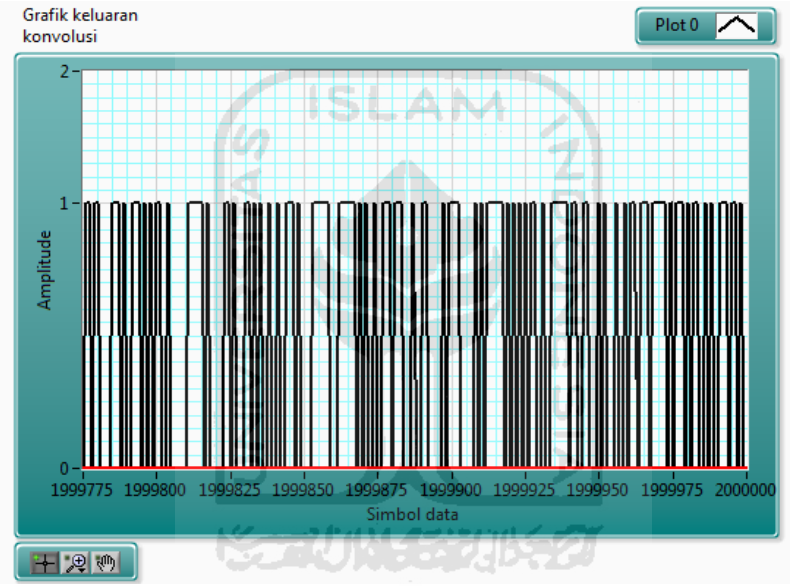
Gambar 4.6 Perancangan Konvolusi encoder di dalam subVI

Data *random* akan masuk satu persatu pada blok konvolusi pada *Feedback node* Transformasi Z sebagai *delay* memori sesuai dalam perancangan konvolusi encoder, data yang ada di *Feedback node* pada awalnya bernilai 0, setelah data masuk akan bergeser kedepan dengan 1 *delay* dan di XOR seperti pada Gambar 4.6. Pada blok konvolusi dengan kode *rate* 1/2 yaitu masukan 1 bit keluaran 2 bit lihat pada Gambar 4.7. Hasil konvolusi encoder berupa jumlah *binary* dilihat pada Gambar 4.8 yaitu amplitudo dari setiap data hanya 0 dan 1. Bit

masukan pada konvolusi encoder adalah 1.000.000 bit sehingga keluaran bit menjadi 2.000.000 bit.



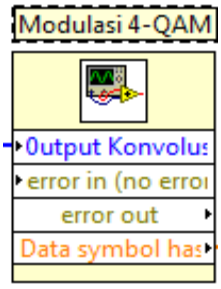
Gambar 4.7 Hasil data blok konvolusi encoder



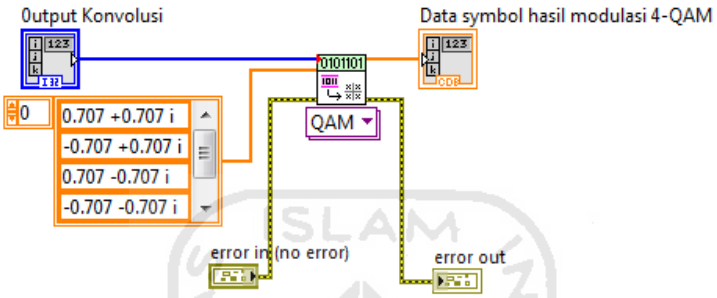
Gambar 4.8 Grafik sinyal keluaran Konvolusi encoder

4.2.1.3 Modulasi 4-QAM

Pada perancangan modulasi 4-QAM menggunakan subVI dengan tampilan pada Gambar 4.9, di dalam subVI perancangan modulasi 4-QAM seperti pada Gambar 4.10.

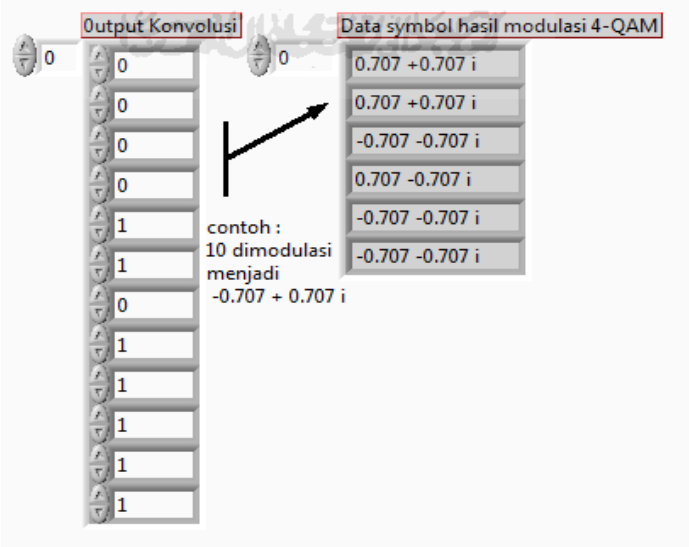


Gambar 4.9 Blok Modulasi 4-QAM



Gambar 4.10 Modulasi 4-QAM di dalam subVI

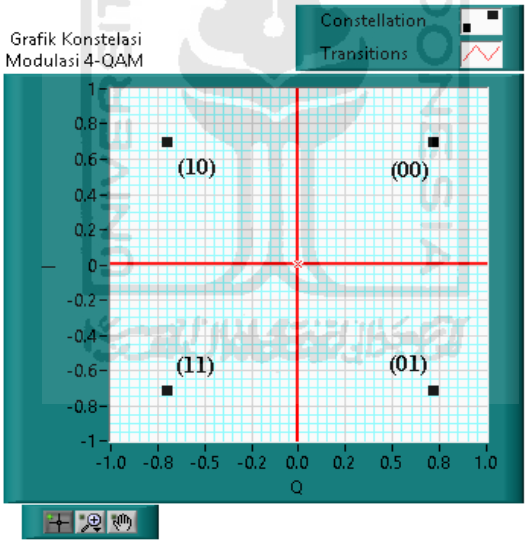
Data masukan pada modulasi 4-QAM adalah bit keluaran konvolusi encoder. Hasil data yang diperoleh dari modulasi 4-QAM bisa dilihat pada Gambar 4.11.



Gambar 4.11 Blok hasil data simbol modulasi 4-QAM

Modulasi 4-QAM adalah modulasi digital yang memiliki empat *fase* dalam konstelasinya. merubah dua data biner menjadi satu simbol. Dari hasil yang terdapat pada Gambar 4.11 di atas, sudah sesuai dengan diagram konstelasi modulasi 4-QAM yaitu Dua data *random* pertama yaitu 00 dirubah menjadi simbol 0,707+0,707i. Pada diagram konstelasi, data 00 terdapat di kuadran satu, dimana simbol yang dihasilkan pada sumbu I (*real*) dan Q (imajiner) berada pada sumbu positif.

Untuk lebih jelasnya, hasil dapat dipastikan melalui diagram kontelasi modulasi 4-QAM dari hasil simulasi implementasi algoritma Viterbi menggunakan *software* LabVIEW pada Gambar 4.12.



Gambar 4. 12 Grafik konstelasi modulasi 4-QAM

Gambar 4.13 Grafik sinyal setelah di modulasi

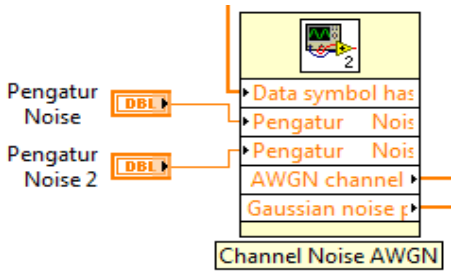
Pada Gambar 4.13 adalah grafik data random setelah dimodulasi, dapat dilihat pada data disumbu X (Simbol Data). Data hasil modulasi sebanyak 1.000.000 data simbol, dengan amplitudo yang terdapat pada sumbu Y menunjukkan amplitudo pada ketinggian 0,707 atau -0,707.

4.2.2 *Channel*

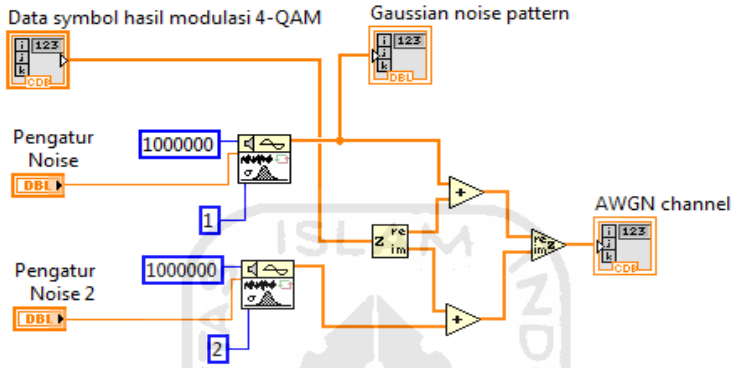
Pada proses selanjutnya, data masuk pada blok *channel*. Pada pengujian dilakukan pada 2 *channel* yaitu *channel noise* AWGN dan *channel noise* flat fading Rayleigh dengan pengaturan *noise* pada tiap *channel* 0.37 atau SNR 8,6 dB.

4.2.2.1 *Channel Noise* AWGN

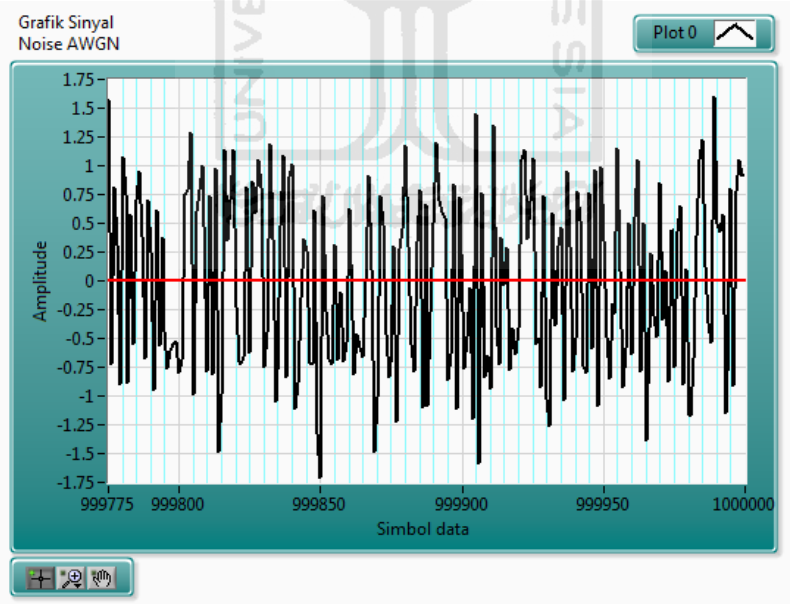
Pada perancangan *channel noise* AWGN menggunakan subVI dengan tampilan pada Gambar 4.14, di dalam subVI perancangan *channel noise* AWGN seperti pada Gambar 4.15.



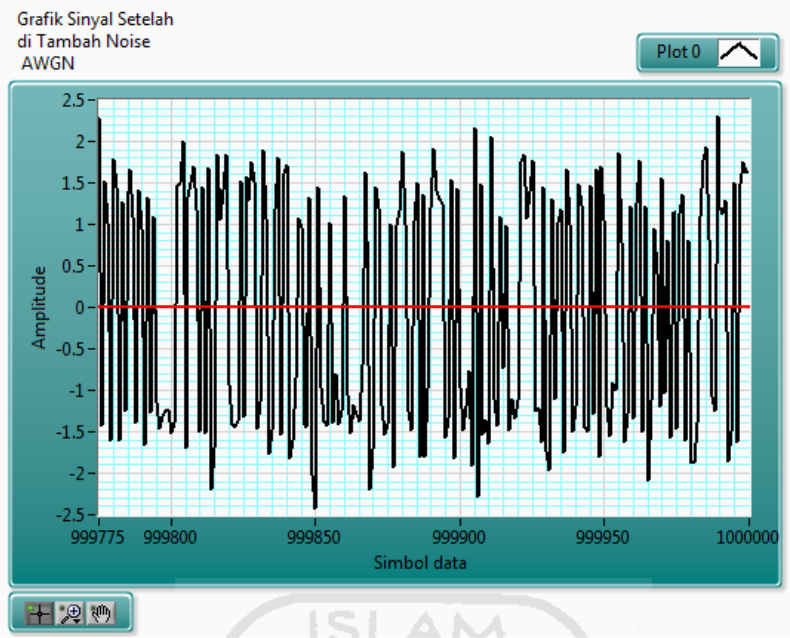
Gambar 4.14 *Channel noise* AWGN



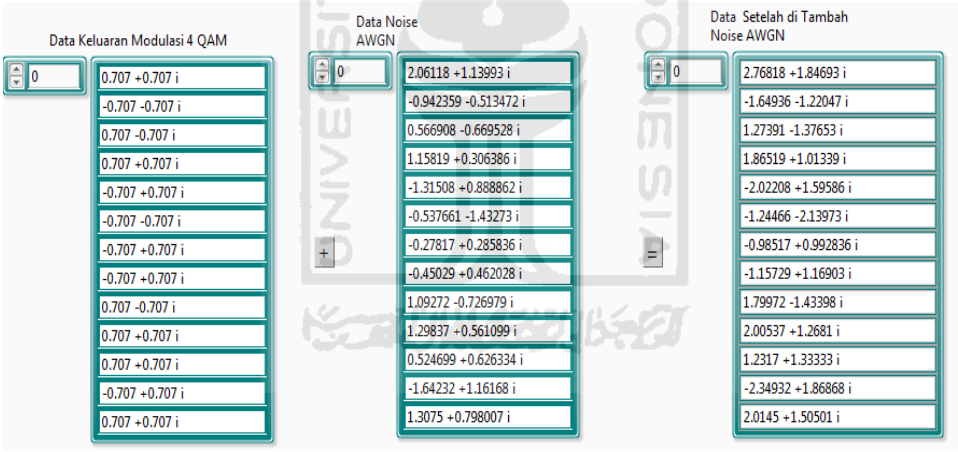
Gambar 4.15 *Channel noise* AWGN di dalam subVI



Gambar 4.16 Grafik sinyal *noise* AWGN



Gambar 4.17 Hasil grafik sinyal setelah di tambah *noise* AWGN

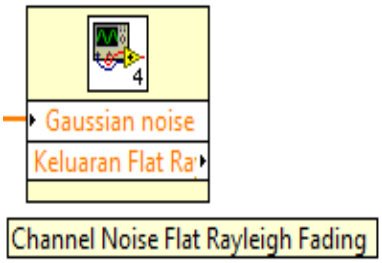


Gambar 4.18 Data pada *channel noise* AWGN

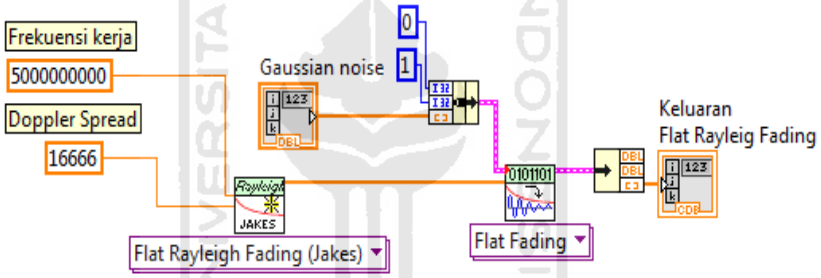
Hasil modulasi 4-QAM setelah ditambahkan *channel noise* AWGN hasil datanya mengalami perubahan atau *error* dilihat pada perbandingan grafik keluaran modulasi 4-QAM pada Gambar 4.13 dengan hasil penambahan *noise* pada Gambar 4.17. Ini sebabkan karena adanya pengatur *noise* sebesar 0,37 atau SNR hanya sebesar 8,6 dB menghasilkan *noise* pada Gambar 4.16. Data dalam penambahan *noise* bisa dilihat pada Gambar 4.18.

4.2.2.2 *Channel Noise* Flat Fading Rayleigh

Pada perancangan flat fading Rayleigh menggunakan subVI dengan tampilan pada Gambar 4.19, di dalam subVI perancangan flat fading Rayleigh seperti pada Gambar 4.20.

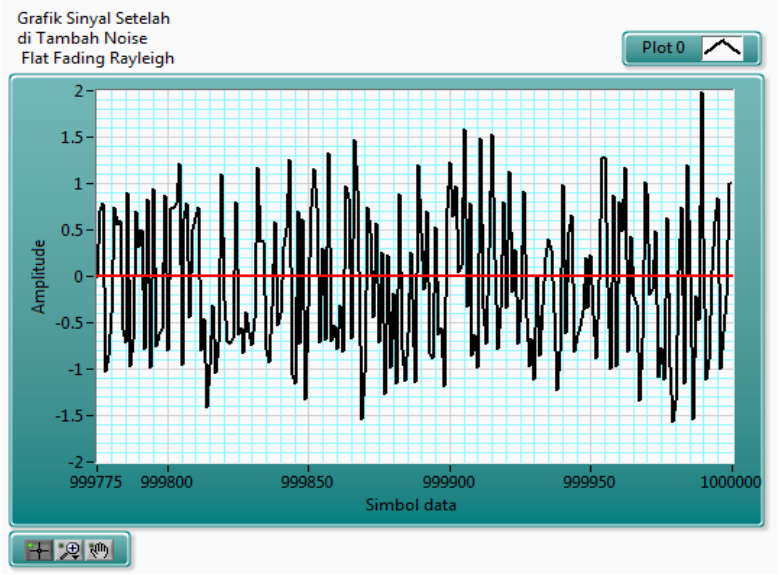


Gambar 4.19 Blok *channel noise* Flat Fading Rayleigh

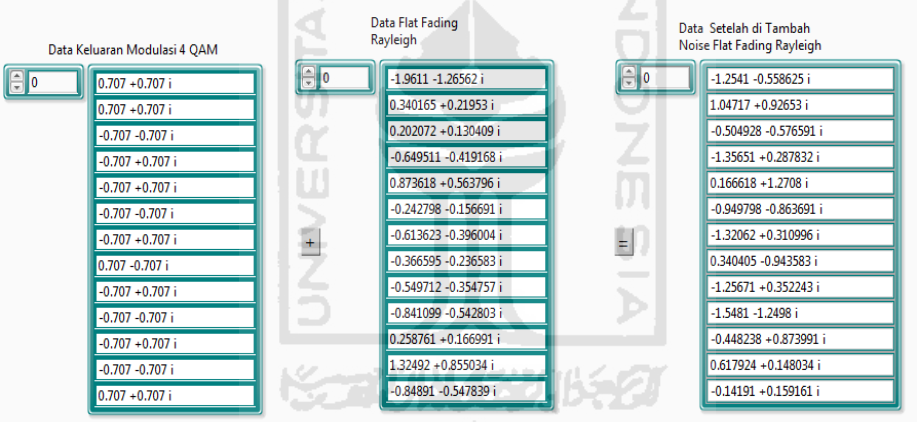


Gambar 4.20 Perancangan data *random* di dalam subVI

Gambar 4.21 Grafik sinyal *noise* Flat Fading Reylaigh



Gambar 4.22 Hasil Grafik Sinyal Setelah di Tambah *Noise* Flat Fading Rayleigh



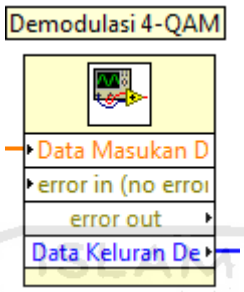
Gambar 4.23 Data pada *channel noise* flat fading Rayleigh

Hasil modulasi 4-QAM setelah ditambahkan *noise channel* Flat Fading Reylaigh hasil datanya lebih banyak mengalami *error* di banding dengan *channel* AWGN ini karena pada *channel* Flat Fading lebih komplek penyebaran *noise* nya. Flat Fading sendiri mengakibatkan berkurangnya daya sinyal dan model Rayleigh dengan datanya yang sangat acak sehingga komponen terdistribusi kompleks Gaussian bisa dilihat pada perbandingan Gambar 4.17 dengan Gambar 4.22. Hasil *noise channel* Flat Fading Rayleigh bisa dilihat pada Gambar 4.21. Data dalam penambahan *noise* bisa dilihat pada Gambar 4.23.

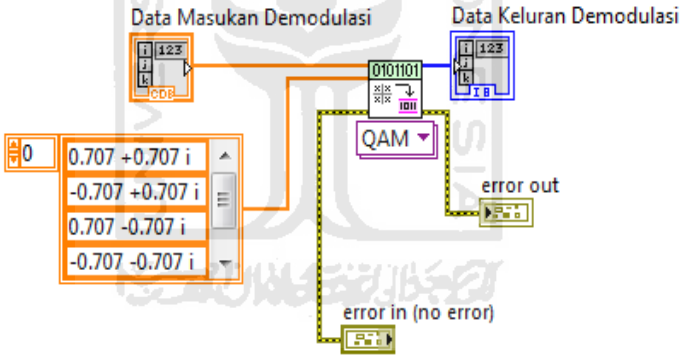
4.2.3 *Receiver*

4.2.3.1 Demodulasi dan Data Keluaran

Pada perancangan demodulasi 4-QAM menggunakan subVI dengan tampilan pada Gambar 4.24, di dalam subVI perancangan demodulasi 4-QAM seperti pada Gambar 4.25.

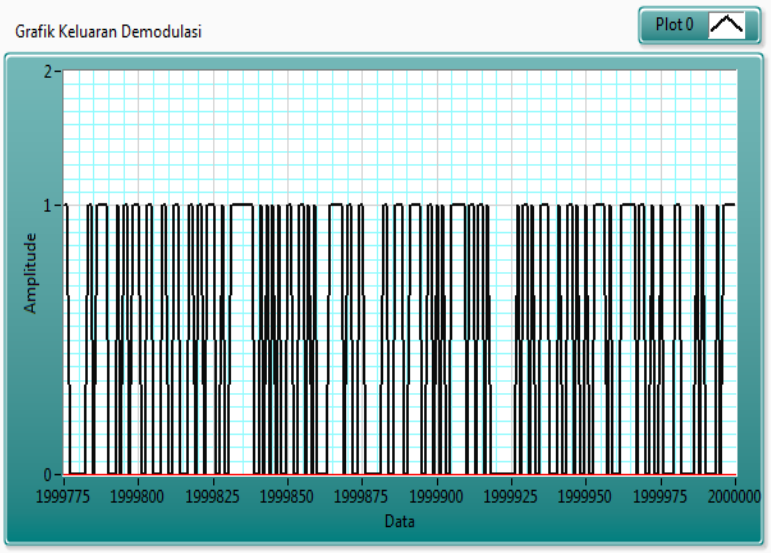


Gambar 4.24 Blok Demodulasi 4-QAM

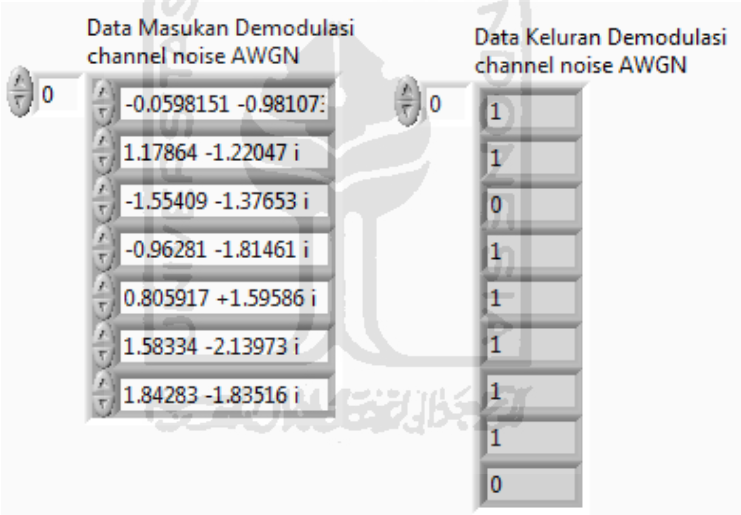


Gambar 4.25 Blok Demodulasi 4-QAM di dalam subVI

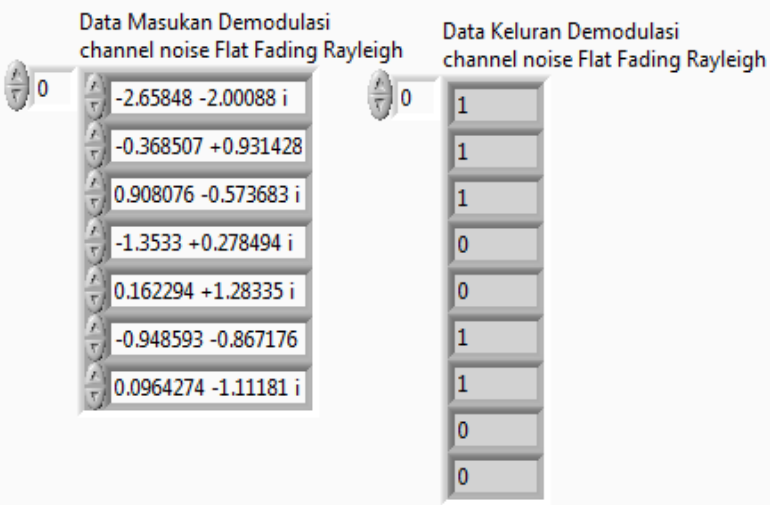
Demodulasi 4-QAM adalah proses mengubah data simbol pada *receiver* menjadi data biner. Sehingga, satu data simbol akan diurai kembali ke dalam dua bit biner, sehingga hasil demodulasi berjumlah 2.000.000 data biner lihat pada Gambar 4.26 Pada pengujian ini dilakukan dengan 2 keadaan yaitu masukan diberi *noise* AWGN dan *noise* flat fading Rayleigh.



Gambar 4.26 Grafik Blok Demodulasi 4-QAM

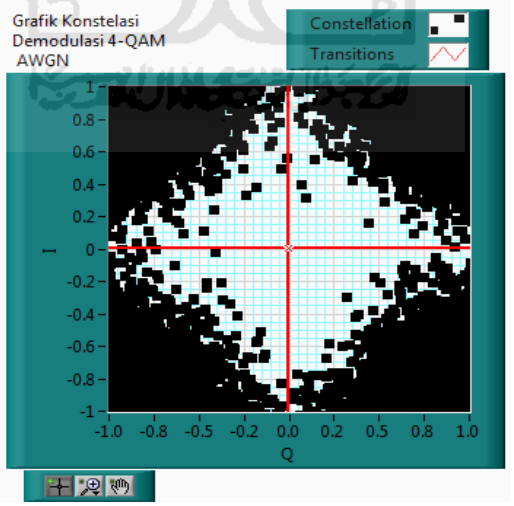


Gambar 4.27 Hasil Demodulasi 4-QAM *channel noise* AWGN

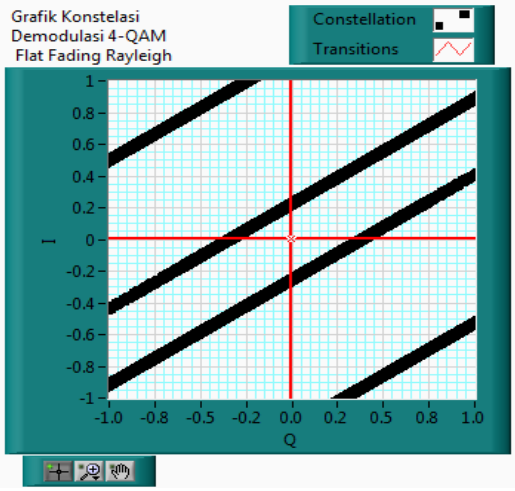


Gambar 4.28 Hasil Demodulasi 4-QAM *channel noise* Flat Fading Rayleigh

Hasil pada demodulasi 4-QAM yaitu penguraian data ke *binary* dengan letak konstelasi data pada modulasi 4-QAM. Dimana sampel pertama -0.0598151-0.98107 berdasarkan letak konstelasi modulasi 4-QAM dilihat pada Gambar 4.12 akan menjadi 11.



Gambar 4.29 Grafik konstelasi Demodulasi 4-QAM dengan penambahan *noise* AWGN.

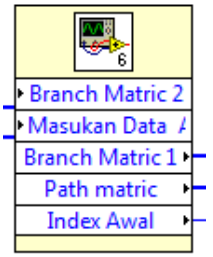


Gambar 4.30 Grafik konstelasi demodulasi 4-QAM dengan penambahan *noise* flat fading Rayleigh

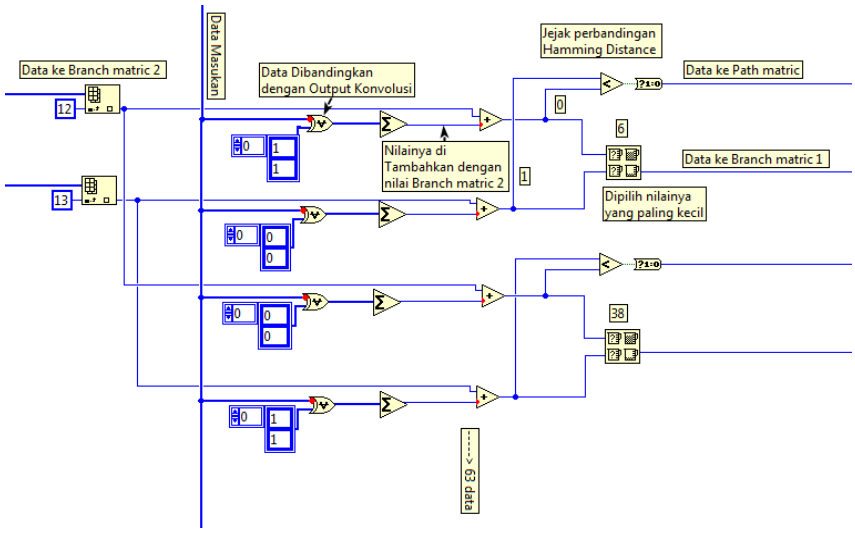
Apabila dilihat dari diagram konstelasi hasil demodulasi 4-QAM pada Gambar 4.29 dan Gambar 4.30, data yang diterima pada *receiver* mengalami *error* yang cukup banyak karena menggunaka SNR sebesar 8,6 dB.

4.2.3.2 Implementasi Algoritma Viterbi

Pada perancangan algoritma Viterbi menggunakan subVI dengan tampilan pada Gambar 4.31, di dalam subVI perancangan algoritma Viterbi seperti pada Gambar 4.32

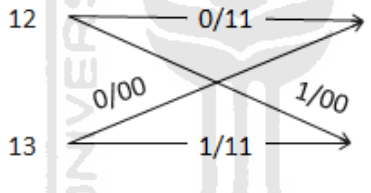


Gambar 4.31 Implementasi Algoritma Viterbi (ACS, *Branc Matric, Path Matric*)

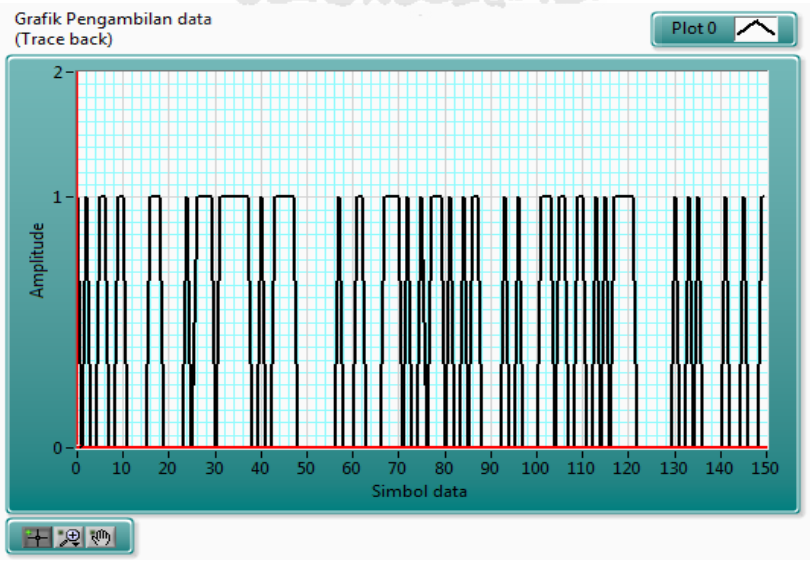


Gambar 4.32 Implementasi Algoritma Viterbi di dalam subVI

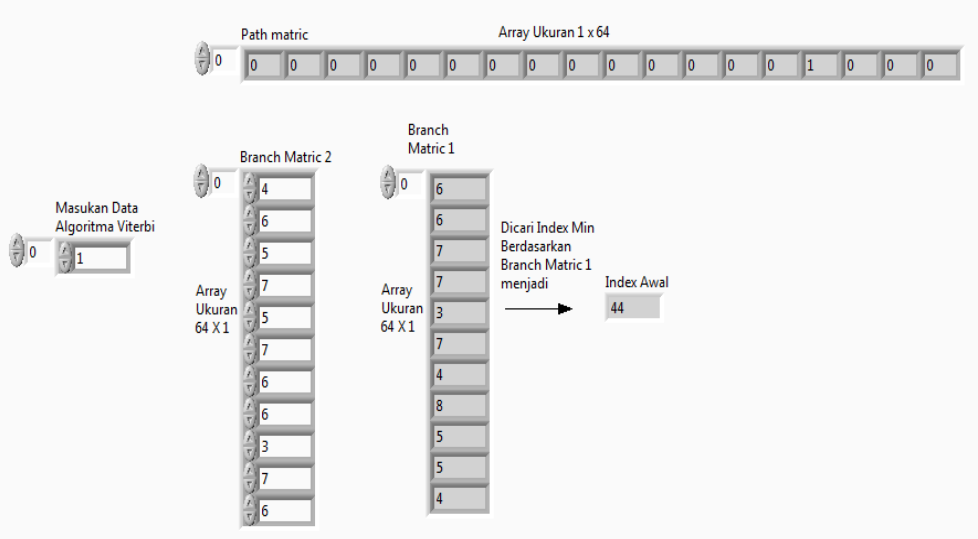
Dalam Gambar 4.32 adalah sebagian dari Implementasi Algoritma Viterbi (ACS, *Branc Matric, Path Matric*) pada *radix 2-butterfly* pada Gambar 4.33.



Gambar 4.33 *Radix 2-Butterfly*



Gambar 4.34 Grafik pengambilan 150 data

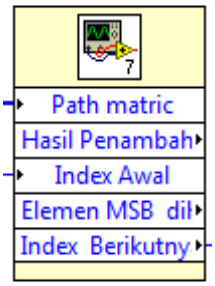


Gambar 4.35 Hasil *Branch matric* dan *Path matric*

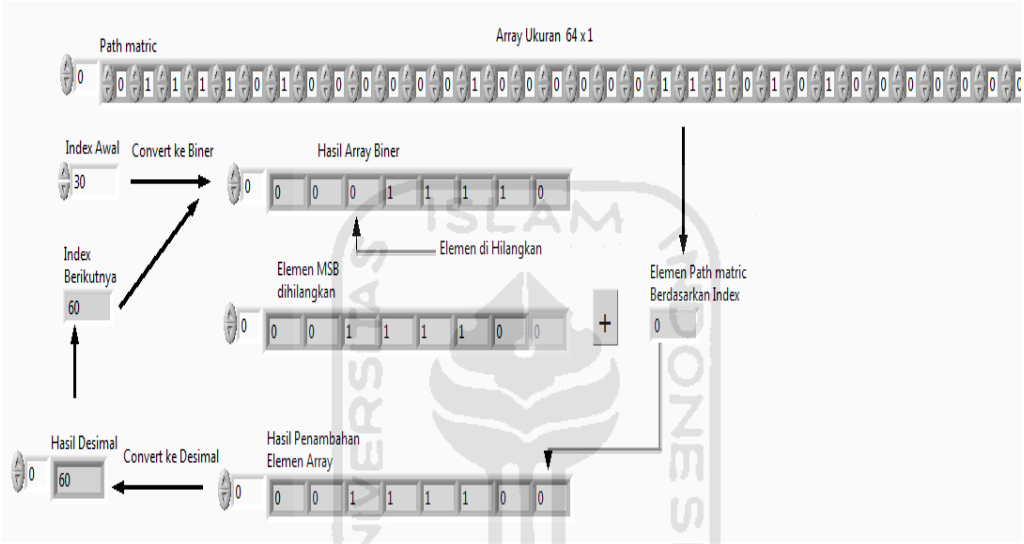
Pada pengujian ini data masuk ke implementasi algoritma Viterbi dengan pengambilan per 150 bit dengan BM meriset keadaan dari awal lagi yaitu 0 pada BM2. Data masuk per 2 bit dari pengambilan data sesuai dengan keluaran konvolusi yaitu 2 bit, data akan di bandingkan atau dicari *hamming distance* terkecil yang akan masuk ke BM1, data akan disalin ke BM2 sebagai nilai BM selanjutnya. Jejak jalur pengambilan data akan disimpan di PM dengan nilai keadaan input masukan konvolusi yaitu 0 dan 1. Perancangan ini sesuai dengan metode *radix-2 butterfly*. Nilai BM terakhir akan menjadi nilai awal dalam *traceback* lihat pada Gambar 4.35.

4.2.3.3 *Traceback*

Pada perancangan *traceback* menggunakan subVI dengan tampilan pada Gambar 4.36, di dalam subVI perancangan *traceback* seperti pada Gambar 4.37.

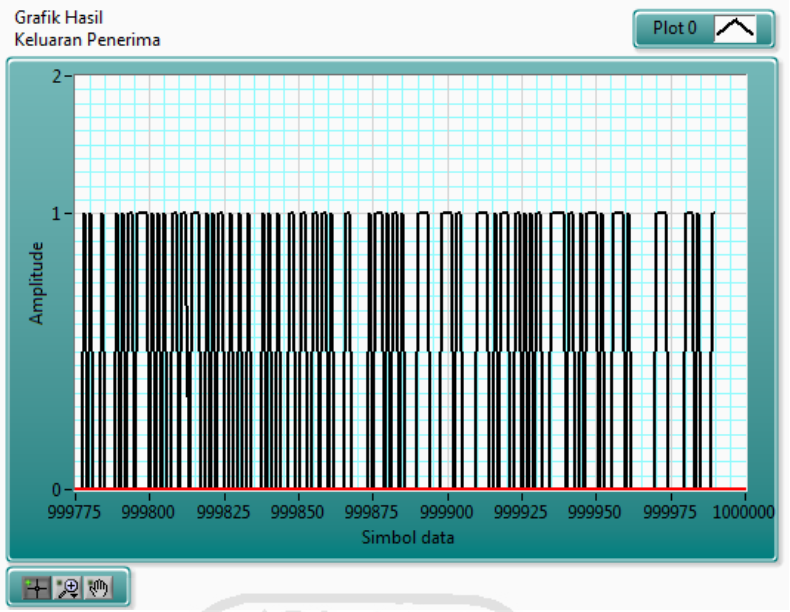


Gambar 4.36 Blok *Traceback*



Gambar 4.37 Blok *Traceback* di dalam subVI

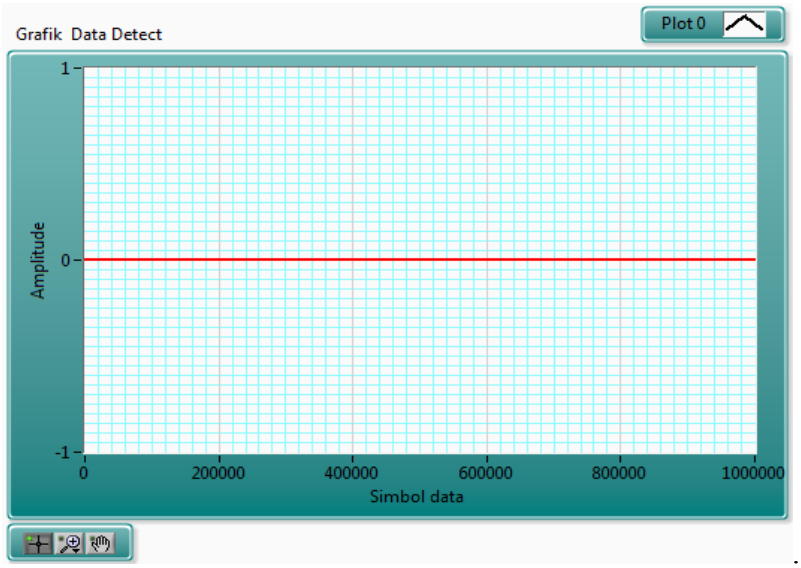
Pada perancangan *tracebak* index BM1 sebagai nilai awal dari pentrellisan data sesuai dengan perancangan, data akan di *convert* ke data biner, dihapus data ke dua atau LSB binary array setelah itu data akan ditambahkan MSB dari elemen PM lihat pada Gambar 4.37. Data dari index awal dan index selanjutnya akan dibandingkan sesuai dari *state* input konvolusi yaitu *range* data 0-31 maka data akan bernilai 0 dan *range* 32-63 data bernilai 1, data akan menjadi hasil data *receiver* atau penerima. Hasil keluaran *traceback* dilihat pada Gambar 4.38.



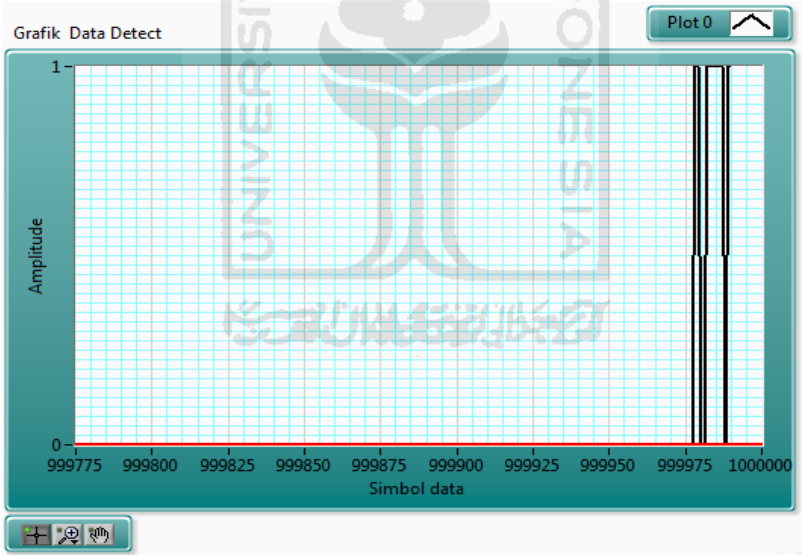
Gambar 4.38 Grafik keluaran *receiver*

4.2.3.4 *Detect error*

Dalam data *detect,* digunakan untuk mendeteksi data apabila terjadi *error.* Apabila terdapat data yang *error*, maka dalam *detect* akan ada data yang bernilai 1 atau yang memiliki amplitudo 1 pada gambar grafik *detect*. Namun apabila pada grafik *detect* bernilai 0, maka data tidak mengalami *error*. Hasil detect pada pengujian dengan penambahan *noise* AWGN dengan SNR sebesar 8.6 dB hasilnya tidak ada data yang *error,* sedangkan pada penambahan *noise* flat fading Rayleigh terdapat 2323 data bit yang *error.*



Gambar 4.39 Grafik *detect error* implementasi algoritma Viterbi pada *noise* AWGN



Gambar 4.40 Grafik *detect error* implementasi algoritma Viterbi pada *noise* flat fading Rayleigh.

4.2.4.Hasil Pengujian dan Performansi Implementasi Algoritma Viterbi

Pada pengujian Implementasi algoritma Viterbi ini, dilakukan dua pengujian yaitu dengan beberapa masukan pengatur *noise* yang berbeda-beda dengan *traceback* 150 dan pada penelitian *traceback* digunakan panjang data yang

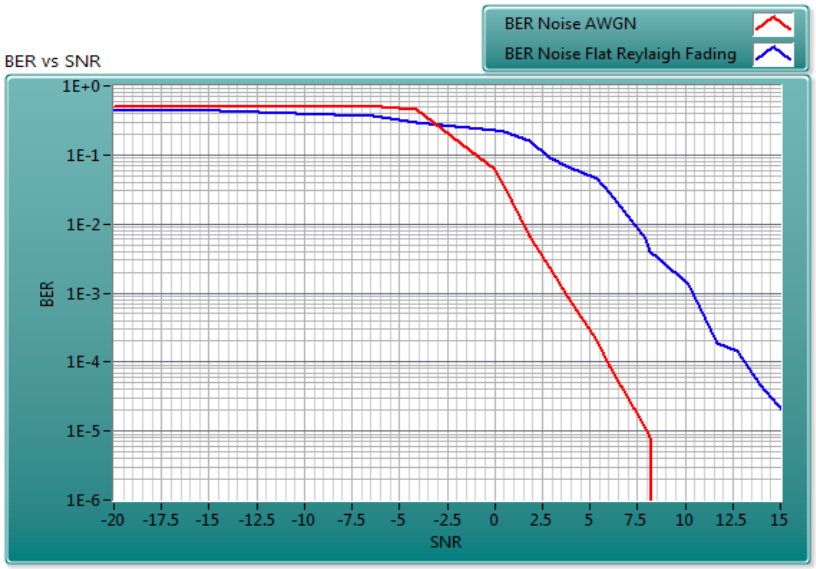
berbeda-beda. Pada pengujian ini, menggunakan SNR sebesar 8,6 dB. Dengan pengujian dilakukan seperti itu, BER yang dihasilkan akan bervariasi dari setiap pengujian. Dari data yang diambil dari pengujian implementasi algoritma Viterbi maka akan diketahui performansi implementasi algoritma Viterbi terhadap *noise* dan panjang data atau *traceback*. Hasil data yang diperoleh dari pengujian implementasi algoritma Viterbi dengan pengaturan *noise* yang berbeda dilihat pada Tabel 4.1.

Tabel 4.1 Hasil data pengujian SNR dan BER dari Implementasi algoritma Viterbi

Pengujian ke-	Noise	SNR (dB)	BER Channel AWGN	BER Channel Flat Fading Rayleigh
1	10	-20.0017	0,500532	0,463712
2	5,25	-14.4049	0,499886	0,442378
3	2,08	-6,363	0,492703	0,359443
4	1,98	-5,93504	0,4896	0,356529
5	1,6	-4,08413	0,451591	0,295586
6	1	-0,0017323	0,060922	0,227668
7	0,95	0,443796	0,036664	0,21313
8	0,81	1,82857	0,006527	0,16337
9	0,72	2,85162	0,002339	0,092856
10	0,64	3.87467	0,000843	0,067391
11	0,54	5,35039	0,0002	0,044689
12	0,5	6,01887	0,000087	0,028071
13	0,4	7,95707	0,000011	0,006068

Pengujian ke-	Noise	SNR (dB)	BER Channel AWGN	BER Channel Flat Fading Rayleigh
14	0,39	8,17698	0,000008	0,003961
15	0,37	8,63423	0	0,003155
16	0,31	10,1764	0	0,001315
17	0,26	11,6991	0	0,000187
18	0,23	12,765	0	0,000143
19	0,2	13,9723	0	0,000047
20	0,175	15,1405	0	0,000019

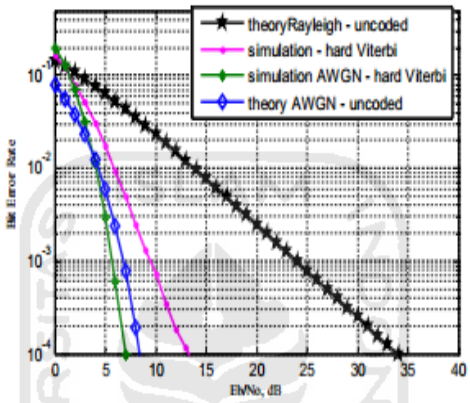
Hasil dari 20 pengujian tersebut dapat dilihat bahwa semakin kecil nilai SNR pada implementasi algoritma Viterbi, maka akan semakin besar BER yang diperoleh. Selain itu, terlihat jelas bahwa dengan pengaturan *noise* yang besar akan mempengaruhi jumlah data yang *error* pada penerima karena nilai SNR semakin mengecil. Penggunaan *noise channel* AWGN dan *channel* flat fading Rayleigh hasilnya juga sangat berbeda dilihat pada Gambar 4.41 dimana menggunakan *channel* flat fading Rayleigh lebih rentan terhadap *noise*, ini dikarenakan *noise* pada *channel* flat fading menggunakan jalur yang berbeda-beda disebabkan karena pemantulan sinyal pada saat pengiriman ke *receiver* sedangkan *noise* AWGN hanya melalui satu jalur yang sama dan tidak mengalami pemantulan sinyal pada saat pengiriman ke *receiver*. Penggunakan *channe*l AWGN mencapai BER $\mathbf{10^{-6}}$ pada SNR 8,6 dB sedangkan pada *channel* flat fading di SNR sebesar 8,6 dB masih mengalami *error* sebesar 3155 bit *error.*



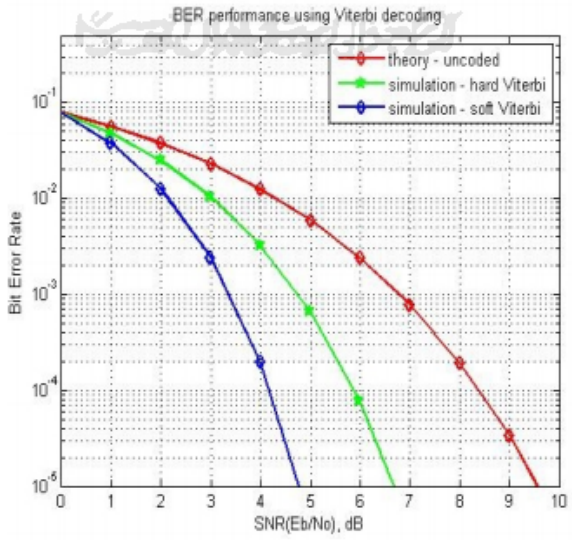
Gambar 4.41 Grafik perbandingan BER dengan SNR pada implementasi algoritma Viterbi

Apabila dibandingkan dengan penelitian Mohammed Safiqul dan Mayeen Uddin Khandaker[14], implementasi algoritma Viterbi yang dilakukan dengan modulasi BPSK (*Binary Phase Shift Keying*) pada *constraint length* 4 dengan *channel* AWGN hasilnya BER mencapai 10^{-4} pada SNR 7 dB dilihat pada Gambar 4.42, sedangkan pada penelitian ini BER sebesar 10^{-4}, SNR yang dihasilkan sebesar 6 dB. Pada penelitian menggunakan *channel* Rayleigh hasilnya BER mencapai 10^{-4} pada SNR sebesar 13 dB dilihat pada *simulation- Hard* Viterbi di Gambar 4.42. Hasil menggunakan *channel* Rayleigh pada penelitian ini pada BER 10^{-4} diperoleh SNR yang sama yaitu sebesar 13 dB. Penelitian ini juga dilakukan oleh Kanchanna Katta[5] dengan *constraint length* 4 hasilnya lihat pada Gambar 4.42 pada *simulation hard* Viterbi hasilnya pada SNR 6,8 hasil BER mencapai 10^{-5} hasilnya lebih baik dari penelitian Kanchana Katta. Pada penelitian menggunakan *soft* Viterbi hasilnya lebih baik dari *hard* Viterbi ini di sebabkan *hamming distance* pada s*oft* Viterbi diperhitungkan nilai kepresisian

data *path* sebelumnya dan tidak langsung terputus seperti pada *hard* Viterbi. Penggunaan modulasi juga berpengaruh pada *error* data karena semakin tinggi tingkat modulasi yang digunakan, maka akan semakin banyak simbol hasil modulasi yang saling berdekatan. Banyaknya simbol hasil yang berdekatan menyebabkan terjadinya *error* semakin besar. Inilah yang menyebabkan performansi semakin tidak bagus.



Gambar 4.42 Hasil Penelitian yang dilakukan Mohammed Safiqul dan Mayeen Uddin Khandaker [15].



Gambar 4.43 Hasil penelitian yang dilakukan Kanchana kata [5].

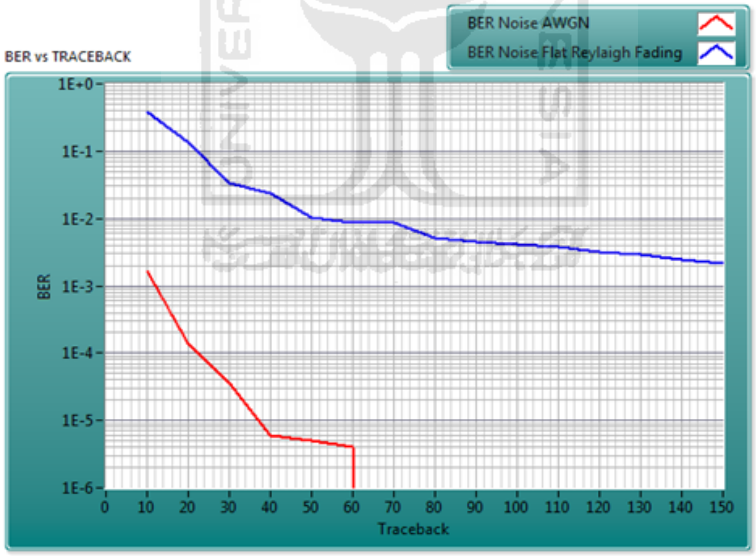
Hasil data yang diperoleh dari pengujian implementasi algoritma Viterbi dengan pengaturan panjang data yang berbeda adalah sebagai berikut:

Tabel 4.2 Hasil data pengujian *TRACEBACK* dan BER dari Implementasi algoritma Viterbi

Pengujian ke	Traceback	BER Channel AWGN	BER Channel Flat Fading Rayleigh
1	10	0,001642	0,380408
2	20	0,000137	0,134455
3	30	0,000036	0,033191
4	40	0,000006	0,023951
5	50	0,000005	0,010462
6	60	0,000004	0,008748
7	70	0	0,00855
8	80	0	0,005099
9	90	0	0,00441
10	100	0	0,004131
11	110	0	0,003773
12	120	0	0,003185
13	130	0	0,002874
14	140	0	0,002409
15	150	0	0,002185

Hasil dari 15 pengujian tersebut dapat dilihat bahwa semakin besar nilai panjang data atau *traceback* yang dikirim pada *receiver* menggunakan

implementasi algoritma Viterbi nilai BER yang dihasilkan semakin kecil, ini disebabkan pada waktu data bit dikirim melalui kanal yang ditambahkan *noise* dimana data bit yang dikirim banyak, maka SNR yang dihasilkan semakin besar sehingga data tidak rentan rusak. Selain itu pada pengujian tugas akhir ini, penggunaan SNR 8,6 dB juga mempengaruhi nilai BER pada pengujian implementasi algoritma Viterbi ini, hasil penggunaan *channel* AWGN dan *channel* flat fading juga sangat berbeda dilihat pada Gambar 4.44 yaitu *channel* AWGN pada traceback 70 hasil BER sudah mencapai $\mathbf{10^{-6}}$ sedangkan pada penggunaan *channel noise* flat fading Rayleigh masih terdapat 8550 data yang *error*. Sehingga penggunaan nilai *traceback* sangat berpengaruh pada hasil data *error*.



Gambar 4.44 Grafik perbandingan BER dengan *TRACEBACK* pada implementasi algoritma Viterbi

BAB V

PENUTUP

5.1 Kesimpulan

Dari serangkaian penelitian yang telah dilakukan, maka didapatkan beberapa kesimpulan sebagai berikut:

1. Pada Tugas Akhir ini, telah berhasil dilakukan pemodelan dan simulasi Implementasi Algoritma Viterbi menggunakan LabVIEW.
2. Hasil pengujian simulasi implementasi algoritma Viterbi menggunakan *traceback* 150 dengan kanal AWGN hasilnya BER mencapai 10^{-6} pada SNR 8,6 dB, sedangkan pada kanal flat fading Raylaigh SNR 8,6 dB dihasilkan BER 10^{-2}.
3. Hasil pengujian simulasi implementasi algoritma Viterbi menggunakan SNR 8,6 dB dengan kanal AWGN, nilai BER mencapai 10^{-6} pada *traceback* 70 sedangkan pada kanal flat fading Rayleigh pada *traceback* 70 dihasilkan BER 10^{-2}

5.2 Saran

Dari serangkaian penelitian yang telah dilakukan, maka didapatkan beberapa saran sebagai berikut:

1. Untuk penelitian selanjutnya, disarankan menggunakan modulasi yang lebih dari satu.
2. Selain itu, pengembangan penelitian ini bisa dilakukan dengan metode *radix 4 butterfly* dan *softdecision decoding.*
3. Untuk penelitian selanjutnya dalam traceback perlu diperhitungkan panjang data yang ganjil.

DAFTAR PUSTAKA

[1] Supriadi, *Analisis Kinerja Forward Eror Correction Menggunakan Algoritma Viterbi Pada Jaringan Wireless Standart IEEE 802.11,* Skripsi, tidak diterbitkan, Departemen Ilmu Komputer Institut Pertanian Bogor, 2007.

[2] Mahyadi Aslam, Arifin, *Visualisasi Kinerja Pengkodean Menggunakan Algoritma Viterbi*, [online] Available at http://www.eepis-its.edu, 2011.

[3] Kurniawan, Nanang, Yoedy Moegiharto, *Perbandingan Rate Kode Konvolusi dan Aplikasinya Pada CDMA*, [online] Available at http://www.eepis-its.edu., 2011.

[4] Adiyanto, Eko Kuncoro, "Perbandimgan Performasi *Convolusi Code* dengan *Convolusi Turbo Code*". Skripsi, tidak diterbitkan, Jurusan Teknik Elektro Universitas Mercu Buana Jakarta, 2009.

[5] Katta. Kanchana, "*Design of Convolution Encoder and Viterbi Decoder using Matlab'', International Journal For Research In Emerging Science and Technology*, Volume- 1, ISSUE -7, 2014.

[6] Sivankar,G,L., Thangarani, "*Performance Analysis Of Viterbi Decoder For Wireless Applications", An International Journal* (ACIJ), Vol.5, No.4, 2014.

[7] Kurniawan. Rendi, *Pembangunan Simulasi dan Analisa Kinerja Optimalisasi Streaming Video Pada Jaringan Wireless dengan Algoritma EAFEC*, Skripsi, tidak diterbitkan, Jurusan Teknik Elektro Universitas Indonesia Jakarta, 2009.

[8] Muhammad Iqbal, "Kode Konvolusi", [online] Available at http://miqbal.staff.telkomuniversity.ac.id/kodekonvolusi/, 2011.

[9] *Wireless LAN Medium Acces Control* (MAC) *and Physical Layer (PHY) Spesifications, IEEE Standart Association*, 2012.

[10] Jabbar, Mohammad Abdul, *Rancang bagun Rangkaian Convolutional Encoder dan Viterbi Decoder Menggunakan DSK TMS320C6713 Berbasis Simulink,* Skripsi, tidak diterbitkan, Jurusan Teknik Elektro Universitas Indonesia Depok, 2008.

[11] Rosita, Erika Kusumasari, Suwardi, Achmad Anshori, "Implementasi Convolutional Code dan Viterbi Decode pada DSK TMS320C6416", Jurnal teknik POMITS vol 2, No. 1, ISSN :2337-3539, 2013.

[12] Pratiwi, Farida, *Implementasi SC-FDMA untuk LTE Arah UPLINK dengan Menggunakan Sofware LabVIEW*, Skripsi, tidak diterbitkan, Jurusan Teknik Elektro Universitas Islam Indonesia Yogyakarta, 2016.

[13] S. Jumianto, "Rancang Bangun Alat Ukur Viskositas Dalam Rangka Pengembangan Modul Praktikum Fisika Dasar". vol. 2, p. 4, 2013.

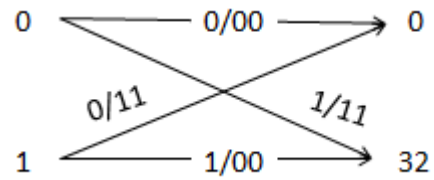
[14] Munadi, *Analisa Forward Kinematic Pada Simulator ARM Robot 5 Dof yang Mengintegrasikan Mikrokontroler Arduino-UNO dan LabView*, [online] Available at http://ejournal.undip.ac.id/index.php/rotasi/article/view/5181/4690, 2013.

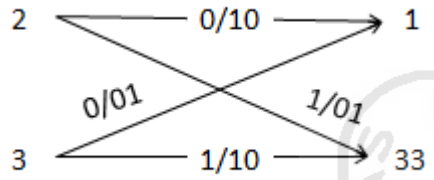
[15] Safiqul, Mohammed, Mayeen Uddin Khandaker,*''Performance comparison of Rate ½ Convolutional Codes with BPSK on Rayleigh and AWGN channels for Memory or Memory less conditi*on'' IJCIT, ISSN 2218-5224, vol 02, issue 02, 2012.

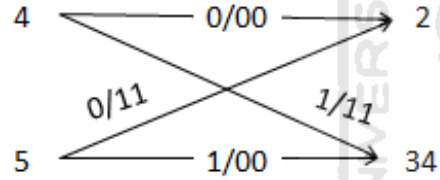
LAMPIRAN

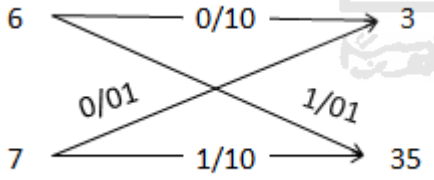
Radix-2 Butterfly pada

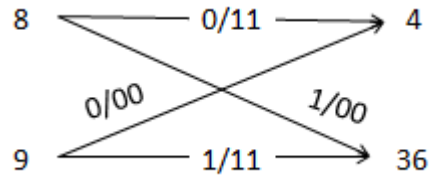
output konvolusi encoder 64 *state*

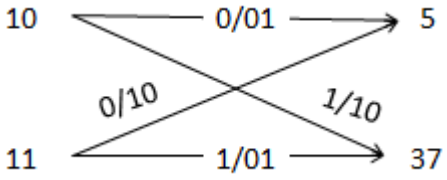


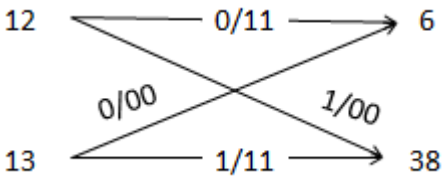


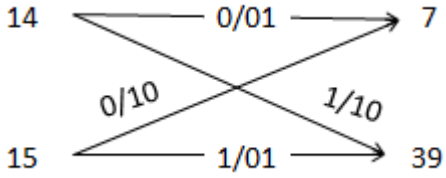


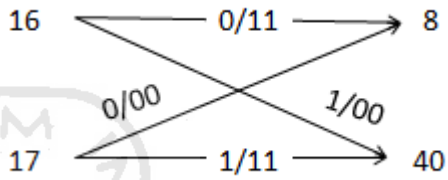


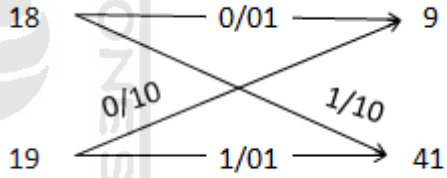


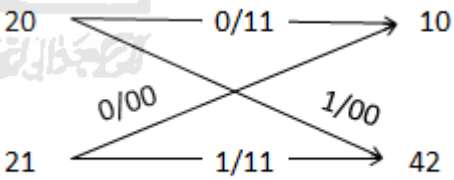


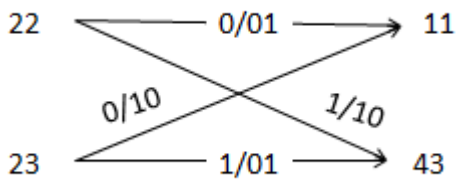


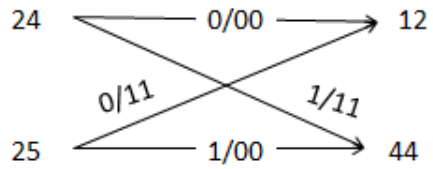


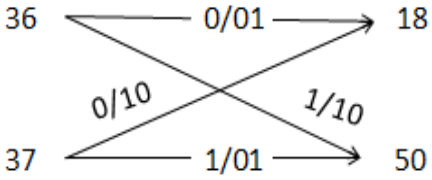


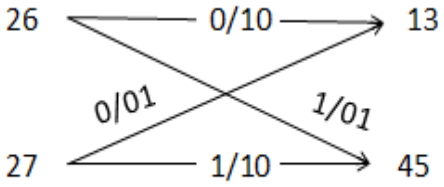


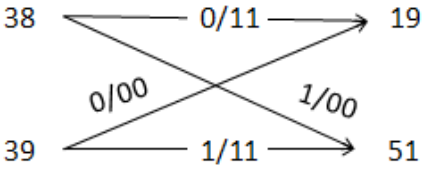


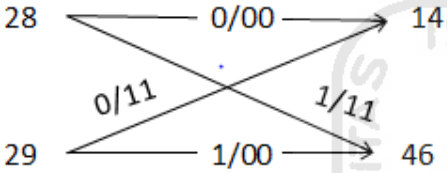


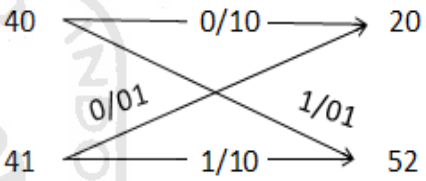

24
0/00
12
0/11
1/11
25
1/00
44

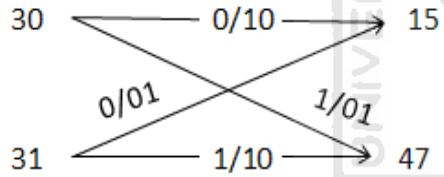

36
0/01
18
0/10
1/10
37
1/01
50

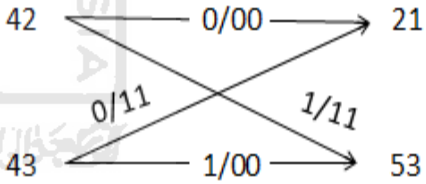

26
0/10
13
0/01
1/01
27
1/10
45

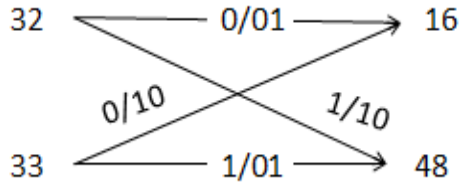

38
0/11
19
0/00
1/00
39
1/11
51

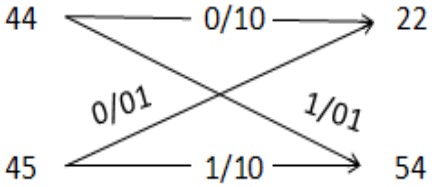

28
0/00
14
0/11
1/11
29
1/00
46

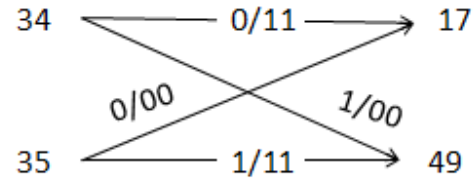

40
0/10
20
0/01
1/01
41
1/10
52

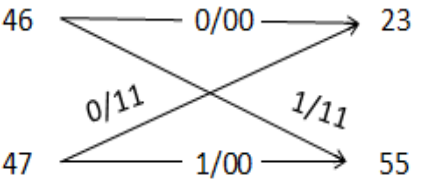

30
0/10
15
0/01
1/01
31
1/10
47

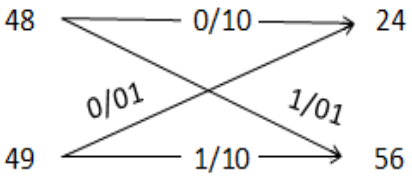

42
0/00
21
0/11
1/11
43
1/00
53

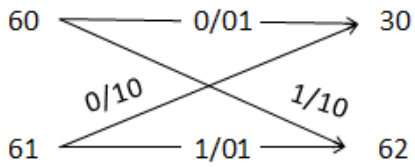

32
0/01
16
0/10
1/10
33
1/01
48

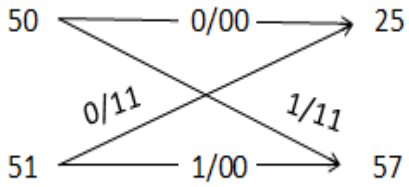

44
0/10
22
0/01
1/01
45
1/10
54

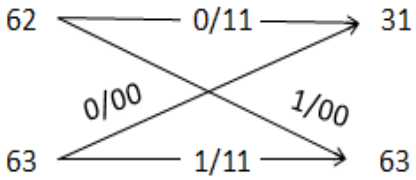

34
0/11
17
0/00
1/00
35
1/11
49

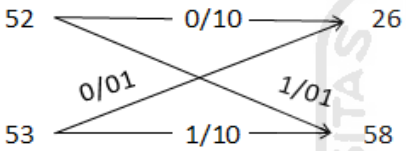

46
0/00
23
0/11
1/11
47
1/00
55

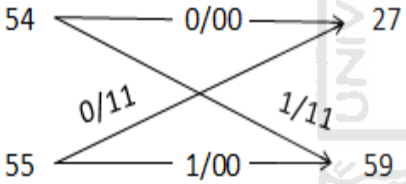

48
0/10
24
0/01
1/01
49
1/10
56

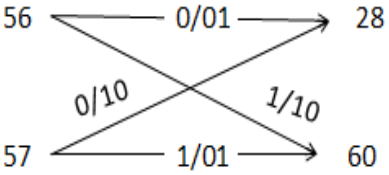

60
0/01
30
0/10
1/10
61
1/01
62

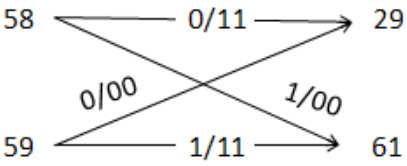

50
0/00
25
0/11
1/11
51
1/00
57


62
0/11
31
0/00
1/00
63
1/11
63


52
0/10
26
0/01
1/01
53
1/10
58


54
0/00
27
0/11
1/11
55
1/00
59


56
0/01
28
0/10
1/10
57
1/01
60


58
0/11
29
0/00
1/00
59
1/11
61

ISLAM
UNIVERSITAS
INDONESIA