PERANCANGAN SIMULATOR RANGKAIAN LOGIKA DENGAN VISUAL C++

Simulator Design Of Digital Logic Gate Using Visual C++

Multazamar Jan1[1], Rismon H. Sianipar2[1], Sultan3[1]

ABSTRAK

Penelitian ini berjudul "Perancangan Simulator Rangkaian Logika Dengan Visual C++" bertujuan untuk membangun sebuah aplikasi simulasi dari Rangkaian Logika Digital menggunakan bahasa pemrograman Visual C agar tercipta sebuah aplikasi yang bersifat interaktif dan mudah dipahami sehingga penggunaanya dapat mengurangi kesalahan didalam melakukan perancangan sebuah rangkaian logika secara nyata.

Simulasi merupakan salah satu cara untuk melakukan teknik uji coba terhadap suatu perencanaan dalam pembuatan suatu alat atau program, maka dibuatlah penelitian yang memfokuskan pada tujuan bagaimana menerapakan metode simulasi pada teori Rangkaian Logika atau Sistem Digital berupa aplikasi berbasis windows. Sehingga dapat dimanfaatkan sebagai referensi pembelajaran sebelum melakukan perancangan atau sebagai acuan dalam membangun sebuah rangkaian logika secara nyata dalam bentuk hardware.

Hasil penelitian menunjukkan bahwa perancangan sebuah aplikasi simulasi rangkaian logika digital dapat diciptakan menggunakan bahasa pemrograman Visual C++ dengan hasil aplikasi yang bersifat interkatif dan mudah dipahami oleh pengguna. Selain itu juga dapat digunakan sebagai bahan dasar didalam mempelajari teori Rangkaian Logika Digital itu sendiri.

Untuk menciptakan sebuah aplikasi yang memiliki wujud atau interface yang menarik dibutuhkan sebuah program berbasis GUI. Sehingga penggunaan Visual C++ sangat membantu didalam merancang sebuah aplikasi simulasi ini.

Kata Kunci *: Simulasi, Rangkaian Logika, dan Visual C++.*

ABSTRACT

This research entitled "Simulator Design Of Digital Logic Gate Using Visual C++" purpose to build simulator application of Digital Logic Gate by using program Visual C in order to create an interactive application with simple concept. Such by use it can resolving problem error when to build digital logic gate obviously

Simulation that is one of some method to trying experiment of a planning to bulid instrument device or program, because of it in this research focus at direction how to apply simulation method on digital logic or digital system theory with basic windows application. Until it can be profit as reference to learning before making plan or as reference in building a hardware digital logic gate obviously.

Result of this research have point that is design application simulator digital logic gate can be product using Visual C++ language program with result a interactive application and simple concept to understanding by user. In other that it can using as basic object in learning digital system theory it self.

To create a application with interesting interface needed a program GUI basically. Then by using Visual C++ is usefull to design this application.

Keywords *: Simulation, Digital Logic Gate, Visual C++*

PENDAHULUAN

Sistem komputer terbagi atas tiga komponen utama yaitu hardware, software dan brainware. Tingkat kemampuan komputer berkembang pesat seiring dengan perubahan level hardware yang dibangun. Namun demikian perkembangan software juga tidak kalah penting. Dikarenakan dengan perangkat software dapat dilakukan perencanaan dan perhitungan terhadap suatu kebutuhan berupa simulasi sebagai tahap awal dalam membangun suatu sistem baru sebagai proses dalam mengiringi perkembangan teknologi.Menurut New Oxford Dictionary (1980) simulasi didefinisikan *sebagai* cara menghasilkan keadaan dalam suatu situasi

[1] Jurusan Teknik Elektro, fakultas Teknik Universitas Mataram, Nusa Tenggara Barat , Indonesia

seperti model untuk tujuan kajian, ujian, atau percobaan. Untuk keempat tujuan tersebut diperlukaan keadaan/perlakuan sistem dinamis dengan menggunakan model komputer sehingga nanti dapat dievaluasi hasilnya dan diperbaiki kinerja sistemnya.

Simulasi merupakan salah satu cara untuk melakukan teknik uji coba terhadap suatu perencanaan dalam pembuatan suatu alat atau program, maka pada kesempatan ini dibuatlah penelitian yang memfokuskan pada tujuan bagaimana menerapakan metode simulasi pada teori Rangkaian Logika atau Sistem Digital berupa aplikasi berbasis windows. Sehingga dapat dimanfaatkan sebagai referensi pembelajaran sebelum melakukan perancangan atau sebagai acuan dalam membangun sebuah rangkaian logika secara nyata dalam bentuk hardware.

Nico Adrian (2002) Judul penelitian adalah perancangan sistem basis data dan simulasi logika IC TTL. Dalam penelitiannya, aplikasi yang digunakan adalah program Borland Delphi 5. Pada penelitian ini menghasilkan aplikasi yang dapat digunakan sebagai penyedia informasi untuk mengetahui jenis IC TTL yang telah ditentukan, data gerbang logika dalam IC, data tabel keluarannya, data kaki IC dan data keluaran dari masukan yang diberikan kedalam IC.

Mohamad Fatchur Rohman (2006) Judul penelitian adalah simulasi gerbang logika menggunakan bahasa pemrograman Borland Delphi 7.0. Dalam penelitiannya, dengan menggunakan bahasa pemrograman Borland Delphi dapat menghasilkan sebuah program aplikasi simulasi gerbang logika dengzo2jjan 8 variabel masukan (A, B, C, D, E, F, G, dan H), dan 3 macam operator digital dengan urutan prioritas derajat opersi logika; not (negasi, ’), and (titik, .), kemudian or (plus, +).

Sistem bilangan. Suatu sistem bilangan adalah suatu himpunan aturan, nama, dan lambang yang digunakan untuk mewakili bilangan. Sistem bilangan itu didefinisikan oleh cara menyajikan suatu urutan bilangan nyata secara wajar dengan pertolongan lambang bilangan atau angka.

Setiap bilangan asli N dalam notasi menurut kedudukan itu secara unik (tunggal) diwakili oleh pernyataan berikut:

$$N = a_n R^n + a_{n-1} R^{n-1} + \cdots + a_1 R^1 + a_0 R^0$$

$$\sum_{i=0}^{n} a_i R^i$$

dengan

N : himpunan angka yang mewakili suatu bilangan;

R : radiks atau dasar sistem bilangan;

n+1: banyaknya angka dalam bilangan itu;

a_i : angka yang bernilai dari 0 sampai dengan R-1 (i=0,1,2,…,R-1)

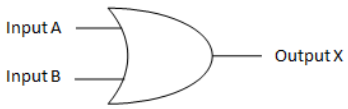
Sandi biner. Kebutuhan manusia untuk menggunakan bilangan desimal tidak mengubah sifat rangkaian elektronika digital. Peralatan itu masih tetap mengolah sinyal biner. Penyelesaian jalan tengah untuk masalah tersebut adalah dengan memanfaatkan **sandi biner** (*binary code*). Bilangan desimal diwakili dalam sistem digital dengan serangkaian bit dan berbagai kombinasi bit itu mewakili bilangan desimal yang diperlukan.

Tabel 1 Sandi biner angka desimal

Bilangan desimal	BCD 8421	XS-3	84-2-1	2421	Bikuiner 5043210	Sandi Gray
0	0000	0011	0000	0000	0100001	0000
1	0001	0100	0111	0001	0100010	0001
2	0010	0101	0110	0010	0100100	0011
3	0011	0110	0101	0011	0101000	0010
4	0100	0111	0100	0100	0110000	0110
5	0101	1000	1011	1011	1000001	0111
6	0110	1001	1010	1100	1000010	0101
7	0111	1010	1001	1101	1000100	0100
8	1000	1011	1000	1110	1001000	1100
9	1001	1100	1111	1111	1010000	1101
10	-	-	-	-	-	1111
11	-	-	-	-	-	1110
12	-	-	-	-	-	1010
13	-	-	-	-	-	1011
14	-	-	-	-	-	1001
15	-	-	-	-	-	1000

Gerbang-gerbang logika dasar. Gerbang logika adalah blok bangunan dasar untuk membentuk rangkaian elektronika digital yang digambarkan dengan simbol-simbol tertentu yang telah ditetapkan. Sebuah gerbang logika memiliki beberapa masukan tetapi hanya memiliki satu keluaran. Keluarannya akan *HIGH* (1) atau *LOW* (0) tergantung pada level digital pada terminal masukan. Dengan menggunakan gerbang-gerbang logika, dapat dibuat rancangan atau desain suatu sistem digital yang akan dikendalikan level masukan digital dan menghasilkan sebuah tanggapan keluaran tertentu berdasarkan rancangan berdasarkan rancangan rangkaian logika itu sendiri.

Gerbang OR. Gerbang logika OR memiliki dua atau lebih isyarat masukan (*input*) tetapi hanya satu isyarat keluaran (*output*). Jika salah satu isyarat masukan 1, maka sinyal keluarannya adalah 1. Simbol atau lambang dari gerbang logika OR dinyatakan pada gambar dibawah ini.



Gambar 1 Gerbang OR dengan dua input (masukan)

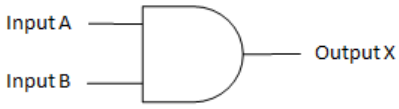
Dalam persamaan aljabar Boole, ini dapat ditulis sebagai

$$X = A + B$$

Tabel 2 Kebenaran gerbang OR dua input

Input		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

Gerbang AND. Pada gerbang AND yang memliki dua atau lebih masukan akan memiliki keluaran 1 (*HIGH*) jika semua masukannya dalam keadaan 1 (*HIGH*). Jika salah satu atau semua dari isyarat masukannya itu 0 maka outputnya akan 0. Simbol dari gerbang AND dinyatakan pada gambar dibawah ini.



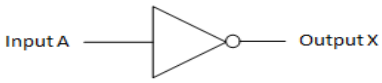
Gambar 2 Gerbang AND dengan dua input dalam persamaan aljabar Boole, ini dapat ditulis

$$X = A \,.\, B$$

Tabel 3 Kebenaran gerbang AND dua input

Input		Output
A	B	X
0	0	0
0	1	0
1	0	0
1	1	1

Gerbang NOT. Gerbang logika NOT adalah sebuah gerbang logika yang memiliki hanya satu input dan hanya satu output, fungsinya sebagai pembalik.



Gambar 3 Simbol dari inverter

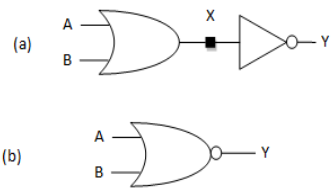
Persamaan aljabar Boole untuk sebuah inverter ditulis:

$$X = \overline{A}$$

Tabel 4 Kebenaran gerbang AND dua input

Input A	Output X
0	1
1	0

Gerbang NOR. Gerbang logika NOR adalah sebuah gerbang logika kombinasi yang sama operasinya dengan gerbang logika dasar OR, tetapi bagian outputnya dibalik dengan gerbang inverter (NOT). Gerbang logika NOR terdiri dari kombinasi atau gabungan dari gerbang logika NOT dan OR.

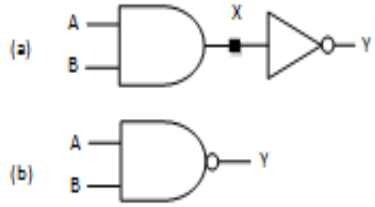


Gambar 4 Lambang gerbang logika NOR

Tabel 5 Kebenaran gerbang NOR dengan dua input

Input		Output
A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

Gerbang NAND. Gerbang logika NAND terdiri dari kombinasi atau gabungan dari gerbang logika NOT dan AND. Sering disebut dengan gerbang logika NAND. Gerbang logika NAND juga memiliki struktur logika yang sama dengan gerbang logika AND, yaitu memiliki dau masukan atau lebih tetapi hanya memiliki satu keluaran saja, namun hasil keluarannya terbalik dengan AND karena memiliki gerbang tambahan yaitu inverter (NOT).



Gambar 5 Lambang gerbang logika NAND

Persamaan aljabar Boole untuk gerbang logika NAND dengan dua masukan adalah:

$$Y = \overline{A \,.\, B}$$

Tabel 6 Kebenaran gerbang NOR dengan dua Input

Input		Output
A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

Gerbang XOR. Gerbang XOR (dari kata EXCLUSIVE-OR) akan memberikan hasil keluaran 1 jika masukan-masukannya mempunyai keadaan yang berbeda. Atau dengan kata lain keluaran pada gerbang XOR merupakan penjumlahan biner dari masukannya.

Gambar 6 Lambang gerbang logika XOR

Untuk lebih jelasnya perhatikan tabel kebenaran berikut ini:

Tabel 7 Kebenaran gerbang XOR dengan dua input

Input		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Persamaan aljabar Boolenya adalah sebagai berikut:

$$X = \bar{A}B + A\bar{B}$$

Gerbang XNOR. Gerbang logika XNOR setara dengan gerbang logika XOR yang ditambah dengan gerbang logika NOT (*Inverter*). Dengan adanya pembalikan (*inverse*) pada sisi keluarannya, maka tabel merupakan komplemen dari tabel kebenaran gerbang XOR.

Gambar 7 Lambang untuk gerbang logika XNOR

Untuk lebih jelasnya perhatikan tabel kebenaran gerbang logika XNOR berikut ini:

Tabel 8 Kebenaran gerbang XOR dengan dua Input

Input		Output
A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

Persamaan aljabar Boole untuk gerbang logika XNOR dinyatakan dengan:

Aljabar boole. Hukum aljabar Boole pada dasarnya tidak jauh berbeda dengan aljabar biasa. Beberapa dasar aljabar Boole memiliki sifat yang sama dengan aljabar biasa, contohnya adalah kepemilikannya atas sifat komutatif, asosiatif, dan distributif. Namun demikian, dalam beberapa hal aljabar Boole memiliki perbedaan dengan aljabar biasa. Perbedaan inilah yang membuat aljabar Boole sangat berguna dalam perancangan teknik digital, misalnya dalam melakukan penyederhanaan rangkaian logika yang rumit dan kompleks menjadi rangkaian logika yang lebih sederhana, agar biaya produksi dapat menjadi jauh lebih murah.

Tabel 9 Ringkasan perbedaan antara aljabar biasa dengan aljabar Boole.

Gerbang logika	Aljabar biasa	Aljabar Boole
Operasi gerbang OR	$A+0=1$	$A+0=A$
	$A+1=A+1$	$A+1=1$
	$A+A=2A$	$A+A=A$
	$A+\bar{A}=0$	$A+\bar{A}=1$
Operasi gerbang AND	$A.0=0$	$A.0=0$
	$A.1=A$	$A.1=A$
	$A.A=A^2$	$A.A=A$
	$A.\bar{A}=-A^2$	$A.\bar{A}=0$

Tabel 10 Ringkasan hukum-hukum aljabar Boole

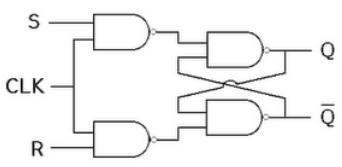
	Gerbang OR	Gerbang AND	Keterangan
1.	$A+B=B+A$	$A.B=B.A$	hukum komutatif
2.	$A+(B+C)=(A+B)+C$	$A.(B.C)=(A.B).C$	hukum asosiatif
3.	$A.(B+C)=A.B+A.C$	$A.(B+C)=A.B+A.C$	hukum distributive
4.	$A+0=A=A.1$	$A.1=A=A+0$	teorema dualitas
5.	$A.(B+C)=A.B+A.C$	$A+B.C=(A+B).(A+C)$	teorema dualitas
6.	$A+0=A$	$A.0=0$	hukum aljabar boole
7.	$A+1=1$	$A.1=A$	jika dengan 1
8.	$A+A=A$	$A.A=A$	identitas
9.	$A+\bar{A}=1$	$A.\bar{A}=0$	jika dengan pembaliknya
10.	$A=\bar{\bar{A}}$	$A=\bar{\bar{A}}$	inverter ganda
11.	$\overline{A+B}=\bar{A}.\bar{B}$	$\overline{A.B}=\bar{A}+\bar{B}$	hukum de morgan
12.	$A+A.B=A$	$A.(A+B)=A$	aljabar boole dengan jumlah hasil kali
13.	$A+\bar{A}.B=A+B$	$A.(\bar{A}+B)=A.B$	aljabar Boole dengan jumlah hasil kali

Flip-flop. Dengan menggunakan gabungan gerbang-gerbang logika menjadi suatu gerbang logika kombinasional, dan kemudian diumpan-balikkan (*feedback*), dapat dibangun suatu rangkaian logika yang dapat menyimpan data. Rangkaian logika inilah yang disebut dengan piranti atau rangkaian Flip-flop.

Flip-flop merupakan piranti yang memiliki dua keadaan stabil yang dinyatakan dalam sistem biner yaitu 0 atau 1. Piranti ini akan tetap bertahan pada salah satu dari dua keadaan itu sampai ada pemicu yang membuatnya berganti keadaan.

S-R Flip-flop. Flip-flop SR terdetak harus menyesuaikan diri dengan sinyal pendetak atau menyinkronkan diri dengan sinyal pendetak. Apabila sinyal pendetak masukan pada logika 0, maka data yang masuk pada S dan R tidak akan ditanggapi atau diproses oleh flip-flop, sehingga keluaran Q tetap tidak berubah.

Jika sinyal pendetak berubah dari 0 menjadi 1, seketika itu juga masukan Set atau Reset akan ditanggapi, sehingga keluaran Q berubah. Pengoperasian flip-flop SR terdetak disebut secara serempak atau sinkron (*Synchronous*). Dinamakan sinkron, karena bekerjanya menyesuaikan dengan irama waktu sinyal pendetak.



Gambar 8 Rangkaian SR Flip-flop terdetak menggunakan NAND

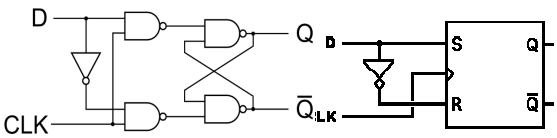
Berikut ini adalah tabel kebenaran untuk rangkaian SR Flip-flop terdetak

Tabel 11 Kebenaran rangkaian SR Flip-flop terdetak

Clock	Set	Reset	Q
0	x	X	nc
1	0	0	nc
1	0	1	1
1	1	0	0
1	1	1	*

nc : not change
* : pacu (keadaan terlarang)

D Flip-flop S-R terdetak dapat dimodifikasi seperti terlihat pada Gambar 9 untuk menambahkan tunda satu bit pada sebuah jalur data masukan, D. Sebuah pembalik ditambahkan pada masukan R sehingga masukan R merupakan komplemen dari masukan S. Dalam keadaan ini, flip-flop selalu berada pada keadaan D=1 (set) atau D=0 (reset).

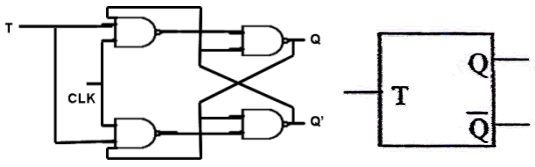


Gambar 9 Flip-flop tipe D, dan simbolnya.

Tabel 12 Kebenaran rangkaian D Flip-flop terdetak

Detak	D_n (t_n)	Q_{n+1} (t_{n+1})
0	X	Tetap
1	0	0
1	1	1

T Flip-flop. Flip-flop tipe T mempunyai satu masukan T (*toggle*) yang akan menyebabkan berubahnya keadaan keluaran pada setiap pulsa masukan. Flip-flop tipe T dapat dibuat dengan mengumpan balik Q ke R dan dari ke S, seperti pada gambar berikut.



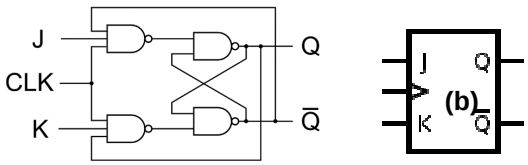
Gambar 10 Flip-flop tipe T, dan simbolnya.

Berikut ini adalah tabel kebenaran untuk rangkaian SR Flip-flop terdetak

Tabel 13 Kebenaran rangkaian T Flip-flop terdetak

Clock	T	Q	Q(t+1)
0	X	X	Nc
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

J-K Flip-flop. Flip-flop J-K mirip dengan flip-flop tipe-T dengan dua tambahan masukan seperti terlihat pada Gambar 11. Masukan tambahan ini disebut sebagai masukan J dan masukan K untuk membedakannya dengan S dan R. Konstruksi J-K menyediakan flip-flop universal yang dapat diprogram.



Gambar 11 (a) Flip-flop J-K, dan (b) simbolnya

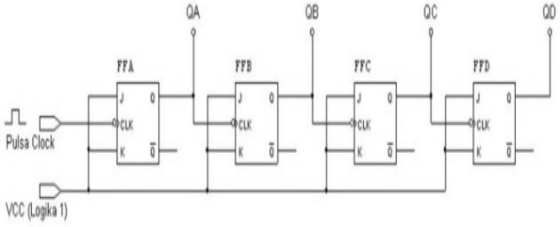
Tabel 14 Kebenaran rangkaian J-K Flip-flop

Detak (Ck)	J	K	Q_{n+1}
0	X	X	Q_n tetap
1	0	0	Q_n tetap
1	1	0	1
1	0	1	0
1	1	1	Toggle

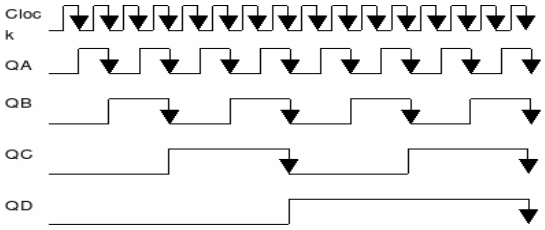
Register. Dalam elektronika digital seringkali diperlukan penyimpanan data sementara sebelum data diolah lebih lanjut. Elemen penyimpan dasar adalah flip-flop. Setiap flip-flop menyimpan sebuah bit data. Sehingga, untuk menyimpan kata *n*-bit, diperlukan *n* buah flip-flop yang disusun sedemikian rupa dalam bentuk register geser. Data biner dapat dipindahkan secara seri atau parallel. Dalam mode seri, bit-bit dipindahkan secara berurutan satu persatu. Dalam mode parallel, bit-bit dipindahkan secara serempak sesuai dengan cacah jalur parallel secara sinkron dengan sebuah pulsa dari sistem detak. Ada empat cara dimana register geser dapat digunakan untuk menyimpan dan memindahkan data dari satu bagian ke bagian system yang lain:

1. Masukan seri ke keluaran paralel (SIPO)
2. Masukan seri ke keluaran seri (SISO)
3. Masukan paralel ke keluaran seri (PISO)
4. Masukan paralel ke keluaran paralel (PIPO)

Counter. Bistabil atau flip-flop dapat digunakan sebagai pembagi biner (membagi 2) untuk membentuk suatu pencacah (*counter*). Pencacah adalah sekelompok flip-flop yang disusun sedemikian rupa sehingga menunjukkan cacah pulsa total yang diumpankan pada masukan. Pencacah dapat dikelompokkan menjadi dua kategori besar: tak sinkron dan sinkron.



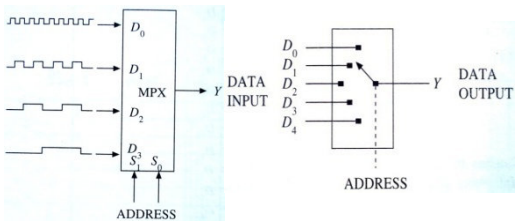
Gambar 1.12 Pencacah *ripple-through* (tak sinkron)



Gambar 13 Diagram pewaktuan dari pencacah *ripple-through*

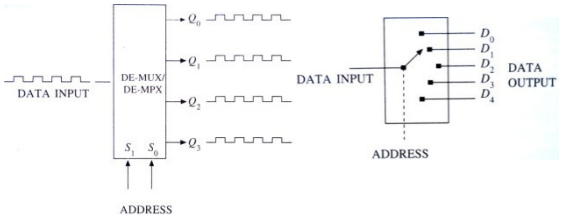
Multiplexer. *Multiplexer* berarti dari banyak kedalam (menjadi) satu. Sebuah multiplexer adalah rangkaian yang memiliki banyak masukan tetapi hanya satu keluaran. Dengan menggunakan sinyal kendali dapat diatur penyaluran masukan tertentu menuju keluarannya. Sinyal kendali ini akan mengatur bagian mana atau alamat (*Address*) mana yang akan diaktifkan atau dipilih. Piranti multiplexer atau disebut juga pemilih data (*data selector*) adalah sebuah rangkaian logika yang menerima beberapa masukan dan hanya satu diantaranya yang dilewatkan ke keluaran pada suatu waktu.

Lambang dari Multiplexer adalah sebagai berikut :



Gambar 14 Simbol logika piranti Multiplexer 4 ke-1

Demultiplexer. Sebuah rangkaian Demultiplexer memiliki banyak keluaran. Dengan menggunakan sinyal kendali, dapat diatur penyaluran masukan tertentu pada keluarannya. Sinyal kendali ini akan mengatur bagian mana atau alamat mana yang akan diaktifkan atau dipilih. Piranti demultiplexer disebut juga distribusi data atau penyalur data (*data distributor*) yaitu sebuah rangkaian logika yang menerima hanya satu masukan data dan melewatkan satu diantara beberapa keluaran.Lambang dari Demultiplexer adalah sebagai berikut:



Gambar 15 Simbol logika piranti Demultiplexer 1 ke-4

Visual Studio. Microsoft Visual Studio adalah sebuah *IDE* dari Microsoft. Hal ini digunakan untuk mengembangkan konsol dan aplikasi antarmuka pengguna grafis bersama dengan aplikasi *Windows Forms*, situs web, aplikasi web, dan layanan web untuk semua *platform* yang didukung oleh *Microsoft Windows, Windows Mobile, Windows CE,. NET Framework, NET Compact Framework* dan *Microsoft Silverlight*. Visual Studio mencakup kode editor pendukung *IntelliSense* serta *refactoring code*.

Visual Studio mendukung bahasa pemrograman berbeda, yang memungkinkan kode editor dan debugger untuk mendukung hampir semua bahasa pemrograman, memberikan layanan bahasa spesifik, seperti C / C + + (melalui Visual C + +), VB.NET (melalui Visual Basic NET.), C # (melalui Visual C #), dan F # (pada Visual Studio 2010). Dukungan untuk bahasa lain seperti M, Python, dan Ruby antara lain tersedia melalui layanan bahasa yang diinstal secara terpisah. Ini juga mendukung XML / XSLT, HTML / XHTML, JavaScript dan CSS. Visual Studio juga menyediakan bahasa pemrograman yang terbatas bagi pengguna, seperti: Microsoft Visual Basic, Visual J #, Visual C #, dan Visual C + +.

METODE PENELITIAN

Pada tahapan ini dilakukan beberapa metode sebagai penunjang didalam

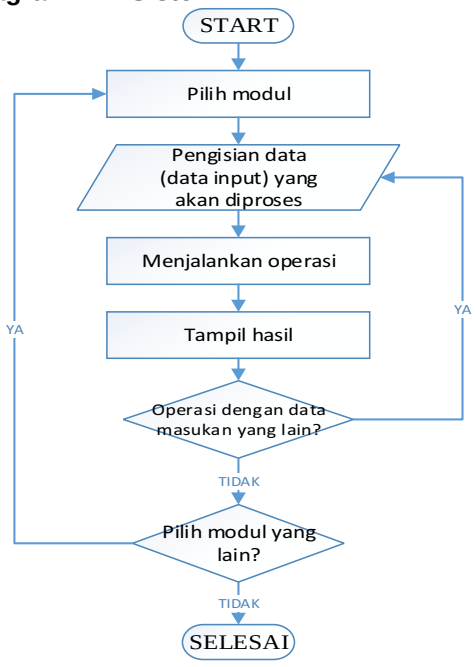
pembuatan program aplikasi simulator rangkaian logika. Diantaranya yaitu :

1. Melakukan observasi mengenai bagaimana proses pelaksanaan praktikum rangkaian logika dilakukan dengan perangkat hardware yang ada.
2. Melakukan studi literature terkait materi-materi rangkaian logika atau sistem digital untuk memperoleh nilai kebenaran dari hasil keluaran dari rangkaian berdasarkan perhitungan yang bernilai pasti dan terbukti.
3. Membandingkan nilai antara hasil praktikum dengan nilai perhitungan berdasarkan teori yang ada.

Setelah menjalankan tahapan-tahapan diatas, penulis menemukan bahwa dalam mempelajari rangkaian logika atau sistem digital sering ditemukan kesalahan mengenai hasil keluaran dari rangkaian. Antara hasil praktikum dengan hasil perhitungan berdasarkan teori sering terjadi error atau tidak sesuai. Ini dimungkinkan oleh adanya *human error* ataupun kualitas hardware yang digunakan kurang baik.

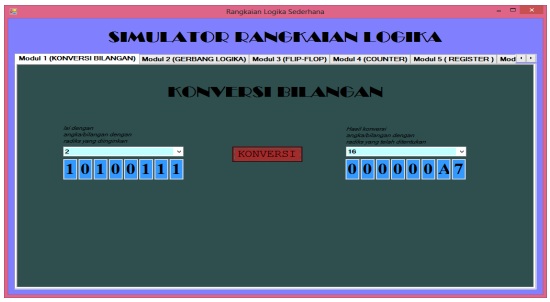
Atas dasar inilah penulis membuat sebuah aplikasi simulasi sebagai sarana pendukung keputusan dalam menenetukan nilai pasti dari hasil keluaran pada saat melaksanakan praktikum. Sehingga pengguna lebih memahami secara aplikatif tentang rangkaian logika itu sendiri.

Diagram Alir Sistem



Gambar 16 diagram alir sistem

Bentuk Perancangan Interface



Gambar 16 Perancangan interface program

HASIL DAN PEMBAHASAN

Berikut merupakan wujud tampilan dari aplikasi Simulator Rangkaian Logika

Modul 1 (KONVERSI BILANGAN). Pada modul ini ditampilkan proses konversi dari beberapa bilangan dengan basis yang berbeda-beda. Untuk lebih jelasnya dapat dilihat pada tampilan berikut :

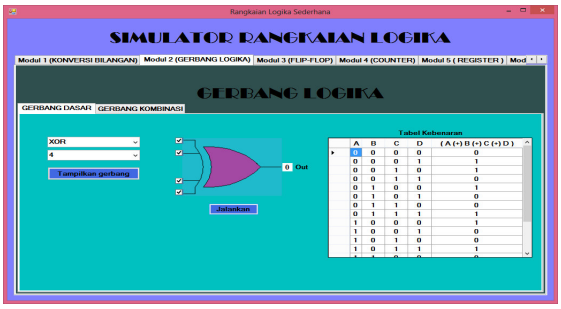
Gambar 17 Contoh proses konversi antara dua buah bilangan

Dari contoh diatas terlihat proses konversi diantara dua buah bilangan yaitu bilangan dengan basis 2 sebagai bilangan yang akan dikonversi dengan bilangan dengan basis 16 sebagai hasil dari proses konversi tersebut. Pada kolom bilangan untuk basis 2 diisi dengan nilai 10100111. Dengan menekan tombol “KONVERSI” maka tampil hasilnya pada kolom bilangan dengan basis 16 yaitu A7. Hasil tersebut telah diuji dan sesuai dengan perhitungan manual berdasarkan teori-teori yang telah dipelajari.

Modul 2 (GERBANG LOGIKA). Pada modul ini ditunjukkan beberapa proses bagaimana satu atau lebih gerbang logika menampilkan nilai keluaran berdasarkan nilai input yang diberikan. Ada dua macam gerbang logika yang ditampilkan berdasarkan proses input dan output-nya saja yaitu gerbang logika tunggal dan kombinasional. Untuk gerbang logika tunggal terdiri dari AND, OR, NOT,

NAND, NOR, XOR, dan XNOR. Sedangkan gerbang kombinasional merupakan gabungan dari beberapa gerbang tunggal untuk menghasilkan gerbang tertentu yang memiliki nilai keluaran yang sama.

Pada tab GERBANG DASAR menampilkan beberapa jenis gerbang logika dasar seperti AND, OR, NOT, NAND, NOR, XOR, dan XNOR dengan jumlah input yang berbeda-beda (maksimal 4 buah masukan). Setelah menentukan jenis gerbang dan berapa jumlah masukan kemudian meng-klik tombol "Tampilkan" maka tampil seperti berikut :



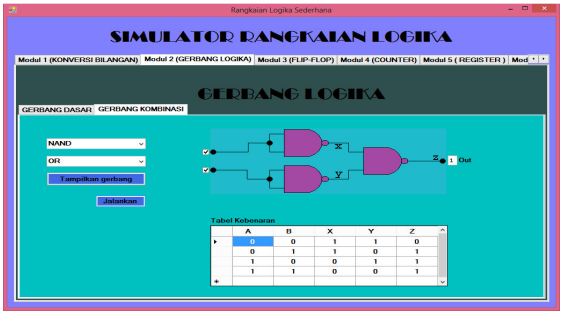
Gambar 18 Contoh tampilan gerbang logika dasar

Gerbang kombinasi bertujuan untuk memberikan alternatif dalam menciptakan sebuah gerbang logika baru berdasarkan kombinasi dari beberapa gerbang yang berbeda, dimana nilai keluaran antara gerbang asli dengan gerbang kombinasinya memiliki nilai yang sama.

Jenis-jenis kombinasi yang bisa dibuat adalah seperti :

- Kombinasi NAND menghasilkan OR, AND, NOT, dan NOR
- Kombinasi NOR menghasilkan OR, AND, NOT, dan NAND
- Kombinasi NOT AND menghasilkan NOR, dan
- Kombinasi NOT OR menghasilkan NAND

Berikut contoh tampilan dari salah satu gerbang kombinasi yang ada :

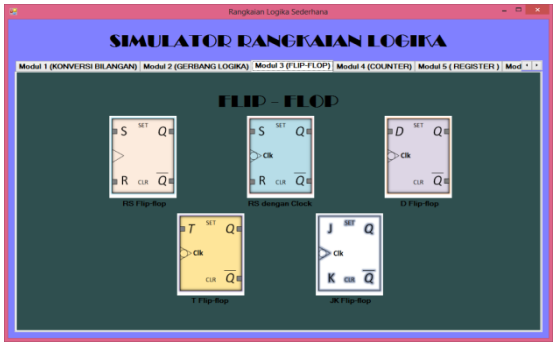


Gambar 19 Contoh tampilan gerbang kombinasi

Dengan memilih kombinasi gerbang tertentu dan persamaan dari gerbang kombinasi tersebut kemudian menekan tombol "Tampilkan gerbang", maka tampil bentuk dari rangkaian gerbang yang dimaksud. Seperti pada contoh diatas.

Pada tampilan diatas terlihat bentuk dari rangkaian kombinasi gerbang NAND untuk menghasilkan sebuah gerbang yang memiliki nilai keluaran yang sama dengan gerbang OR. Untuk membuktikan kebenaran nilai keluaran dapat dilihat pada tabel kebenaran yang telah disesuaikan dengan perhitungan manual berdasarkan teori-teori yang ada. Untuk menampilkan gerbang kombinasi yang lain dapat dilakukan dengan terlebih dahulu melakukan pemilihan kombinasi gerbang yang ada pada menu kotak kombo.

Modul 3 (FLIP-FLOP). Pada modul 3 ini diberikan menu pilihan beberapa jenis Flip-flop yang sudah ada berdasarkan data-data dari teori yang sudah dipelajari, seperti; Set Reset Flip-flop (S-R FF), Data Flip-flop (D FF), Toggle Flip-flop (T FF), dan Master Slave Flip-flop (J-K FF). Flip-flop sendiri merupakan salah satu dari jenis rangkaian gerbang skuensial karena memiliki kemampuan menyimpan data. Dimana nilai keluaran atau output yang tersimpan dapat ditentukan berdasarkan jenis masukan yang diberikan pada flip-flop tersebut. Sebagai tampilan awal dari modul ini dapat dilihat pada gambar berikut :

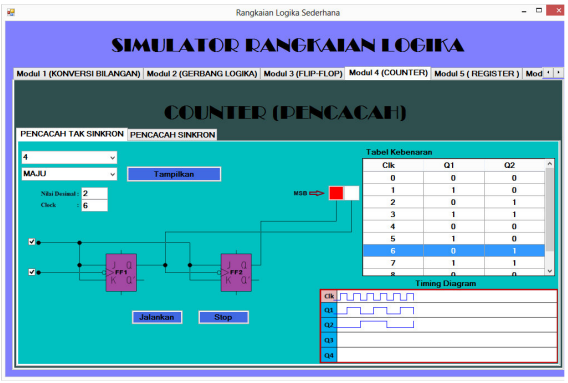


Gambar 20 Tampilan awal modul 3 (Flip-flop)

Modul 4 (COUNTER). Pada tab Pencacah Tak Sinkron terdapat dua jenis yaitu pencacah maju dan pencacah mundur. Dimana jenis-jenisnya dapat ditentukan dengan memilih pada kotak kombo yang tersedia.

Untuk kombo MODULO memberikan pilihan modulo yang akan ditampilkan.

Sedangkan kombo JENIS untuk menentukan jenis pencacah yang diinginkan apakah maju ataukah mundur. Pada tab ini diberikan beberapa contoh pencacah tak sinkron dengan jenis modulo yang berbeda, diantaranya yaitu modulo 4, modulo 8, modulo 10, dan modulo 16. Masing-masing modulo juga ditampilkan kedalam dua jenis yaitu pencacah maju dan pencacah mundur.

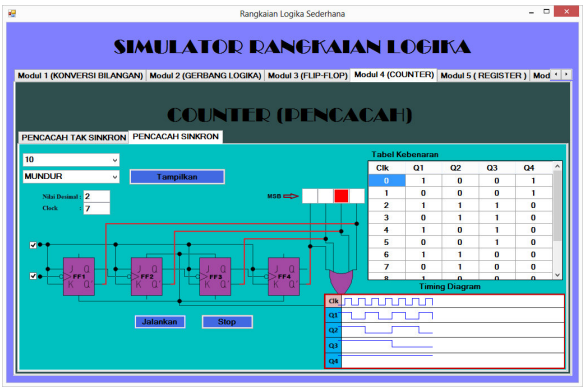


Gambar 20 Tampilan dengan clock dan masukan J-K dalam kondisi aktif

Tampilan diatas memberikan gambaran dari rangkaian pencacah tak sinkron modulo 4 dengan jenis pencacah maju berikut dengan tabel kebenaran dan timing diagramnya. Dengan menekan tombol “Jalankan” maka program akan memberikan respon hasil keluaran yang ditandai dengan perubahan warna pada kotak yang berada di ujung rangkaian sebagai gambaran dari led/indicator output pada rangkaian. Warna merah berarti keluaran bernilai 1 atau ON, sedangkan keluaran warna putih berarti keluaran bernilai 0 atau OFF. Proses mencacah hanya dapat dijalankan jika masukan J-K dan Clock dalam kondisi aktif/ON. Timing diagram akan terlihat mengikuti setiap detak clock yang berjalan.

Setelah dijalankan, program mulai mencacah. Tampilan diatas menunjukkan bahwa pada clock ke 6, nilai keluaran dari pencacah modulo 4 jenis maju bernilai 2 yang ditunjukkan dengan perubahan warna led yang berarti 10 dalam bentuk biner. Tabel kebenaran menunjukkan hasil yang sesuai dengan keluaran program yang sedang berjalan.

Pada tab Pencacah Sinkron terdapat kesamaan dengan Pencacah Tak Sinkron yaitu dalam memberikan tampilan hasil keluarannya. Perbedaan diantara keduanya yaitu pada bentuk rangkaiannya saja.



Gambar 21 Tampilan program setelah dijalankan

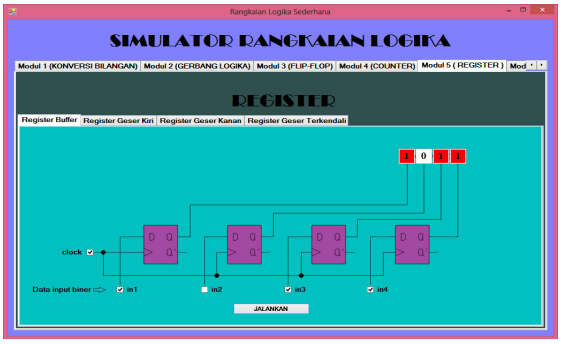
Seperti pada tampilan Pencacah Tak Sinkron, pada Pencacah Sinkron juga menggunakan cara yang sama didalam menampilkan rangkaian yang diinginkan. Yaitu dengan memilih modulo dan jenis dari rangkaian pencacah sinkron maka setelah menekan tombol “Tampilkan” akan tampil bentuk rangkaian dan masukan maupun keluaran yang dilengkapi dengan tabel kebenaran dan timing diagram yang telah disesuaikan dengan teori yang ada.

Tampilan diatas menunjukkan bentuk dari salah satu rangkaian pencacah sinkron yaitu modulo 10 jenis mundur yang dilengkapi dengan masukan clock dan masukan J-K kemudian ada 4 buah led keluaran yang merupakan gambaran dari runtun biner. Terdapat pula sebuah tabel kebenaran yang dijadikan sebagai acuan dalam menguji hasil keluaran dari rangkaian yang ada. Sedangkan untuk timing diagram memperlihatkan bagaimana disetiap detak clock terjadi perubahan bentuk sinyal yang semakin menurun hingga mencapai batas dimana seluruh keluaran berada pada kondisi 0.

Dengan memberikan masukan J-K dan clock dalam kondisi aktif, kemudian menekan tombol “Jalankan”, maka program mulai melakukan proses mencacah. Dimana pada tampilan diatas terjadi proses mencacah mundur dari $1001_2 = 9_{10}$ sampai dengan $0000_2 = 0_{10}$ dan terus berulang begitu seterusnya.

Modul 5 (REGISTER). Pada modul 5 ini memberikan gambaran tentang beberapa macam register yang dibangun dari beberapa buah Flip-flop D. Rangkain register sederhana yang ditampilkan diantaranya yaitu; Register Buffer, Register Geser Kiri, Register Geser Kanan, dan Register Geser Terkendali.

Salah satu bentuk dari rangkaian register adalah register buffer, seperti pada tampilan berikut:



Gambar 22 Tampilan Register Buffer saat dijalankan

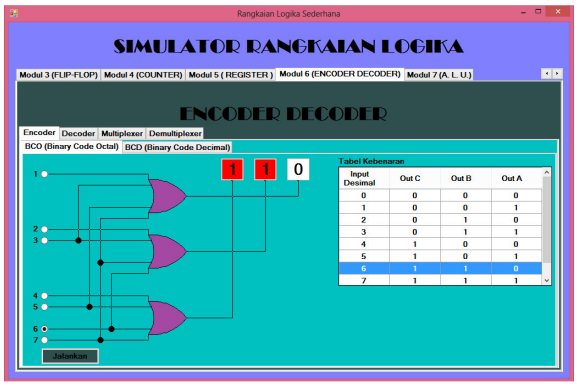
Pada tampilan diatas terdapat 4 buah masukan data yang diatur secara parallel ke setiap masukan D pada masing-masing flip-flop dengan hasil keluaran Q dihasilkan dari setiap flip-flop tersebut. Ini membuktikan bahwa Register Buffer merupakan salah satu bentuk dari rangkaian register model PIPO (*Parallel In Parallel Out*) artinya masukan maupun keluaran dari register tersebut diatur secara parallel. Sedangkan untuk masukan clock diberikan secara serentak pada masing-masing flip-flop untuk mendapatkan nilai keluaran yang bersamaan.

Disebabkan register buffer tersusun dari rangkaian D Flip-flop maka setiap nilai masukan yang diberikan akan disimpan kedalam memori register sesuai dengan nilai masukannya tanpa ada perubahan.

Modul 6 (ENCODER dan DECODER). Modul 6 ini menunjukkan beberapa macam rangkaian penyandi yang tersusun dari beberapa kombinasi gerbang sederhana seperti AND, OR, dan NOT. Dari setiap rangkaian penyandi yang ada, masing-masing penyandi diberikan tampilan tabel kebenaran yang dijadikan sebagai acuan terhadap nilai dari hasil keluaran yang diperoleh selama program berjalan.

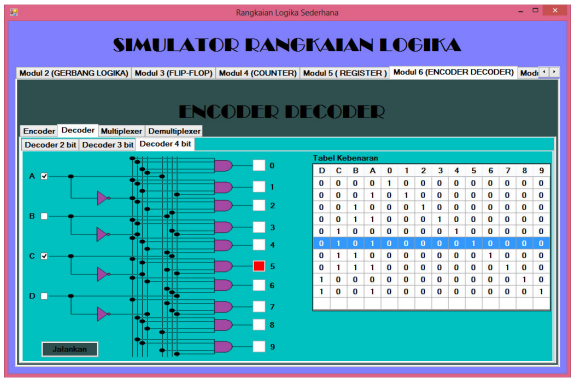
Encoder. Encoder merupakan penyandian bilangan tertentu menjadi sederetan angka biner. Dengan kata lain merubah angka atau bilangan menjadi kode biner. Pada sub menu ini diberikan 2 macam rangkaian encoder yaitu BCO dan BCD. Salah satu bentuk Encoder yang tersedia yaitu encoder BCO (Binary Code Octal)

BCO atau *Binary Code Octal* berarti Oktal ke Biner. Maksudnya adalah merubah bilangan oktal menjadi kode biner. Gambar 4.31 menunjukkan bentuk dari rangkaian BCO yang tersusun dari 3 buah gerbang OR dengan jumlah masukan sebanyak 7 (1 sampai 7). Dengan menekan salah satu bilangan oktal yang menjadi masukan, kemudian menekan tombol “Jalankan” maka tampil hasil keluaran kode biner pada rangkaian berupa led dengan nilai biner yang terdapat didalamnya. Berikut salah satu tampilan pada saat menjalankan program:



Gambar 23 Encoder BCO dengan masukan = 6

Decoder. Pada menu Decoder ini terdapat 3 macam penguraian angka biner. Berdasarkan jumlah bit dari deretan angka biner yang ada, diantaranya yaitu; decoder 2 bit, decoder 3 bit, dan decoder 4 bit. Masing-masing memiliki memliki kemampuan penguraikan sandi biner yang berbeda beda. Semakin besar jumlah bit maka semakin banyak nilai bilangan yang dapat diuraikan. Berikut tampilan dari salah satu menu Decoder:

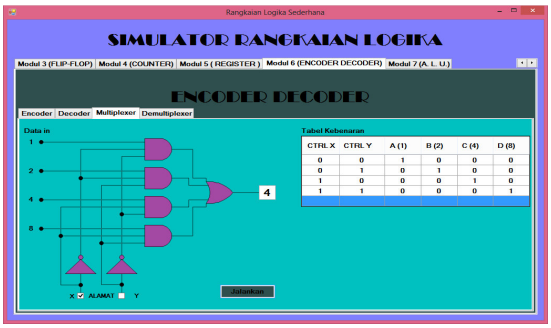


Gambar 24 tampilan decoder 4 bit dengan nilai keluaran aktif 5

Untuk bisa menjalankan program pada tampilan dengan cara pertama yaitu

menentukan nilai masukan pada rangkaian yang digambarkan dalam bentuk kotak ceklist. Artinya pada saat cek aktif maka masukan bernilai 1, jika tidak maka bernilai 0. Kemudian diikuti dengan menekan tombol "Jalankan", maka akan tampil perubahan warna pada led keluaran. Keluaran bernilai 1 ditandai dengan perubahan warna tampilan led menjadi merah, dengan nilai yang berbeda pada masing-masing keluaran (0 sampai 9). Tabel kebenaran menunjukkan 10 macam nilai keluaran yang dapat dihasilkan dari rangkaian decoder dengan nilai masukan yang telah ditentukan.

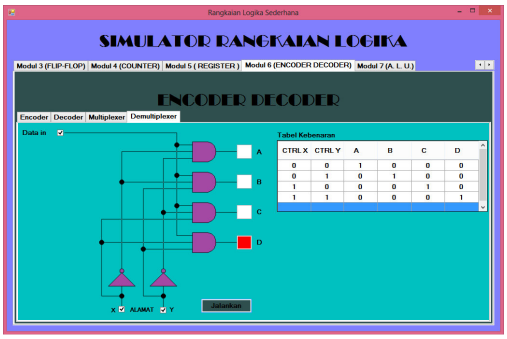
Multiplexer. Pada sub modul ini diberikan gambaran dari sebuah rangkaian multiplexer sederhana dengan masukan 4 buah data aktif dengan asumsi nilai masing-masing yaitu 1, 2, 4, dan 8. Terdapat pula set ALAMAT sebagai *data selector* 2 bit yaitu alamat *x* dan alamat *y* sebagai penentu data manakah yang akan dilewatkan sebagai hasil keluaran. Dengan satu jalur hasil keluaran yang digambarkan sebagai kotak nilai yang akan menampilkan data yang diloloskan oleh kendali alamat.



Gambar 25 Tampilan multiplexer dengan alamat x = 1 dan y = 0 yang melewatkan masukan dengan nilai 4.

Pada tampilan diatas terdapat 2 bit kendali alamat yang bisa menghasilkan kombinasi alamat sebanyak 4 jalur. Sehingga setiap kombinasi alamat akan menentukan nilai masukan mana yang akan dilewatkan.

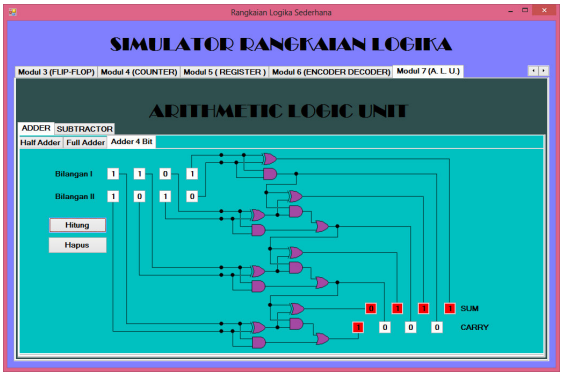
Demultiplexer. Rangkaian demultiplexer pada program ini menggambarkan sebuah data masukan yang sebelumnya merupakan hasil keluaran dari proses *multiplexing* yang kemudian akan di atur kembali untuk dilewatkan ke beberapa alamat atau jalur data yang berbeda-beda. Demultiplexer pada program ini menggunakan 2 bit alamat sehingga dapat menghasilkan 4 jalur data yang berbeda. Digambarkan pada rangkaian demultiplexer ini tidak terfokus pada nilai spesifik dari data masukan, akan tetapi lebih terfokus pada penandaan jalur mana yang akan dilewati oleh data tersebut. Untuk lebih jelasnya dapat dilihat pada tampilan berikut:



Gambar 26 Tampilan program demultiplexer dengan alamat x=1 dan y=1

Pada tampilan diatas, data in merupakan gambaran dari data masuk yang telah melewati proses *multiplexing* yang kemudian diteruskan ke rangkaian Demultiplexer. Dengan mengubah data in pada kondisi aktif kemudian menekan tombol "Jalankan", maka data tersebut akan diteruskan ke beberapa jalur yang ada sesuai dengan alamat yang diinginkan.

Modul 7 (A. L. U.) atau *Arithmetic Logic Unit*. Pada modul terkahir ini digambarkan bagaimana bentuk dari beberapa kombinasi rangkaian logika yang digunakan untuk melakukan perhitungan-perhitungan sederhana dalam sistem digital. Diantaranya yaitu rangkaian half adder, full adder, half subtractor, dan full subtractor. Untuk lebih jelasnya dapat dilihat pada salah satu tampilan rangkaian ALU berikut:



Gambar 27 Contoh hasil keluaran dari penjumlahan dua bilangan pada rangkaian

Adder 4 bit menggambarkan rangkaian yang digunakan untuk menjumlahkan dua

bilangan biner dengan ukuran maksimal 4 bit. Dimana rangkaiannya tersusun dari satu buah rangkaian half adder dan tiga rangkaian full adder yang disusun beruntun. Dimana *carry out* dari rangkaian pertama menjadi *carry in* rangkaian selanjutnya.

Pada rangkaian diatas digambarkan dua baris bilangan biner sepanjang 4 digit atau bit dimana Bilangan I akan dijumlahkan dengan Bilangan II. Dengan menekan tombol "Hitung" maka program akan melakukan perhitungan dengan menjumlahkan dua bilangan yang dimasukkan kemudian memberikan hasil keluaran pada tampilan output yang telah ada.

KESIMPULAN

Berdasarkan hasil percobaan yang dilakukan, maka dapat ditarik kesimpulan sebagai berikut:

1. Aplikasi Simulator Rangkaian Logika yang penulis buat dapat memberikan hasil keluaran dengan nilai yang tepat berdasarkan perhitungan dari teori-teori Rangkaian Logika yang ada.
2. Dengan adanya aplikasi Simulator Rangkaian Logika para pengguna dapat lebih memahami bagaimana melakukan perancangan maupun perhitungan data pada saat melakukan praktikum Rangkaian Logika. Sehingga dapat dihindari terjadinya kesalahan atau error dalam menampilkan hasil keluaran pada rangkaian yang dibuat.
3. Program Visual C++ sebagai bahasa pemrograman C++ berbasis Windows sangat cocok digunakan sebagai media didalam merancang aplikasi dalam bentuk simulasi.

SARAN

Aplikasi yang penulis buat lebih terfokus tentang bagaimana suatu rangkaian dapat memberikan hasil keluaran yang tepat berdasarkan jenis dari masukan yang diberikan. Sehingga diaharapkan untuk kedepannya bagi pengguna atau para pengembang aplikasi dapat menambahkan bentuk proses bagaimana rangkaian tersebut mengolah, mengontrol ataupun menyimpan nilai masukan yang ada sebelum memperoleh hasil keluaran.

Diharapkan bagi para pembaca yang ingin mengembangkan alikasi ini, untuk dapat lebih memperkaya event dan control agar interface dari aplikasi ini dapat terlihat lebih menarik dan interaktif. Seperti *class event mouse drag*, *control line*, *design picture*, dan lain-lain.

DAFTAR PUSTAKA

Anonim : http://new-funday.blogspot.com/2012/12/penyerderhanaan-fungsi-logika-dengan-k.html, diakses tanggal 27 mei 2015.

Anonim : http://www.ziddu.com/download/14648847/SevenSegmentindo.docx.html, diakses tanggal 27 mei 2015.

Ibrahim K.F., 1996 "*Teknik digital*". Diterjemahkan oleh Ir. P. Insap Santosa, M.Sc, Penerbit Andi, Yogyakarta.

Mismail Budiono, 1997, "*Dasar-dasar rangkaian logika digital*", Penerbit ITB.

Paul A. M., 1987, "*Elektronika komputer digital pengantar mikrokontroller*". Diterjemahkan oleh Tjia May On, Ph.D., Pencetak PT. Gelora Aksara Pratama, Penerbit Erlangga.

Peterson Dr., "*The Math Forum*", http://mathforum.org/library/drmath/view/60405.html, diakses tanggal 13 mei 2015.

Sianipar, R. H., 2015 "*Soal & penyelesaian Visual C++*". Pencetak dan Penerbit CV. ANDI OFFSET, Yogyakarta.

The New Oxford Dictionary, 1980 "*Definition of Simulation*".

Widjanarka W. N., Ir., 2006, "*Teknik digital*", Pencetak PT. Gelora Aksara Pratama, Penerbit Erlangga, Jakarta.

Wikipedia bahasa Indonesia., 2015 "*Microsoft Visual Studio*" Alamat situs : http://id.wikipedia.org/wiki/Microsoft_Visual_Studio, diakses tanggal 1 mei 2015.