UNIVERSITAS MUHAMMADIYAH PROF. DR. HAMKA
FAKULTAS TEKNIK
Jl. Tanah Merdeka No. 6, Kp. Rambutan, Ps. Rebo, Jakarta Timur. Telp. (021) 8400941; Fax. (021) 87782739
Website : www.ft.uhamka.ac.id; Email : ft@uhamka.ac.id

SURAT TUGAS

Nomor : 881/F.03.08/2022

Assalamu'alaikum warahmatullahi wabarakatuh,

Pimpinan Fakultas Teknik Universitas Muhammadiyah Prof. DR. HAMKA, memberikan tugas kepada:

Nama	:	Estu Sinduningrum, ST., MT
Tugas	:	Membuat Buku Pengembangan Keilmuan dengan Judul ***"Pengantar Teknik Digital Dengan Menggunakan Proteus Profesional"***
Waktu	:	Januari s.d. maret 2022
Tempat	:	Fakultas Teknik UHAMKA
Lain-lain	:	Setelah melaksanakan tugas agar memberikan laporan secara tertulis kepada Pimpinan Fakultas Teknik UHAMKA

Demikian surat tugas ini dibuat, agar dilaksanakan dengan sebaik-baiknya sebagai amanah dan ibadah kepada Allah SWT.

Wabillahittaufiq walhidayah,
Wassalamu'alaikum warahmatullahi wabarakatuh.

Jakarta, 5 Januari 2022

Dekan,

Dr. Dan Mugisidi ST., MSi

Tembusan :
1. Wakil Dekan I, II
2. Kaprodi TI, TM & TE

Irawati,
Estu Sinduningrum

PENGANTAR TEKNIK DIGITAL MENGGUNAKAN PROTEUS PROFESIONAL

penapersada

PENGANTAR TEKNIK DIGITAL
MENGGUNAKAN PROTEUS PROFESIONAL

Irawati
Estu Sinduningrum

PENERBIT CV. PENA PERSADA

PENGANTAR TEKNIK DIGITAL
MENGGUNAKAN PROTEUS PROFESIONAL

Penulis:
Irawati
Estu Sinduningrum

ISBN : 978-623-315-999-9

Design Cover :
Retnani Nur Briliant

Layout :
Eka Safitry

Penerbit CV. Pena Persada
Redaksi :
Jl. Gerilya No. 292 Purwokerto Selatan, Kab. Banyumas
Jawa Tengah
Email : penerbit.penapersada@gmail.com
Website : penapersada.com Phone : (0281) 7771388
Anggota IKAPI

Cetakan pertama : 2022

KATA PENGANTAR

Puji syukur Kami panjatkan kepada Allah Subhanallaahu wa Ta'ala yang telah memberikan hidayah dan karunia-Nya kepada penulis, sehingga dapat menyelesaikan buku pembelajaran yang berjudul "**PENGANTAR TEKNIK DIGITAL DENGAN PROTEUS PROFESIONAL**". Sholawat dan salam tercurah kepada Nabi Muhammad Shalallahu 'Alaihi wa Sallam beserta keluarga dan para sahabat.

Buku ini dibuat dengan tujuan untuk mempermudah proses pembelajaran Teknik digital sehingga diharapkan para pembaca khususnya mahasiswa Teknik Elektro dan Teknik Informatika dapat terbantukan dalam melakukan belajar secara mandiri pada buku ini terdapat teori sistem digital dan praktik rangkaian digital menggunakan proteus profesional. Proteus merupakan software aplikasi yang multifungsi, selain untuk aplikasi simulasi bisa juga untuk layout PCB secara otomatis dari rangkaian skematik.

Kami mengucapkan terima kasih kepada pihak-pihak dimana telah membantu dalam penulisan buku ini yang tidak bisa Kami sebut satu persatu. Kami juga berterima kasih kepada semua pihak yang telah membagi ilmu pengetahuan sehingga buku ini dapat diselesaikan.

Kami menyadari, buku ini masih jauh dari dari kata sempurna. Oleh karena itu kritik dan saran yang membangun akan saya nantikan demi kesempurnaan dari Buku ini.

Jakarta, Mei 2022

Penyusun

TIM PENULIS

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

BAB I
PENGANTAR SISTEM DIGITAL

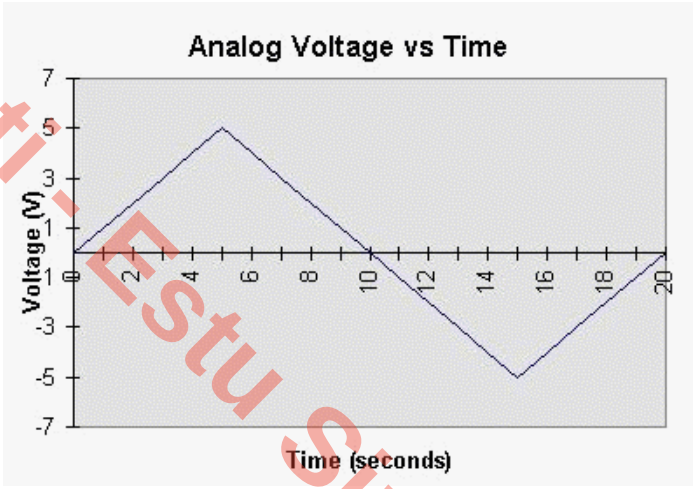
A. Pengertian Sistem Analog dan Digital

1. Sinyal Analog

Untuk representasi analog adalah suatu kuantitas direpresentasikan dengan kuantitas lain yang nilainya berbanding lurus dengan kuantitas pertama tersebut. Untuk contoh dari representasi analog adalah speedometer untuk kendaraan terutama mobil, dimana simpangan jarum sebanding dengan kecepatan pada mobil. Posisi sudut dari jarum menunjukkan besarnya kecepatan mobil, dan pada jarum tersebut mengikuti setiap perubahan yang terjadi pada saat kecepatan mobil untuk naik ataupun turun.

Dan contoh lain adalah pada thermostat ruang, dimana melengkungnya batang bimetal sebanding dengan temperatur ruang. Saat temperatur berubah secara bertingkat, lengkungan batang berubah sebanding dengan perubahan temperatur. Dan untuk kuantitas-kuantitas analog seperti yang diutarakan di atas mempunyai suatu karakteristik penting: pada kuantitas bisa berubah secara bertingkat pada suatu rentang harga kontinyu.

Pada gambar 1.1 terlihat bahwa besarnya tegangan analog berubah secara kontinyu untuk setiap perubahan waktu.

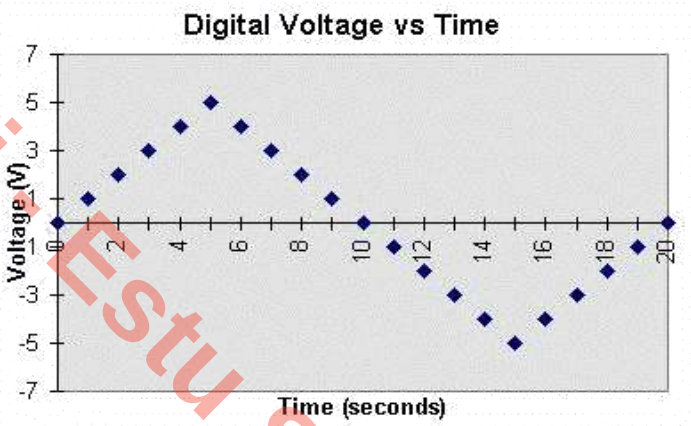


Gambar 1. 1 Diagram dari tegangan analog versus waktu

2. Sinyal Digital

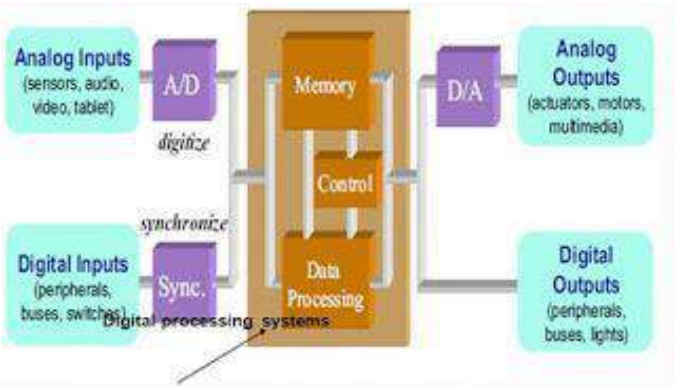
Untuk representasi digital, kuantitas-kuantitas tidak dinyatakan dengan kuantitas- kuantitas sebanding tetapi untuk symbol-simbol yang disebut digit. Adapun contoh, perhatikan pada jam digital, yang menunjukkan waktu dalam bentuk digit-digit desimal yang menyatakan pada jam, menit, dan detik. Seperti diketahui, waktu berubah secara kontinyu, tetapi yang terbaca dalam jam digital tidak berubah secara kontinyu, ia berubah satu step demi satu step per detik. Dengan kata lain, dan representasi digital dari waktu berubah dalam step-step diskrit. Dibandingkan pada representasi analog dari waktu yang ditunjukkan oleh jarum jam, dimana pembacaan skala berubah secara kontinyu.

Gambar 1.2 berikut ini menunjukkan diagram dari tegangan digital versus waktu Pada gambar terlihat bahwa besarnya tegangan digital berubah secara step demi step untuk setiap perubahan waktu.



Gambar 1. 2 Diagram dari tegangan digital versus waktu

Untuk sistem digital adalah sistem elektronika yang setiap rangkaian penyusunnya melakukan pengolahan pada sinyal diskrit. Sistem digital terdiri dari beberapa rangkaian digital/logika, komponen elektronika, dan elemen gerbang logika untuk suatu tujuan pengalihan tenaga ataupun energi. Rangkaian elektronika adalah suatu kesatuan dari komponen-komponen elektronika baik pasif maupun aktif yang membentuk suatu fungsi pengolahan sinyal (signal processing)



Gambar 1. 3 Representasi numeric digital dan analog

B. Perbedaan rangkaian Digital dan Sistem Digital

1. Rangkaian Digital

Rangkaian Digital ataupun rangkaian logika adalah kesatuan dari komponen-komponen elektronika pasif dan aktif yang mebentuk sebuah fungsi pemrosesan sinyal digital, komponen pasif dan aktif itu membentuk elemen logika. Adapun bentuk elemen logika terkecil adalah gerbang logika (logic gate). Gerbang logika adalah kesatuan dari komponen elektronika pasif dan aktif yang dapat melakukan operasi AND, OR, NOT.

Rangkaian Digital terdiri atas:

- Bagian-bagiannya terdiri atas beberapa gerbang logika
- Outputnya merupakan fungsi pemrosesan sinyal digital
- Input dan outputnya berupa sinyal digital

2. Sistem Digital

Untuk sistem Digital terdiri atas bagian-bagiannya terdiri atas beberapa rangkaian digital, gerbang logika dan komponen lainnya serta outputnya merupakan fungsi pengalihan tenaga sedangkan input dan outputnya merupakan suatu tenaga / energy.

a. Kelebihan Sistem Digital :
   - Sistem digital secara umum lebih mudah dirancang
   - Penyimpanan informasi lebih mudah
   - Ketelitian lebih besar
   - Operasi dapat diprogram
   - Untai digital lebih kebal terhadap noise
   - Lebih banyak sistem digital dapat dikemas dalam keping IC (Integrated Circuit)

b. Keuntungan Sistem Digital :
   - Kemampuan mereproduksi sinyal yang lebih baik dan akurat

- Mempunyai reliabilitas yang lebih baik dengan noise lebih rendah akibat immunitas yang lebih baik
- Mudah untuk didesain, tidak memerlukan kemampuan matematika khusus untuk memvisualisasikan sifat-sifat rangkaian digital sederhana
- Fleksibelitas dan fungsionalitas yang jauh lebih baik
- Kemampuan pemrograman yang lebih mudah
- Lebih cepat (debug IC complete complex digital dapat memproduksi sebuah keluaran lebih kecil dari 2 nano detik)
- Ekonomis jika dilihat dari segi biaya IC yang akan menjadi rendah akibat pengulangan dan produksi massal dari integrasi jutaan elemen logika digital pada sebuah chip miniatur tunggal

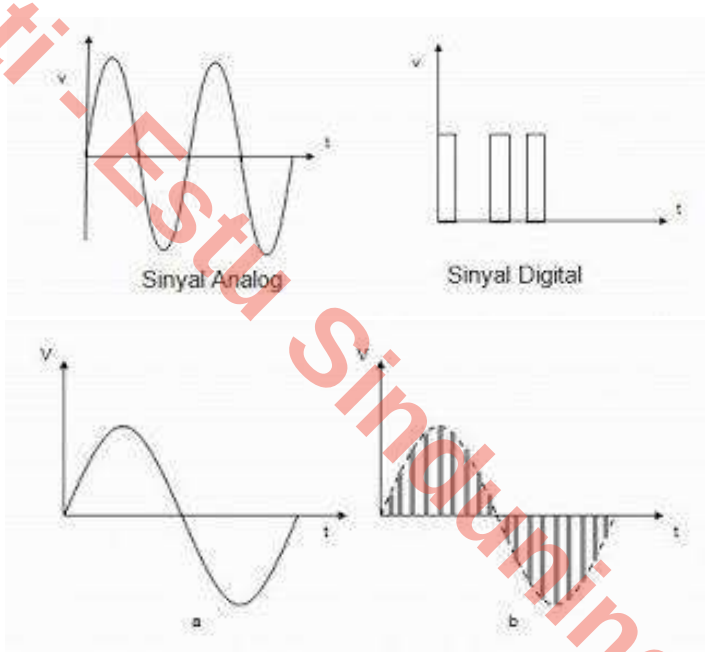
c. Bentuk Gelombang Sinyal Digital :
   - Untuk Sistem digital hanya mengenal dua buah kuantitas untuk mewakili dua kondisi yang ada. Kuantitas tersebut disebut dengan logika
   - Logika 1 mewakili kondisi hidup sedangkan logika 0 untuk kondisi mati. Sehingga bentuk untuk gelombang pada sistem digital hanya mengenal 2 arah, yaitu logika 1 dan logika 0 atau kondisi hidup dan kondisi mati.

Gambar 1. 4 Gelombang sinyal digital

C. Perbedaan Sinyal Analog dan Sinyal Digital

- Sinyal Digital adalah Sinyal yang berubah secara diskrit ataupun step by step serta berbentuk gelombang kotak

- Sinyal Analog adalah Sinyal yang berubah secara kontinyu ataupun berbentuk gelombang Sinus



Gambar 1. 5 Perbedaan sinyal analog dan digital

SOAL – SOAL LATIHAN

1. Apa perbedan antara besaran analog dan besaran digital?
2. Berikut ini yang manakah menyatakan kuantitas analog dan manakah yang digital?
   a. Tekanan tabung
   b. Perubahan temperatur dalam perioda 24 jam
   c. Switch sepuluh-posisi
   d. Skala penalaan radio

BAB II
INSTALASI DAN PENGENALAN KOMPONEN PROTEUS

A. Pengertian Proteus

Proteus professional merupakan salah satu kelompok software elektronika yang digunakan untuk membantu para desainer dalam merancang dan mensimulasikan suatu rangkaian elektronika. Software ini memiliki dua fungsi sekaligus dalam satu paket, yaitu paket satu sebagai software untuk menggambar rangkaian skematik dan dapat disimulasikan atau ujicoba yang diberi nama **ISIS**. Sedangkan Paket kedua digunakan untuk merancang gambar Printed Circuits Board (PCB) dari rangkaian skematik yang sudah dibuat yang diberi nama **ARES**. Secara langsung, pengubahan dari skematik ke PCB dapat dilakukan dalam software Proteus Profesional ini. Proteus Profesional ISIS memiliki versi yang selalu diperbarui, mulai dari versi 7.0 sampai dengan 8.0. Setiap kenaikan versi memiliki penambahan akan library komponen yang dapat diambil dan digunakan dalam penggambaran atau perancangan sistem. Sebagai perancang rangkaian skematik terlebih dahulu menggunakan ISIS sebagai media yang memudahkan dalam peracangan skematik dan melakukan simulasi. Banyaknya library dari Proteus Profesional ISIS membuat software ini dikatakan software simulasi lengkap, yaitu terdiri dari komponen-komponen pasif, Analog, Trasistor, SCR, FET, jenis button/tombol, jenis saklar ataupun relay, IC digital, IC penguat, IC programmable (mikrokontroller) dan IC memory. Selain didukung dengan kelengkapan komponen, juga didukung dengan kelengkapan alat ukur seperti Voltmeter, Ampere meter, Oscilloscope, Signal Analyzers, serta pembangkit Frekuensi. Kelengkapan fitur yang disediakan ini

menjadikan Proteus Profesional ISIS menjadi salah satu software simulasi elektronik terbaik.

Software Proteus adalah sebuah software yang digunakan untuk mendesain PCB yang juga dilengkapi dengan simulasi pada level skematik sebelum rangkaian skematik di-upgrade ke PCB untuk memastikan PCB dapat berfungsi dengan semestinya. Proteus mengkombinasikan program ISIS untuk membuat skematik untuk mendesain rangkaian dengan program ARES untuk membuat layout PCB dari skematik yang dibuat. ISIS disini digunakan sebagai program untuk perancangan, sedangkan ARES atau disebut juga Advanced Routing and Editing Software digunakan untuk membuat modul layout PCB.

1. Fitur-fitur dari Proteus adalah sebagai berikut :
   - Memiliki kemampuan untuk mensimulasikan hasil rancangan baik digital maupun analog maupun gabungan keduanya.
   - Mendukung instrumen-instrumen virtual seperti voltmeter, ammeter, osciloscope, logic analyser, dan masih banyak lagi.
   - Memiliki model-model peripheral yang interactive seperti LED, tampilan LCD, RS232, dan berbagai jenis library lainnya.
   - Memiliki kemampuan menampilkan berbagi jenis analisis secara grafis seperti transient, frekuensi, noise, distorsi, AC dan DC, dan masih banyak lagi.
   - Mendukung simulasi berbagai jenis microcontroller.
   - Mendukung berbagai jenis komponen-komponen analog.
   - Mendukung open architecture sehingga pengguna dapat memasukkan program seperti C++/ Arduino untuk keperluan simulasi.
   - Mendukung pembuatan PCB yang di-update secara langsung dari program ISIS ke program pembuat PCB-ARES.

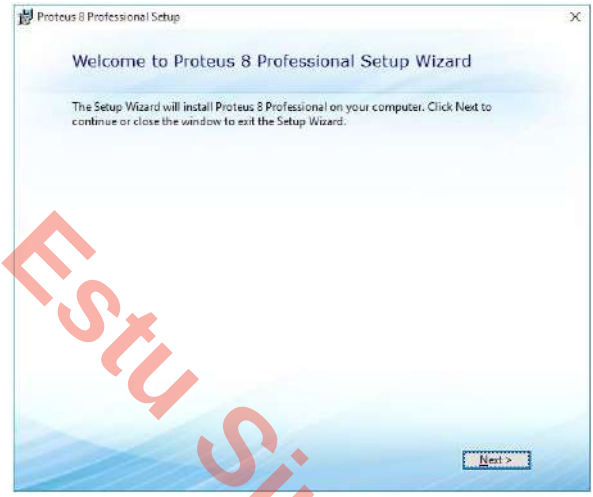
2. Fitur-Fitur dari ISIS adalah sebagai berikut :
   - Dapat dioperasikan pada Windows 98/XP/7/10 sampai dengan Windows terbaru.
   - Adanya fasilitas pemilihan komponen serta pemberian properties.
   - Memiliki fasilitas report terhadap kesalahan-kesalahan pada perancangan serta simulasi elektrik.
   - Routing secara otomatis dan memiliki fasilitas penempatan dan penghapusan pada dot.
   - Mendukung untuk system perancangan berbagai jenis bus dan komponen-komponen pin, port modul dan jalur.
   - Mendukung untuk fasilitas interkoneksi dengan program pembuat PCB-ARES.
   - Memiliki fasilitas untuk menambahkan package dari para komponen yang belum didukung.

3. Fitur-fitur dari ARES adalah sebagai berikut :
   - Terintegrasi dengan program pembuat skematik ISIS, memiliki kemampuan untuk menentukan informasi routing pada skematik.
   - Memiliki database dengan tingkat keakuratan 32-bit dan memberikan resolusi sampai 10 nm, resolusi angular 0,1 derajat serta ukuran maksimum board sampai kurang lebih 10 m. ARES mendukung sampai 16 layer.
   - Visualisasi pada board 3-Dimensi.
   - Penggambaran 2-Dimensi dengan simbol library.

B. Instalasi Proteus Profesional

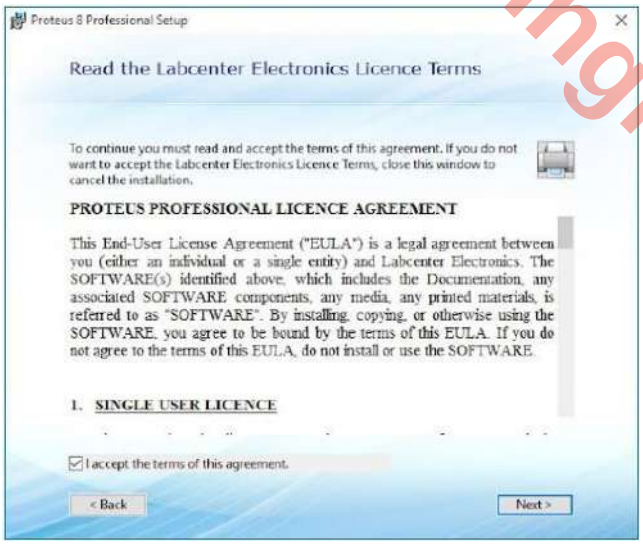
Untuk menjalankan program Proteus Profesional 8.0 perlu dilakukan installasi dulu pada computer. Adapun langkah installasi pada computer sebagai berikut :

1. Buka Folder Proteus 8.0, jalankan **setup proteus 8.0** dengan cara double click
2. Klik setup.exe kemudian klik Next.



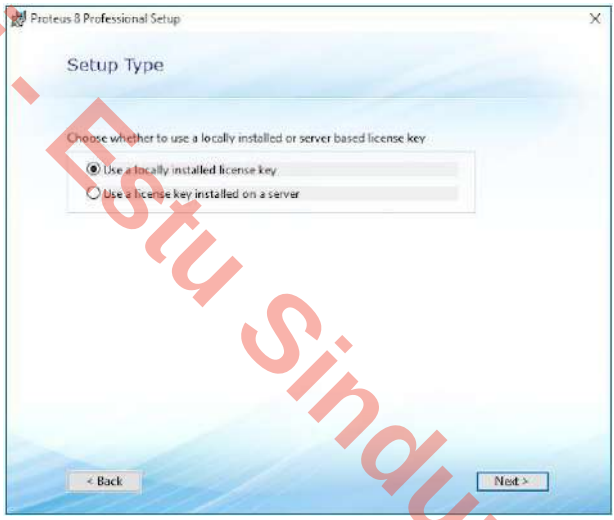
Gambar 2. 1 Instalashield wizard

3. Lalu ceklis kolom *I accept the terms of this aggrement* setelah itu lalu Next.



Gambar 2. 2 Licence Agreatment

4. Setelah klik Next tindakan tadi maka kembali anda klik *Use a locally installed license key*, lalu klik Next lagi

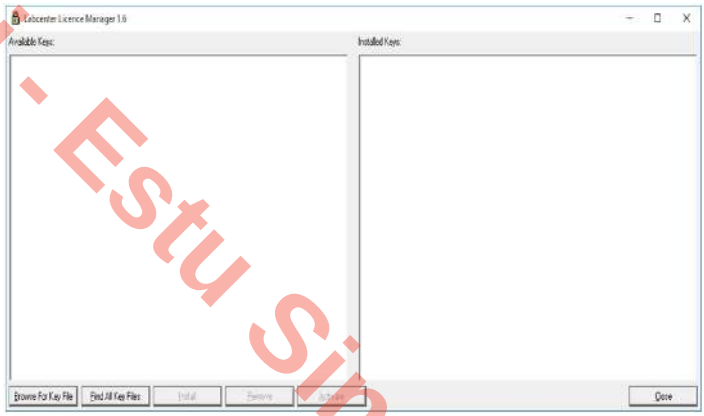


Gambar 2. 3 setup type

5. Apabila belum pernah terinstall proteus, akan muncul window baru sebagai berikut

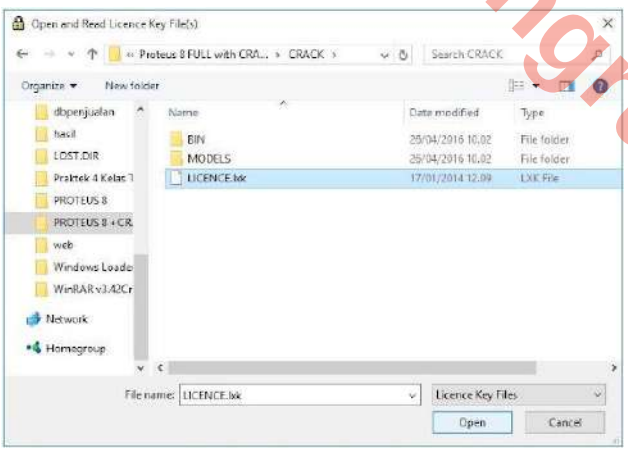
Gambar 2. 4 License key

6. Lalu klik *Browse For Fey File* (hal ini akan membuka window explore baru).



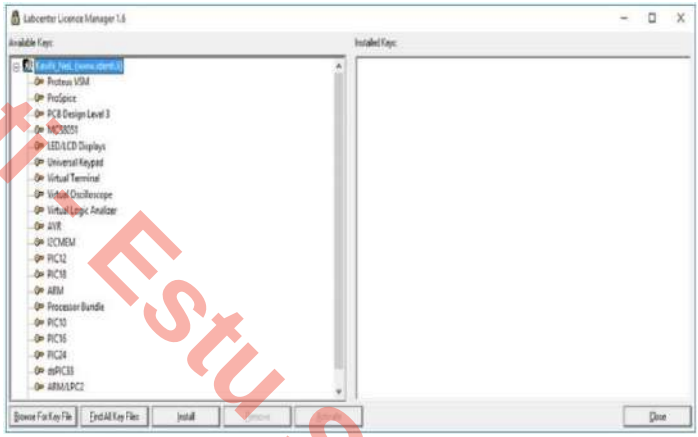
Gambar 2. 5 labcenter Licence Manager

7. Lalu buka folder Proteus 8 dan buka lagi folder Crack maka Pilih *Licence - Open*



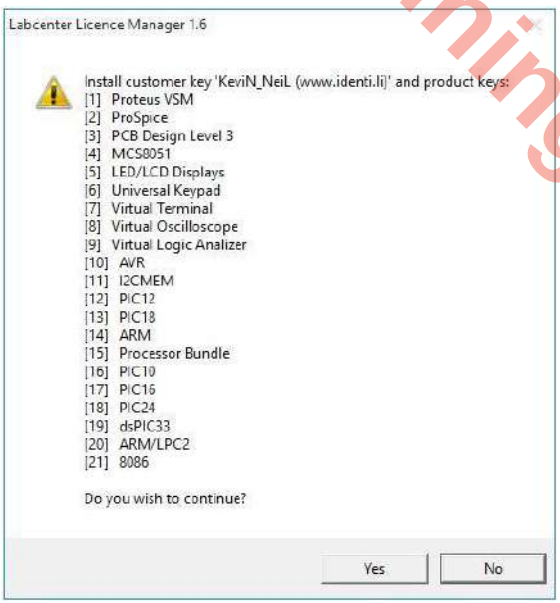
Gambar 2. 6 Licence

8. Jika anda sudah lakukan tahap diatas dengan benar anda akan ditampilkan beberapa text beserta gambar kunci kunci sebelah kiri, dan sekarang anda klik Install



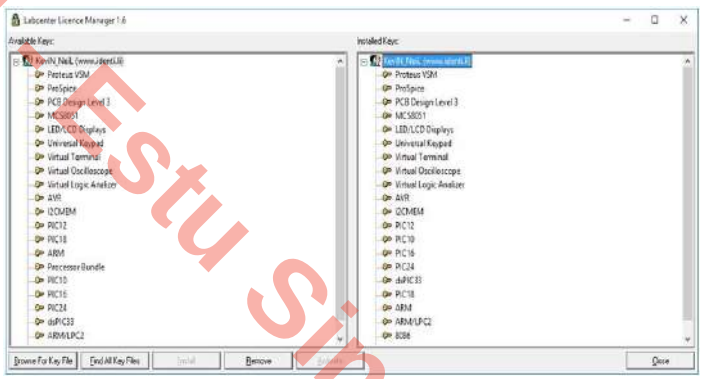
Gambar 2. 7 Licence Manager

9. Selanjutnya jika menemukan suatu bertanda **PERHATIAN**, maka klik Yes.



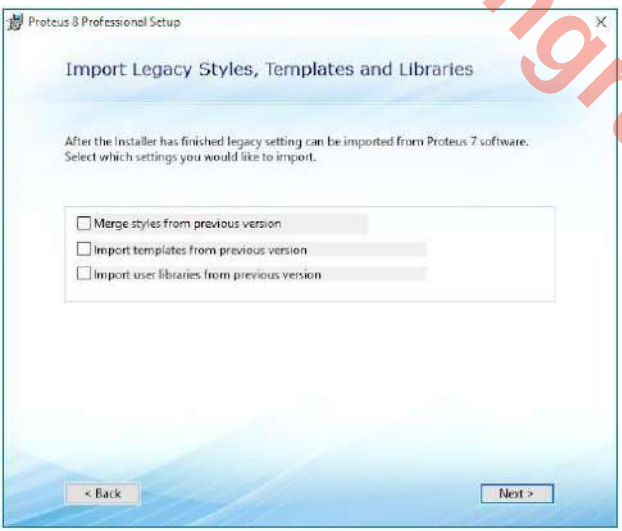
Gambar 2. 8 Tampilan license key

10. Setelah tindakan diatas maka anda lakukan sisi kanan yang awalnya kosong akan terisi persis seperti pada sisi kiri, dan kemudian anda klik Close



Gambar 2. 9 Tampilan License Key

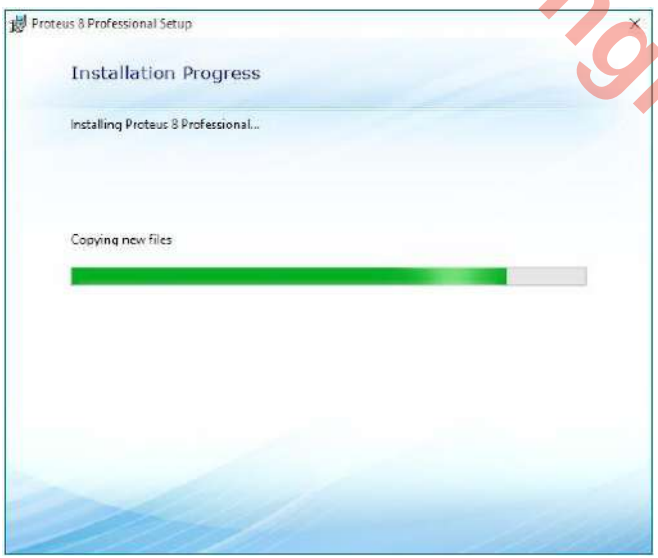
11. Untuk tahap dibawah ini jangan ceklis ketiga kolom yang ditampilkan, klik saja langsung Next



Gambar 2. 10 Legacy Style

12. Ok selanjutnya anda tinggal pilih Mode *Typical*

Gambar 2. 11 Proteus Setup

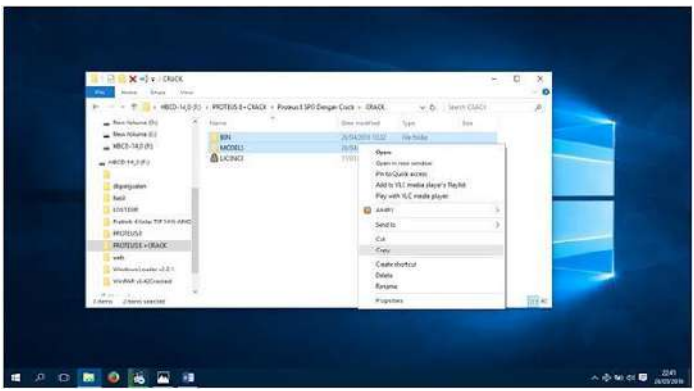


Gambar 2. 12 Proses Instal

13. Installation Progress, JANGAN di klik Run Proteus 8 Professional, tapi klik Close terlebih dahulu (hal ini dilakukan untuk penggunaan cracknya).

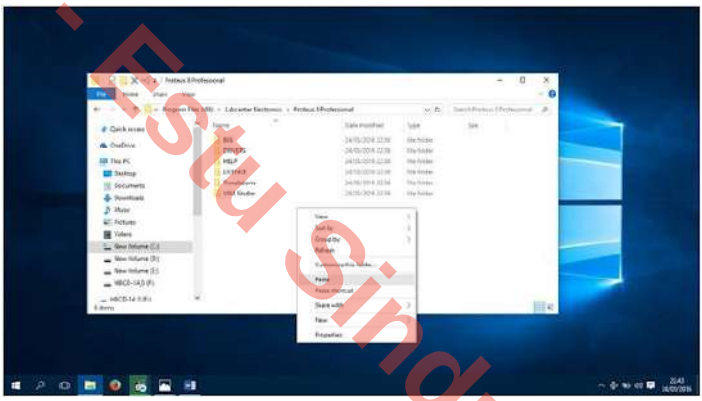
Gambar 2. 13 Proses Setup

14. Buka kembali folder Proteus 8 SP0 dan buka lagi folder CRACK anda dan copy folder BIN dan MODELS



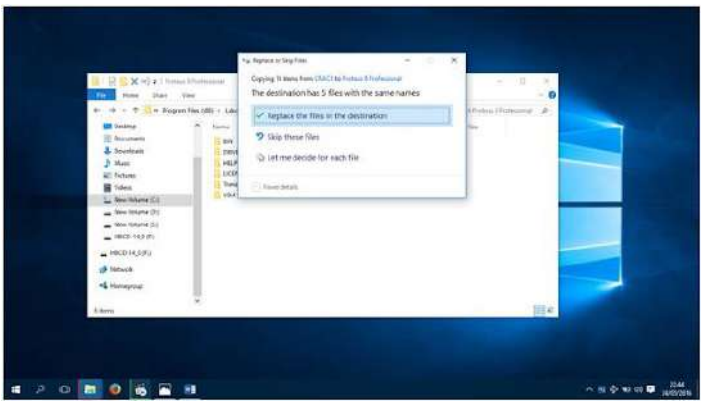
Gambar 2. 14 Crack

15. Lalu klik CTRL+V pada tempat anda melakukan install proteus, umumnya hal ini berada pada contoh folder berikut *C:\ProgramFiles\Labcenter Electronics\Proteus 8 Professional*.



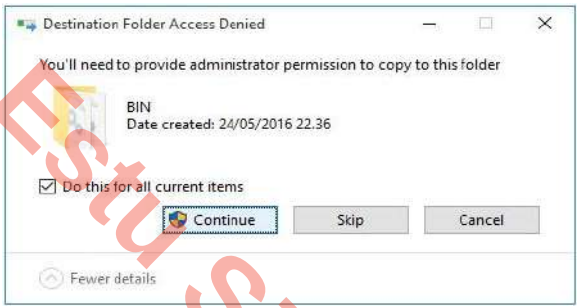
Gambar 2. 15 Crack

16. Pada folder terakhir anda pastikan BIN yang sudah kita copy sebelumnya, Jika anda mendapati Konfirmasi bersifat Folder anda atau Klik Yes. (jika anda menggunakan windows 7 tampilannya berbeda, anda saja Copy and Replace/continue, karena saat ini saya menggunakan win 10)



Gambar 2. 16 Penempatan BIN

17. Dan jika anda mendapati sebuah konfirmasi bersifat file maka anda Ceklis terlebih dahulu lalu di pilih Copy And Replace dan klik Continue

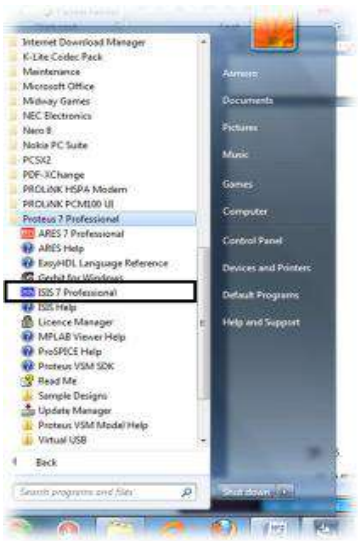


Gambar 2. 17 Proses BIN

C. Menjalankan Proteus Profesional

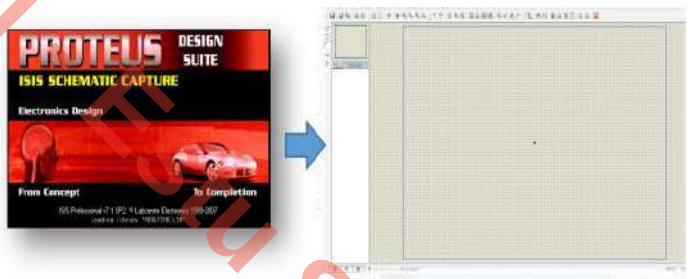
Tahap Tahap menjalankan Proteus Professional 8.0 sebagai berikut:

1. Klik Start - All Program - Proteus 8.0 Profesional - Klik ISIS 7 Profesional



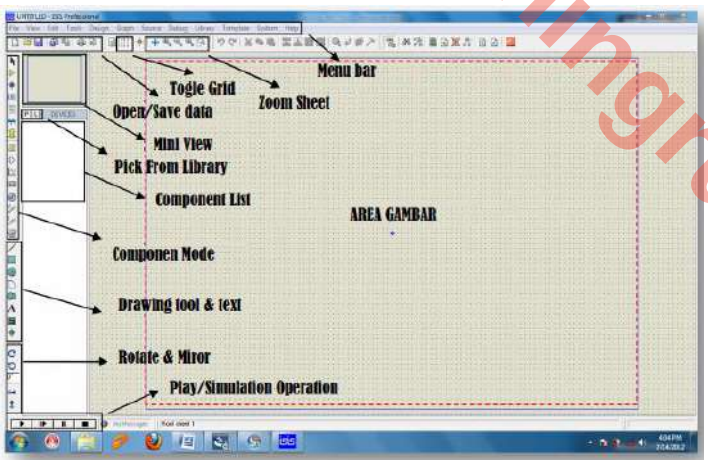
Gambar 2. 18 windows

2. Sambil menunggu loading file library seperti dibawah ini, sampai program Proteus ISIS 8 Profesional telah jalan/terbuka



Gambar 2. 19 tampilan proteus

3. Tampilan pada window Proteus Profesional ISIS 8 seperti tampak dibawah ini, dan memiliki fungsi difitur-fiturnya yang sering digunakan.



Gambar 2. 20 Tampilan new design

- Menu Bar : Merupakan list menu yang dapat digunakan dalam perancangan ataupun pengolahan pada suatu gambar rangkaian

- Open Save Data meliputi:
  -  New File: Tampilan untuk membuat file baru dengan area gambar baru.
  -  Open File: Tampilan untuk membuka file yang pernah disimpan
  -  Save : Tampilan untuk menyimpan file yang telah dibuat.
- Togle Grid : Menampilkan bantuan titik-titik panduan pada area gambar
- Zoom Sheet meliputi: (dapat menggunakan scroll mouse)
  -  Centre at Cursor: Menentukan area tengah tampilan gambar dengan bertumpu pada cursor
  -  Zoom in : Tampilan untuk memperbesar gambar
  -  Zoom out: Tampilan untuk memperkecil gambar
  -  Zoom to view sheet: Tampilan untuk menampilkan keseluruhan gambar dalam layar monitor
  -  Zoom to area: Tampilan untuk memperbesar gambar dengan memilih area yang dikehendaki.
- Mini view: Tampilan untuk menampilkan gambar dalam bentuk tampilan kecil seluruh area gambar.

- Component List: Daftar komponen yang telah diambil dari library.
  -  Pick From Library: Tampilan untuk mengambil komponen pada library yang akan diletakkan pada component list.

- Componen Mode meliputi:
  -  Selection mode: Tampilan untuk memilih dan melakukan aksi pada komponen yang dipilih
  -  Component Mode: Tampilan untuk mengambil komponen pada library
  -  Terminal Mode: Tampilan untuk mengambil dan menggunakan terminal yang dibutuhkan dalam rangkaian (VCC,Gnd,Input,Output)
  -  Generator Mode: Tampilan untuk memilih pembangkit pulsa yang akan digunakan
  -  Voltage Probe: Tampilan untuk terminal dengan tampilan nilai dari jalur koneksi komponen dengan menampilkan besaran tegangan
  -  Virtual Instrument Mode: Tampilan untuk mengambil alat ukur yang akan digunakan (CRO, Voltmeter, Ampere meter, AFG, Signal Analyzer)

- Drawing Tool and Text meliputi:
  -  2D Graphic line Mode: Tampilan untuk membuat garis jalur rangkaian 2D
  -  2D Graphic box Mode: Tampilan untuk membuat gambar kotak/persegi 2D pada area gambar

-  2D Graphic Circle Mode: Tampilan untuk membuat gambar lingkaran 2D pada area gambar
-  2D Graphic Arc Mode: Tampilan untuk membuat gambar Arc/garis lengkung 2D pada area gambar
-  2D Graphic Text Mode: Tampilan untuk menambahakan tulisan text 2D pada area gambar

➢ Rotate And Mirror meliputi:

-  Rotate Clockwise: Tampilan untuk merotasi obyek searah jarum jam
-  Rotate Anticlockwise: Tampilan untuk merotasi obyek berlawanan dengan arah jarum jam
-  X mirror: Tampilan untuk mencerminkan obyek kearah X
-  Y mirror: Tampilan untuk mencerminkan obyek kearah Y

➢ Play and Simulation Operation

-  Play: Menjalankan simulasi rangkaian yang telah dibuat
-  Step: Menjalankan simulasi secara tahap pertahap
-  Pause: Memberhentikan simulasi rangkaian yang telah dibuat
-  Stop: Menghentikan simulasi rangkaian yang telah dibuat

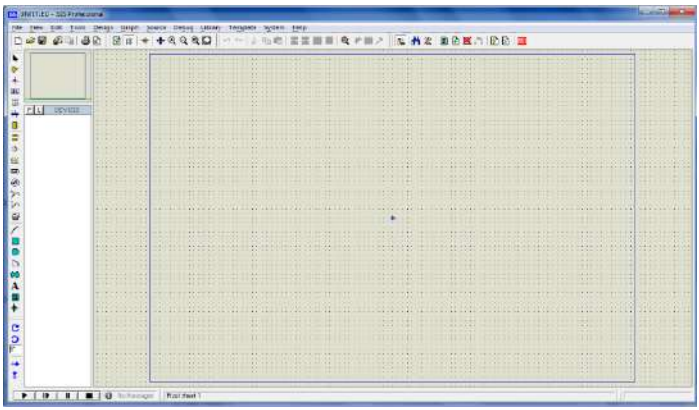
BAB III
CARA MENGGUNAKAN PROTEUS

A. Penggunaan Proteus

Proteus Profesional adalah salah satu jenis software yang dapat kita gunakan untuk membuat simulasi rangkaian elektronika seperti membuat rangkaian seri, pararel, simulasi mikrokontroler dan mikroprosesor. Proteus juga bisa memastikan bahwa sistem yang kita gunakan berhasil ataupun tidak berhasil. Fungsi lain dari sebuah proteus adalah membantu kita untuk membuat PCB dari embedded system yang akan anda buat dan software ini hanya bisa berjalan dioperating system windows.

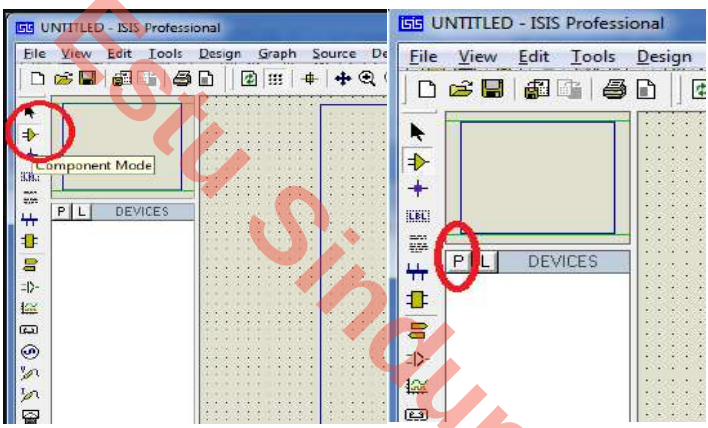
Berikut langkah-langkah menggunakan software proteus untuk membuat rangkaian skematik, sebagai berikut :

1. Proses instalasi software proteus 8.0 dapat dilakukan dengan cara mudah. Cari source (sumber/ file setup) dari proteus ini, lalu double click pada file setup. Tentukan tempat tujuan proteus yang akan diinstall (misalnya C:\Program Files\proteus), lalu klik **OK**. Tunggu saat proses instalasi selesai, lalu ke klik - start menu– proteus 8.0 profesional – isis profesional. Proteus siap dipakai



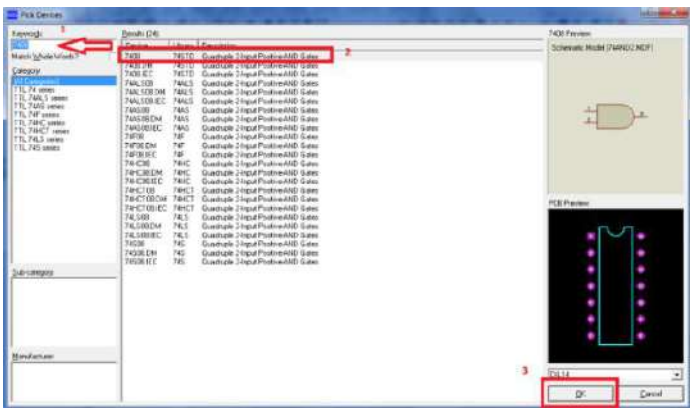
Gambar 3. 1 Tampilan utama

2. Kemudian pilih komponen yang akan digunakan, lihat gambar. Pada toolbox sebelah kiri, pilih Component mode kemudian klik tombol yang berisi huruf P Untuk mengaktifkan Pick Device. Pick Device adalah box dialog untuk memilih komponen yang akan kita gunakan.



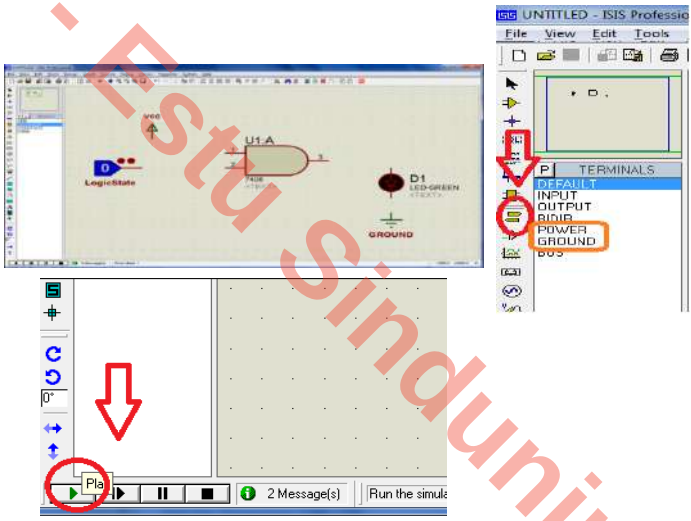
Gambar 3. 2 Box dialog

3. Maka akan muncul box dialog, isikan komponen yang Anda inginkan pada kolom keywords. Sebagai contoh diisi 7408 kemudian pilih salah satu list komponen yang muncul, klik OK



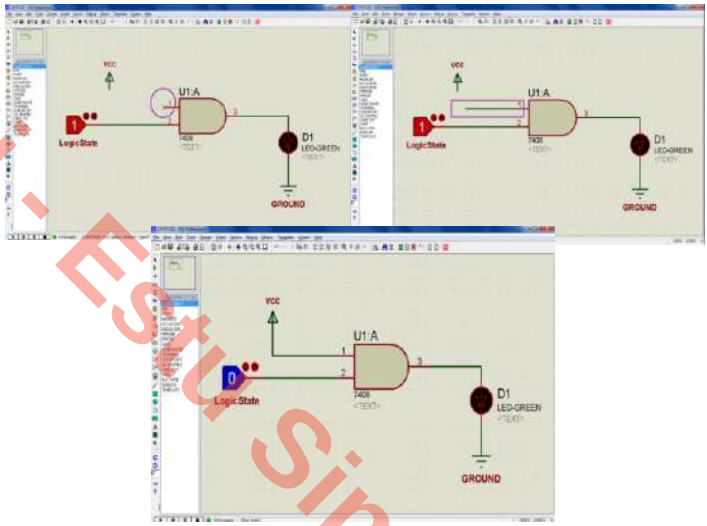
Gambar 3. 3 Pemilihan komponen

4. Letakkan komponen yang telah Anda pilih (dalam contoh ini adalah gerbang AND dari IC 7408). Selanjutnya silahkan mencari komponen Logicstate, Led Green, VCC dan GROUND kemudian letakkan ke stage.



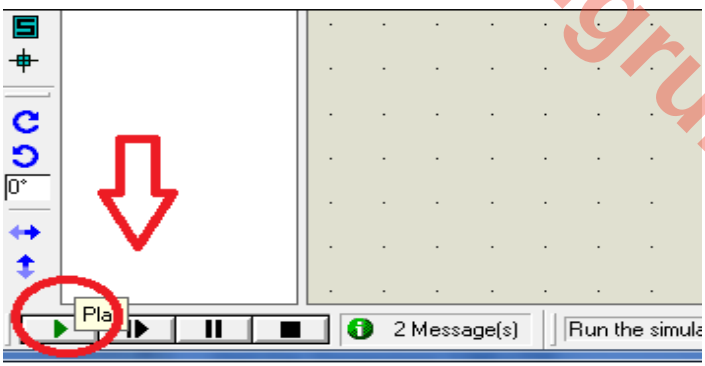
Gambar 3. 4 Peletakan komponen

5. Langkah berikutnya adalah menyambungkan komponen satu dengan komponen lainnya dengan cara, arahkan kursor mendekati ujung komponen lalu klik mouse kemudian arahkan kursor menuju ujung komponen yang lain, seperti gambar dibawah ini

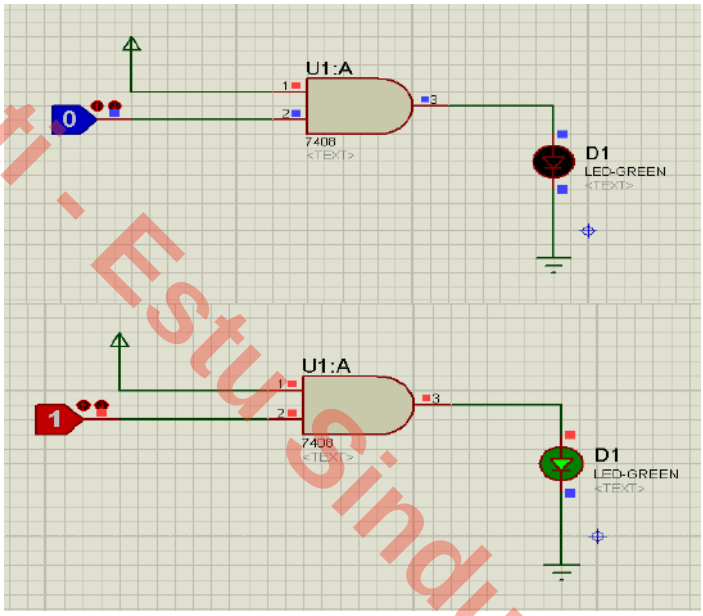


Gambar 3. 5 Penyambungan antar komponen

6. Setelah selesai menggambar komponen, saatnya kita menjalankan simulasi. Simulasi akan berjalan setelah kita menekan tombol PLAY di pojok kiri bawah



Gambar 3. 6 Run rangkaian



Gambar 3. 7 Hasil run rangkaian

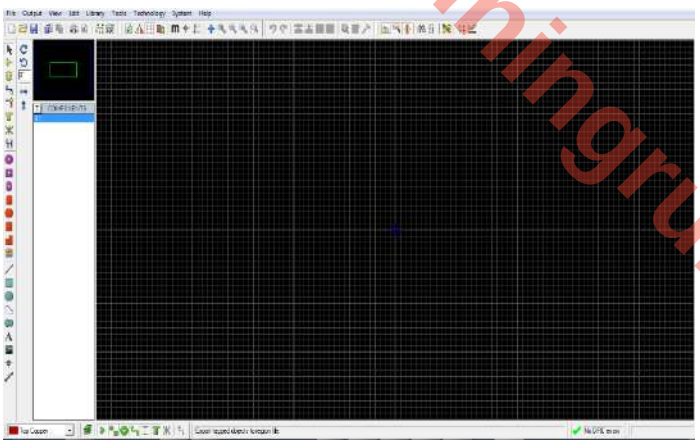
Dari uji coba diatas adalah ketika logicstate bernilai 1, maka LED akan menyala. Hal ini disebabkan karena ujung gerbang AND (kaki 1) di hubungkan dengan VCC sehingga logikanya pada kaki ini bernilai 1. Seperti yang telah kita ketahui bersama, 1 di AND kan dengan 1 maka hasilnya adalah 1. LED hidup karena mendapat suplay tegangan sebesar 5 v (logika 1 = 5 v) kemudian dialirkan ke GROUND.

B. Cara Membuat Lay Out Pcb 1 (Satu) Layer Dengan Proteus

Pembuatan layout PCB dapat dilakukan dengan dua cara yaitu konvensional dan modern. Untuk dapat melayout PCB secara konvensional adalah bisa kita langsung menggambar pola layout pada PCB (biasanya menggunakan Permanent Marker atau bisa menggunakan PCB Desagner). Cara ini mungkin sudah banyak ditinggalkan, karena dalam

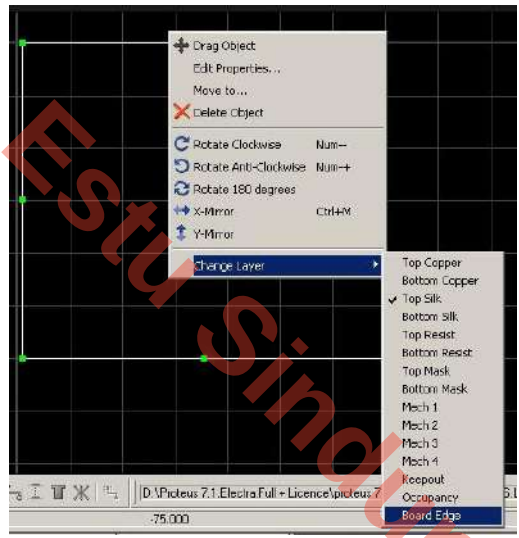
pembuatannya layouter harus benar-benar mahir agar layout yang dihasilkan sesuai dengan skema rangkaian dan sesuai dengan ukuran PCB yang dipergunakan. Pembuatan layout PCB yang kedua adalah cara modern dengan memanfaatkan kemajuan teknologi sekarang. Pembuatan layout PCB dapat menggunakan software computer. Banyak sekali software pembuatan layout PCB yang bisa digunakan seperti PROTEL, EXPRESS PCB, DIPTRACE, EAGLE, PROTEUS, dll. Disini saya akan sedikit berbagi tentang cara pembuatan layout PCB 1 (satu) layer menggunakan PROTEUS. langkah-langkahnya sebagai berikut :

1. Pada gambar skema rangkaian yang akan dibuat layout PCBnya menggunakan ISIS PROTEUS seperti cara diatas..
2. Simpan dokumen yang telah dibuat (Ctrl + S) lalu klik Icon Netlist Transfer to ARES yang terletak pada pojok kanan atas



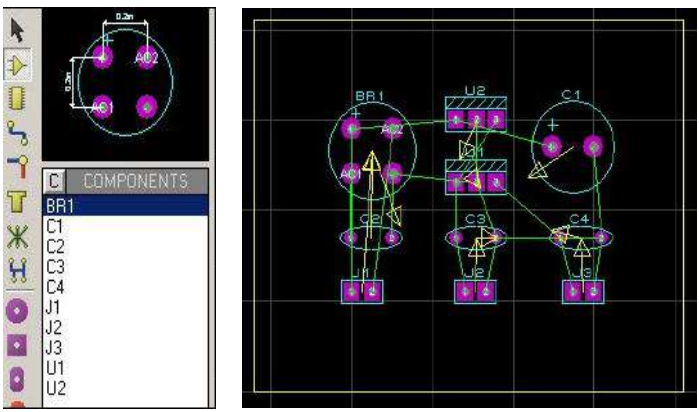
Gambar 3. 8 Tampilan awal

3. Buat kotak sesuai dengan ukuran PCB yang akan dibuat menggunakan 2D Graphics Box Mode



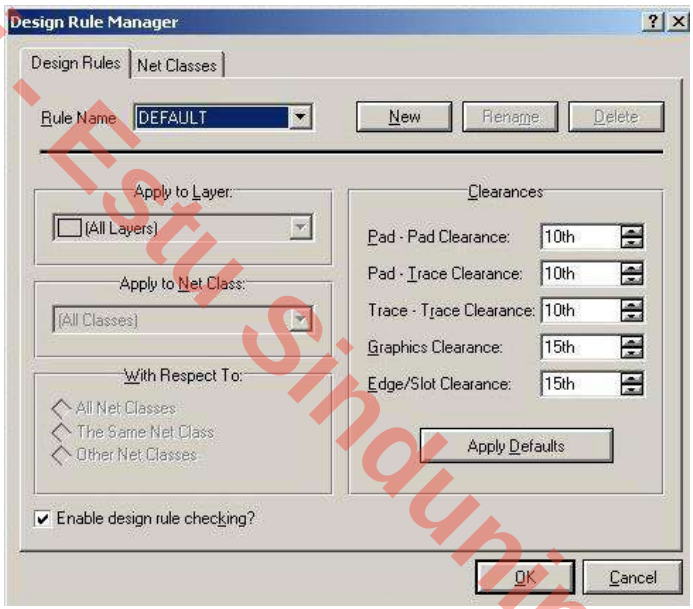
Gambar 3. 9 Kotak pada tampilan pcb

4. Ubah layer kotak tersebut menjadi board edge dengan cara klik kanan pada kotak pilih change layer pilih board edge, mulailah meletakkan komponen kedalam kotak

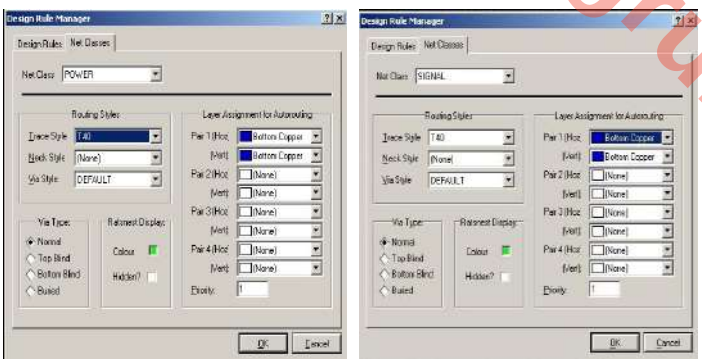


Gambar 3. 10 tampilan layout

5. Kemudian Klik Tools >> Design Rule Manager. Sehingga akan muncul tampilan seperti dibawah ini



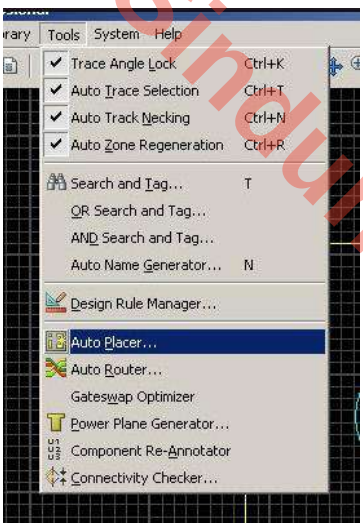
Gambar 3. 11 Rule manager



Gambar 3. 12 net class POWER dan net class SIGNAL

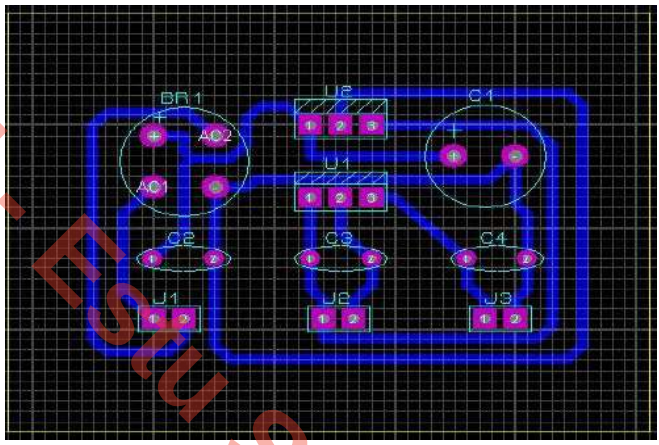
Inti dari pembuatan layout PCB 1 layer adalah pada kotak pilihan Layer Assignment for Autorouting. Jika pada pair1 dipilih top copper dan bottom copper maka pembuatan layout menjadi 2 layer (atas bawah/dua sisi). Jika dipilih top layer saja atau bottom layer saja maka layout akan menjadi satu sisi saja. Pengaturan tebal jalur dapat dipilih kotak pilihan Routing styles pada kolom Trace Style. Semakin besar nilai Trace yang digunakan maka nantinya jalur rangkaian akan semakin lebar.

6. Selanjutnya klik tools >> Auto Router.



Gambar 3. 13 Auto Router

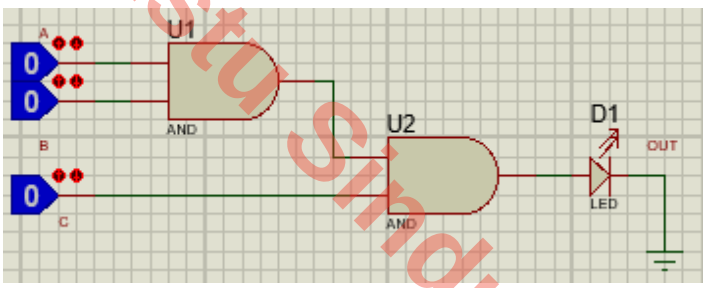
7. Lalu klik Begin routing dan keluar hasilnya.



Gambar 3. 14 Hasil Layout yang sudah jadi

SOAL – SOAL LATIHAN

1. Jelaskan mengenai langkah-langkah menjalankan program Proteus ARES dan ISIS.
2. Buatlah rangkaian sederhana seperti pada gambar dibawah ini, dimana Anda bisa menggunakan rangkaian digital 2 buat gerbang AND dengan 3 input, serta sebuah led sebagai indikator keluaran



BAB IV
SISTEM BILANGAN

A. Pengertian Sistem Bilangan

Sistem bilangan dapat diklasifikasikan menjadi dua, yaitu simbolis seperti sistem bilangan Romawi dan terbobot posisi (positional weighted) seperti sistem bilangan desimal yang menggunakan numeral Arabik. Untuk peralatan, dua nilai seperti komputer maka yang digunakan adalah sistem bilangan biner yang juga merupakan sistem bilangan terbobot posisi.

Ada 4 Sistem bilangan, yaitu :

1. Bilangan Desimal
2. Bilangan Biner
3. Bilangan Oktal
4. Bilangan Hexadesimal

B. Jenis-jenis Sistem Bilangan

Ada beberapa jenis sistem bilangan, diantaranya adalah:

1. Bilangan Desimal

Manusia dalam kehidupan sehari-hari selalu menggunakan sistem bilangan desimal untuk berbagai keperluan perhitungan. Maka, desimal telah menjadi sistem yang paling dikenal oleh manusia dibandingkan dengan sistem bilangan yang lain.

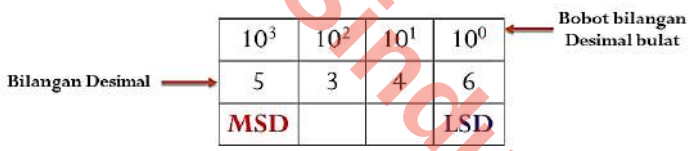
- Angka penyusun bilangan desimal disebut digit.
- Setiap digit memiliki BOBOT digit yang berbeda.
- Desimal merupakan sistem bilangan dengan basis 10, artinya digit/angka yang digunakan untuk menyajikannya berjumlah 10 buah yakni : 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

- Setiap penyusunnya memiliki bobot kepangkatan 10n - n merupakan
- bilangan bulat positif dan negative.
- BOBOT terbesar - Most Significant Digit (MSD) - paling KIRI.
- BOBOT terkecil - Least Significant Digit (LSD) - paling KANAN

Contoh 1 :

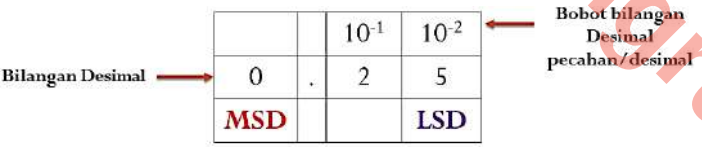
Bilangan $(5346)_{10}$ *atau* 5346_{10} memiliki arti :

$(5346)_{10} = 5 \times 10^3 + 3 \times 10^2 + 4 \times 10^1 + 6 \times 10^0$



Contoh 2 :

Bilangan Desimal $(0.25)_{10}$ *atau* 0.25_{10} memiliki arti :



Tabel 4. 1 Bobot Bilangan desimal untuk -3 ≤ n ≤ +3

	Bilangan Bulat				Bilangan Pecahan		
N	3	2	1	0	-1	-2	-3
Bobot	10^3	10^2	10^1	10^0	$_{10}-1$	$_{10}-2$	$_{10}-3$

2. Bilangan Biner

Adapun bilangan Biner merupakan sistem bilangan dengan basis 2, yang artinya dalam sistem ini digit yang digunakan berjumlah 2 buah yakni 0 dan 1. Setiap digit penyusunnya memiliki bobot kepangkatan 2^n dengan n

bilangan bulat positif dan negatif

Contoh 1 :

Bilangan $(10011)_2$ *atau* 10011_2 memiliki arti :

$(10011)_2 = 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0$

$= 16 + 0 + 0 + 2 + 1$

$= 19_{10}$

Contoh 2 :

Bilangan (0.011)2 atau 0.0112 memiliki arti:

$(0.011)_2 = 0 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3}$

$= 0 + 0.25 + 0.125$

$= 0.375_{10}$

Contoh 3 :

Bilangan $(100.111)_2$ atau 100.111_2 memiliki arti

$(100)_2 = 1 \times 2^2 + 0 \times 2^1 + 0 \times 2^0$

$= 4 + 0 + 0$

$= 4_{10}$

$(0.111)_2 = 1 \times 2^{-1} + 1 \times 2^{-2} + 1x2^{-3}$

$= 0.5 + 0.25 + 0.125$

$= 0.875$

$(100.111)_2 = 4 + 0.875$

$= 4.875_{10}$

Tabel 4. 2 Penyajian Bobot Bilangan Biner Untuk -4 ≤ n ≤ +8

	Bilangan Bulat									Bilangan Pecahan			
N	8	7	6	5	4	3	2	1	0	-1	-2	-3	-4
Bobot	$_2 8$	$_2 7$	$_2 6$	$_2 5$	$_2 4$	$_2 3$	$_2 2$	$_2 1$	$_2 0$	$_2 -1$	$_2 -2$	$_2 -3$	$_2 -4$
	256	128	64	32	16	8	4	2	1	0.5	0.25	0.125	0.0625

3. Bilangan Oktal

Adapun bilangan oktal merupakan sistem bilangan dengan basis 8, artinya dalam sistem ini digit yang digunakan berjumlah 8 buah yakni 0, 1, 2, 3, 4, 5, 6, dan 7. Setiap posisi digit penyusunnya memiliki bobot kepangkatan 8^n dengan n bilangan bulat positif dan negatif.

Contoh 1 :

Bilangan $(132)_8$ *atau* 132_8 memiliki arti :

$(132)_8 = 1 \times 8^2 + 3 \times 8^1 + 2 \times 8^0$

$= 16 + 24 + 2$

$= 42_{10}$

Contoh 2:

Bilangan $(132.14)_8$ atau 132.14 memiliki arti :

$(132.14)_8 = 1 \times 8^2 + 3 \times 8^1 + 2 \times 8^0 + 1 \times 8^{-1} + 4 \times 8^{-2}$

$= 64 + 24 + 2 + 1/8 + 4/64$

$= 64 + 24 + 2 + 0.125 + 0.0625$

$= 90.1875_{10}$

Tabel 4. 3 Bobot Bilangan Oktal untuk -3 ≤ n ≤ +3

	Bilangan Bulat				Bilangan Pecahan		
N	3	2	1	0	-1	-2	-3
Bobot	$_8 3$	$_8 2$	$_8 1$	$_8 0$	$_8 -1$	$_8 -2$	$_8 -3$

4. Bilangan Heksadesimal

Bilangan Heksadesimal merupakan sistem bilangan dengan dikenal basis 16, artinya dalam sistem ini digit yang digunakan berjumlah 16 buah yakni 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, dan F. Setiap digit penyusunnya memiliki bobot kepangkatan 16^n dengan n bilangan bulat positif dan negatif

Contoh 1 :

Bilangan $(AB8)_{16}$ *atau* $AB8_{16}$ memiliki arti :

$(AB8)_{16} = 10 \times 16^2 + 11 \times 16^1 + 8 \times 16^0$

$= 2560 + 176 + 8$

$= 2744_{10}$

Contoh 2 :

Bilangan $(CD7.FE)_{16}$ *atau* $CD7.FE_{16}$ memiliki arti :

$(CD7.FE)_{16} = 12 \times 16^2 + 13 \times 16^1 + 7 \times 16^0 + 15 \times 16^{-1} + 14 \times 16^{-2}$

$= 3072 + 208 + 7 + 0.9375 + 0.0547$

$= 3287.9922_{10}$

Tabel 4. 4 Bobot Bilangan desimal untuk -3 ≤ n ≤ +3

	Bilangan Bulat				Bilangan Pecahan		
N	3	2	1	0	-1	-2	-3
Bobot	16^3	16^2	16^1	16^0	$_{16}-1$	$_{16}-2$	$_{16}-3$

C. Konversi Sistem Bilangan

Prinsip dari pengkonversian ini adalah menjumlahkan nilai dari setiap digit atau lebih dikenal dengan bit dengan satu bilangan yang telah dikalikan dengan bobotnya

1. Konversi Sistem Desimal ke Biner

Bilangan biner dapat dicari dari bilangan Desimal dengan membagi terus menerus dengan angka 2, sisa dari yang terakhir sampai yang pertama merupakan angka biner yang didapat.

- **Metode Bagi Dua**

  Sistem konversi bilangan decimal ke sistem biner dengan cara bagi 2 dapat dilakukan dengan cara seperti ditunjukkan pada cara dibawah ini:

Contoh 1 :

$N = 21_{10} = (\quad)_2$

Hasil : $N = 21_{10} = (10101)_2$

Bilangan	Bagi		Hasil	Sisa Bagi
21	: 2	=	10	1
10	: 2	=	5	0
5	: 2	=	2	1
2	: 2	=	1	0

Contoh 2:

$N = 122_{10} = (\quad)_2$

Bilangan	Bagi		Hasil	Sisa Bagi
122	: 2	=	61	0
61	: 2	=	30	1
30	: 2	=	15	0
15	: 2	=	7	1
7	: 2	=	3	1
3	: 2	=	1	1

Hasil : $N = 122_{10} = (1111010)_2$

2. Metode Nilai Digit

Sistem konversi bilangan desimal ke sistem bilangan biner dengan cara ini dapat dilakukan dengan menuliskan terlebih dahulu bobot bilangan biner pecahan dan bobot bilangan biner bulat sebagai contoh dari 2^{-3} sampai dengan 2^{-8} . dengan penulisan bobot bilangan biner tersebut diurutkan dari bobot terkecil sampai bobot terbesar dari kanan ke kiri. Batas Desimal

Tabel 4. 5 Bobot Bilangan Biner Bulat dan Pecahan

	Bilangan Bulat									Bilangan Pecahan			
	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
Nilai Desimal	256	128	64	32	16	8	4	2	1	0.5	0.25	0.125	0.0625
21_{10}					1	0	1	0	1				
122_{10}			1	1	1	1	0	1	0				
122.125_{10}			1	1	1	1	0	1	0	0	0	1	

3. Konversi Sistem Desimal ke Oktal

Pengkonversian sistem bilangan desimal ke sistem oktal dapat dilakukan dengan cara bagi, dan dalam hal ini faktor pembaginya adalah 8.

Contoh 1:

$N = 1242_{10} = (\quad)_8$

Bilangan	Bagi		Hasil	Sisa Bagi
1242	: 8	=	155	2
155	: 8	=	19	3
19	: 8	=	2	3

Hasil : $N = 1242_{10} = (2332)_8$

4. Konversi Sistem Desimal ke Heksadesimal

Konversi sistem desimal ke heksadesimal dapat dilakukan dengan menggunakan metode bagi 16.

Contoh 1:

Bilangan	Bagi		Hasil	Sisa Bagi
1372	: 16	=	85	12
85	: 16	=	5	5

Hasil : $N = 1372_{10}\ (55C)_{16}$

5. Konversi Sistem Biner ke Desimal

a. Bilangan Bulat

10110_2 = … … … … … . .$_{10}$

Bilangan = $(10110)_2$ *atau* 10110_2 memiliki arti :

$(10110)_2$ = $1 \times 2^4 + 0 \times 2^3 + 1 \times 2^2 + 1 \times 2^1 + 0 \times 2^0$

= $16 + 0 + 4 + 2 + 0$

= 22_{10}

b. Bilangan Pecahan

Bilangan = $(1001.\ 101)_2$ *atau* $1001.\ 101_2$ memiliki arti:

$(1001)_2$ = $1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$

= $8 + 0 + 0 + 1$

= 9_{10}

$(0.101)_2$ = $1 \times 2^{-1} + 0 \times 2^{-2} + 1 \times 2^{-3}$

= $0.5 + 0 + 0.125$

= 0.625

$(1001.\ 101)_2$ = $9 + 0.625$

= 9.625_{10}

6. Konversi Sistem Oktal ke Desimal

a. Bilangan Bulat

$(137)_8$ = … … … … … . .$_{10}$

$(137)_8$ = $1 \times 82 + 3 \times 81 + 7 \times 80$

= $64 + 24 + 7$

= 95_{10}

b. Bilangan Pecahan

$(135.\ 25)_8$ = … … … … … . .$_{10}$

$(135)_8$ = $1 \times 8^2 + 3 \times 8^1 + 5 \times 8^0$

= $64 + 24 + 5$

= 93_{10}

$(0.25)_8$ = $2 \times 8^{-1} + 5 \times 8^{-2}$

= $2/8 + 5/64$

= $0.25 + 0.078125$

= 0.328125_{10}

$(135.25)_8$ = $93 + 0.328125$

= 93.328125_{10}

7. Konversi Sistem Heksadesimal ke Desimal

a. Bilangan Bulat

$(2BD)_{16}$ = ( )$_{10}$

$(2BD)_{16}$ = $2 \times 16^2 + 11 \times 16^1 + 13 \times 16^0$

= 512 + 176 + 13

= 701_{10}

b. Bilangan Pecahan

$(C9.\ 6A5)_{16}$ =( )$_{10}$

= $12 \times 16^1 + 9 \times 16^0 + 6 \times 16^{-1} + 10 \times 16^{-2} + 5 \times 16^{-3}$

= 192 + 9 + 6/16 + 10/256 + 5/4096

= 192 + 9 + 0.375 + 0.039 + 0.00122

= 201.41522_{10}

8. Konversi Sistem Oktal, dan Heksadesimal ke Sistem Biner

a. Konversi Sistem Oktal ke Biner

Konversi suatu bilangan oktal menjadi bilangan biner dilakukan dengan cara mengubah setiap bilangan oktal ke dalam bilangan biner 3-bit, dan jika ditemukan bit 0 pada MSB, bit tersebut dibuang.

Contoh :

$(1235)_8$ = (… … … … … …)$_2$

$(1235)_8$ = 1 2 3 5

= 001 010 011 101

= 1010011101_2

b. Konversi Sistem Heksadesimal ke Sistem Biner

Konversi ini dilakukan menggunakan cara mengubah setiap bilangan heksadesimal ke bilangan biner 4-bit. Jika ditemukan bit 0 pada MSB bit tersebut dibuang.

Contoh :

$(45A2)_{16}$ = (……)$_2$

= 4 5 A 2

= 0100 0101 1001 0010

= 100010110010010_2

c. Konversi Sistem Heksadesimal ke Oktal

Konversi ini dilakukan dengan cara mengubah setiap bilangan heksadesimal ke dalam bilangan biner (4-4-4 bit), kemudian baru konversi ke Oktal (3-3-3 bit).

$(1AD7)_{16}$ = ( )$_8$

$(1AD7)_{16}$ = (… … … … … … …)$_2$ →( )$_8$

Berikut penyelesaian dari contoh soal diatas:

1) Pertama pisahkan per-angka, kemudian konversi ke biner.

   $(1AD7)_{16}$ = 1 A D 7

   = 0001 1001 1101 0111

   = 1100111010111_2

2) Bagi dari deret biner (3-3-3 bit)

   = 1100111010111

   = 1 100 111 010 111

   = 1 4 7 2 7

3) Maka $(1AD7)_{16} = 14727_8$

d. Konversi Sistem Oktal ke Heksadesimal

Konversi ini dilakukan dengan cara mengubah setiap bilangan Oktal ke dalam bilangan biner (3-3-3) bit, kemudian baru konversi ke Heksadesimal (4-4-4) bit.

$(1235)_8$ = (… … … … … …)$_{16}$

$(1235)_8$ = (… … … … … …)$_2$ = (… … … … … …)$_{16}$

Berikut penyelesaian dari contoh soal diatas:

1) Pertama pisahkan per-angka, kemudian konversi ke biner.

   $(1235)_8$ = 1 2 3 5

   = 001 010 011 101

   = 1010011101_2

2) Bagi dari deret biner (4-4-4 bit)

   = 1010011101

   = 10 1001 1101

   = 2 9 13=D

   Maka $(1235)_8 = 29D_{16}$

D. Sistem Kode

Komputer sebenarnya tidak dapat beroperasi dengan bilangan-bilangan desimal. Melainkan yang diproses oleh komputer adalah kode-kode biner, yaitu bilangan-bilangan yang dinyatakan dengan angka 0 (nol) dan 1 (satu). Apa sih binary code/kode biner itu?

- Binary atau biner itu pengganti huruf atau abjad dalam bentuk kode
- angka 1 dan 0.
- Angka 1 dan 0 itu adalah representasi dari on dan off. Jadi, 1=on dan 0=off.
- Alasannya karena alat-alat elektronik yang terdapat dalam sebuah komputer dirancang untuk beroperasi hanya pada dua keadaan (biner).
- Sebuah komputer sering disebut sebagai pengolahan atau processor (pemroses) data.
- Tetapi dalam sistem digital, digunakan penerjemah elektronika yang disebut dengan pengkode (encoders) dan pendekode (decoder) untuk pengubahan dari kode yang satu ke kode lainnya.

Data yang diproses di dalam sistem digital umumnya direpresentasikan dengan menggunakan kode tertentu. Terdapat berbagai macam sistem kode seperti :

1. Decimal dikode biner atau binary-coded decimal (BCD),
2. Gray,
3. Excess-3,
4. Kode peraga 7 segmen, dan
5. ASCII.

Kode-kode tersebut disusun dengan cara menggunakan bilangan biner yang membentuk suatu kelompok tertentu. Kelompok bilangan biner yang membentuk suatu kode dibedakan penyebutannya.

- Kode biner 4-bit dinamakan **Nibble**, contoh 1101, 1010, dan 1110.
- Kode biner 8-bit dinamakan **Byte**, contoh 11011001, 10101100, dan 11101010.
- Kode biner 16-bit dinamakan **Word**, contoh 11011010 11101010.
- Kode biner 32-bit dinamakan **Double Word**, contoh 1101 1010 1110 1010 1010 1110 1101 1001

Dalam hal ini dinyatakan dengan :

- 1 byte = 8 bit
  1 Kilo byte = 1 KB = 1024 byte = $2^{10} byte$.

1. Kode BCD (Binary-Coded-Decimal)

- Kode BCD (binary-coded-decimal) membuat pengubahan menjadi jauh lebih mudah.
- Kode BCD merupakan suatu kode biner berbobot.
- Bit paling signifikan mempunyai bobot 8, sedangkan bit yang paling tidak signifikan hanya mempunyai bobot 1.
- Kode ini lebih tepat dikenal sebagai kode BCD 8421.
- Bagian 8421 dari nama tersebut menunjukan pembobotan dari masing- masing angka pada kode 4-bit.

Keuntungan kode BCD :

a. Dari titik pandang perangkat keras, mudah mengkonversi biner ke dan dari desimal.
b. Dapat dipakai pada mesin digital yang informasi masukan atau keluarannya disajikan dalam desimal. Misalnya, voltmeter digital, pencacah frekumsi, jam digital, dan kalkulator elektronis.

Kekurangan BCD :

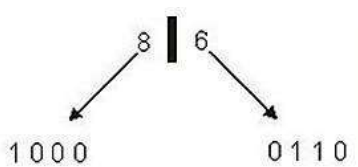
a. Memerlukan cacah bit yang lebih banyak dibandingkan dengan sandi biner lurus (*straight binary*).
b. Proses arithmatika dengan sandi BCD lebih luas dibandingkan dengan sandi biner lurus sehingga diperlukan rantai elektronis yang lebih banyak. Hal ini menurunkan laju operasi arithmatika.

Untuk lebih memahami system kode BCD, coba perhatikan konversi bilangan dari bilangan Desimal ke kode BCD berikut ini!

a. Jika Anda hendak mengkonversikan bilangan Desimal ke dalam bentuk kode BCD, terdapat dua langkah, yaitu :

   Contoh: 86_{10} = ($)_{BCD}$

   1) Pertama, yaitu membagi bilangan tersebut menjadi dua bagian.



   2) Kedua, yaitu bilangan yang sudah dibagi dua bagian, kemudian diubah ke bentuk biner dengan membagi tiap-tiap bilangan dengan angka Dua.

b. bentuk bilangan Biner, terdapat tiga langkah, yaitu dengan contoh sebagai berikut: 01110011_{BCD} = ( )$_2$

1) Pertama, yaitu membagi 4-digit pada tiap baris biner .

01110011_{BCD} □ 0111 0011_{BCD}

2) Kedua, yaitu biner yang sudah dibagi 4-bit (dua bagian), kemudian diubah ke bentuk biner dengan mengalikan tiap-tiap bilangan dengan 2^n.

0111 0011$_{BCD}$

BCD → Desimal				
0	1	1	1	
$0*2^3$	$1*2^2$	$1*2^1$	$1*2^0$	7
0	4	2	1	

0	1	0	1	
$0*2^3$	$1*2^2$	$0*2^1$	$1*2^0$	5
0	4	0	1	

75$_{10}$ =

A . B

A . $\bar{\bar{B}}$ +

A

7!

	Sisa
75 : 2	1
37 : 2	1
18 : 2	0
9 : 2	1
4 : 2	0
2 : 2	0
1	

8	Sisa
8 : 2	0
4 : 2	0
2 : 1	0
1	

6	Sisa
6 : 2	0
3 : 2	1
1 : 2	1
0	

Hasil : 8610 = (1000 0110)BCD

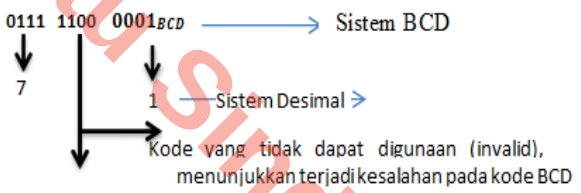
3) Ketiga, yaitu Hasil konversi kode BCD ke bilangan biner, dikonversikan lagi ke bilangan biner, dengan cara membagi dengan angka dua.

c. Jika Anda hendak mengkonversikan kode BCD ke dalam Hasil tersebut adalah sebagai berikut : 0111 $0011_{BCD} = 1001\ 1011_2$

d. Jika Anda hendak mengkonversikan kode BCD ke dalam bentuk bilangan desimal, terdapat satu langkah, yaitu :

Contoh : Contoh : 011111000001_{BCD} = ( )$_2$

- Bagi 4-bit pada baris biner dari sebelah kanan.

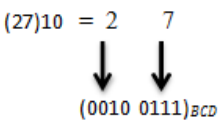


Kode BCD Hanya bernilai desimal 0 s.d 9

e. Jika Hendak mengkonversi bilangan Biner ke kode BCD terdapat dua langkah, yaitu :
Contoh : 01110011_2 = ( )$_{BCD}$

1) Untuk konversi kedalam bentuk bilangan BCD yaitu terlebih dahulu Anda harus mengubahnya kedalam bentuk desimal.
1101101_2 = (… … … … …)$_{BCD}$
= $1 \times 2^0 + 1 \times 2^1 + 0 \times 2^2 + 1 \times 2^3 + 1 \times 2^4 + 0 \times 2^5 + 1 \times 2^6$
= 1+2+0+8+16+0 = $(27)_{10}$

2) Setelah itu baru Anda membaginya kedalam empat-empat bagian. Untuk jelasnya lihat cara kerja berikut ini.



2. Kode Excess-3 (XS-3)

Sistem kode lain yang mirip dengan BCD adalah Excess-3. Untuk menyusun kode XS-3 dari suatu bilangan desimal, masing-masing digit dari suatu bilangan desimal yang akan dikode dengan XS-3, ditambah dengan 3 desimal, kemudian hasilnya dikonversi seperti cara pada konversi BCD.

Beberapa kode biner adalah kode biner yang tak berbobot.

- Dua kode biner yang tak berbobot adalah kode ekses 3 (excess3) dan kode kelabu (gray).
- Kode excess 3 (XS-3) berhubungan dengan kode BCD 8421 ini disebabkan oleh sifat BCD-nya.
- Namun demikian, masing-masing kelompok 4-bit dalam kode XS-3 sama dengan suatu digit desimal tertentu.
- Angka pada XS-3 adalah selalu tiga angka lebih besar daripada angka BCD 8421.

Contoh : Tulis dalam bentuk kode XS-3 bilangan 158_{10}

Jawab :

1 + 3 = 4 dan ekuivalennya adalah **0100**

5 + 3 = 8 dan ekuivalennya adalah **1000**

8 + 3 = 11 dan ekuivalennya adalah **1011**

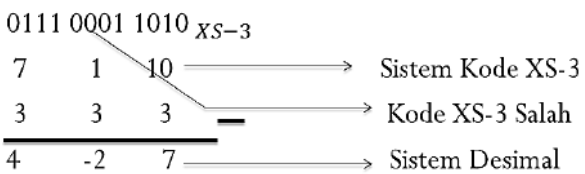
Jadi 158_{10} = 0100 1000 1011_{EX-3}

Pada XS-3, terdapat 6 kode yang tidak dapat digunakan yakni: 0000, 0001, 0010, 1101, 1110 dan 1111.

Contoh :

Ubah kode XS-3 : 0111 0001 1010 $_{XS-3}$ ke sistem desimal

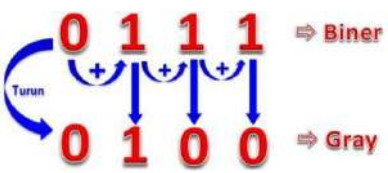
Jawab:



3. Kode Gray

- Kode kelabu (*Gray code*) merupakan salah satu kode biner yang tak berbobot selain dari kode eksess (excess) 3.
- Kode kelabu bukan merupakan kode jenis BCD. Masing-masing kenaikan hitungan (penambahan) pada kode kelabu dilakukan dengan pengubahan satu bit saja.
- Dalam biner 4 bit, semua berubah keadaan (dari 0111 ke 1000). Dalam garis yang sama, kode kelabu hanya mempunyai perubahan keadaan pada bit sebelah kiri (0100 ke 1100).
- Kode Gray memiliki keunikan yakni setiap kali kode itu berubah nilainya secara berurutan misalnya dari 2 ke 3 atau dari 5 ke 6, hanya terdapat 1 bit saja yang berubah. Contoh: jika nilai kode Gray berubah dari 0011 ke 0010.
- Kode Gray biasanya digunakan sebagai data yang menunjukkan posisi dari suatu poros mesin yang berputar.

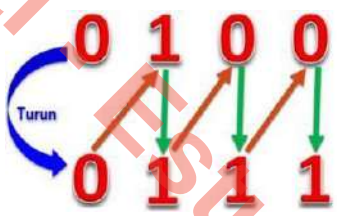
Cara mengubah bilangan Desimal ke dalam kode Gray adalah sebagai berikut:
Ubah 7_{10} dalam bentuk kode Gray!



1 + 1	0
0 + 0	0
1 + 0	1
0 + 1	1

Hasil: $7_{10} = 0111_{10} = 0100_{Gray}$

Cara mengubah Kode Gray ke dalam Bilangan Desimal adalah sebagai berikut:
Ubah 0100_{Gray} dalam bentuk Bilangan Desimal!



1 + 1	0
0 + 0	0
1 + 0	1
0 + 1	1

Hasil: $0100_{Gray} = 0111_2 = 7_{10}$

4. Kode 7-Segment Display

Hasil pemprosesan suatu sinyal dari suatu rangkaian digital merupakan sinya digital dalam bentuk kode-kode biner. Jika hasil tersebut tetap disajikan dalam bentuk aslinya yakni kode biner, maka akan mengalami kesulitan didalam membacanya karena kita tidak terbiasa menggunakan kode biner dalam kehidupan sehari-harinya. Agar menjadi mudah dibaca, maka kode- kode biner tersebut perlu diubah tampilannya menggunakan tampilan bilangan desimal. Piranti yang digunakan untuk menampilkan data dalam bentuk bilangan desimal adalah LED Seven Segment Display atau dinamakan peraga atau output 7-segmen.

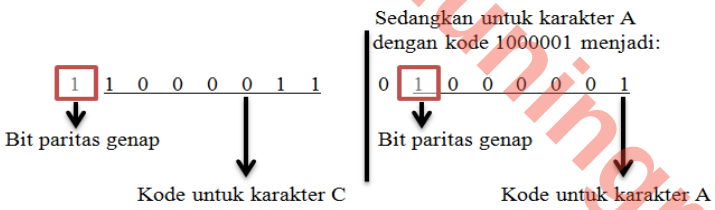
5. Kode ASCII

ASCII (*American Standart Code for Information Interchange*) merupakan kode biner untuk merepresentasikan bilangan, huruf, dan simbol, sehingga disebut juga kode alfanumerik. Kode alfanumerik yang lain, misalnya EB-CDIC (*Extended Binary-Coded Decimal Interchange Code*). Komunikasi data yang terjadi antar bagian-bagian di dalam sistem komputer atau antar komputer yang satu dengan lainnya memungkinkan terjadinya kesalahan pada bagian-bagian data. Untuk

mendeteksi kemungkinan adanya kesalahan pada kode-kode tersebut, ditambahkan bit yang disebut *parity bit* atau bit paritas yang ditempatkan sebagai MSB. Terdapat dua buah metode paritas yang digunakan oleh suatu sistem untuk mendeteksi adanya kesalahan yakni metode paritas genap dan metode paritas ganjil. Dalam metode genap, nilai bit paritas dipilih sedemikian rupa sehingga jumlah bit 1 dalam suatu kode ASCII (termasuk bit paritasnya adalah genap).

Contoh:

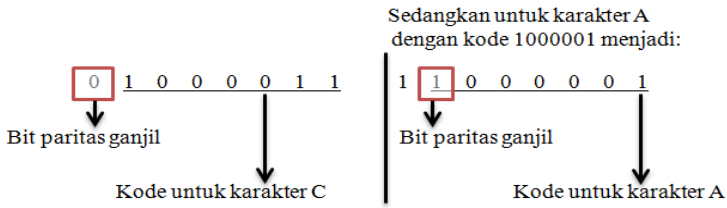
Kode ASCII untuk karakter C adalah 1000011, memiliki jumlah bit 1 ganjil yakni 3 buah. Karena digunakan metode paritas genap, maka bit paritas yang ditambahkan adalah 1, sehingga kodenya menjadi:



Gambar 4. 1 Bit Parintas Genap Untuk Karakter C dan A

Contoh:

Kode ASCII untuk karakter C adalah 1000011, memiliki jumlah bit 1 ganjil yakni 3 buah. Karena digunakan metode paritas ganjil, maka bit paritas yang ditambahkan adalah 0, sehingga kodenya menjadi:



Gambar 4. 2 Bit Parintas Ganjil Untuk Karakter C dan A

Dalam pengkodean system kode ASCII memanfaatkan 8 bit. Pada saat ini kode ASCII telah tergantikan oleh kode UNICODE (Universal Code). UNICODE dalam pengkodeannya memanfaatkan 16 bit sehingga memungkinkan untuk menyimpan kode-kode lainnya seperti kode bahasa Jepang, Cina, Thailand dan lain sebagainya.

Pada papan keyboard, aktifkan numlock, tekan tombol ALT secara bersamaan dengan kode karakter maka akan dihasilkan karakter tertentu. Misalnya: ALT + 44 maka akan muncul karakter koma (,). Mengetahui kode- kode ASCII sangat bermanfaat misalnya untuk membuat karakter-karakter tertentu yang tidak ada di keyboard.

Pada tabel 6 Menunjukkan nilai heksadesimal dari kode ASCII 7-bit. Untuk memperoleh nilai biner dari kode ASCII tersebut, Anda harus mengubah nilai tercantum nilai heksadesimal dari karakter C adalah 43, maka nilai binernya adalah 1000011. Untuk karakter A yang memiliki nilai heksadesimal 41, nilai binernya adalah 1000001.

Dari uraian tentang sistem bilangan dan sistem kode di atas dapat dituliskan tingkasan nilai biner, oktal, dan heksadesimal, serta nilai kode BCD, XS-3, Gray dan kode peraga 7-segmen untuk bilangan desimal 0 sampai dengan 15 seperti pada tabel 7

Tabel 4.6 a Nilai Heksadesimal Untuk Beberapa Kode ASCII 7-bit

Simbol	Kode ASCII	Simbol	Kode ASCII	Simbol	Kode ASCII	Simbol	Kode ASCII	Simbol	Kode ASCII
0	30	I	49	b	62	{	7B	¶	B6
1	31	J	4A	c	63	|	7C	·	B7
2	32	K	4B	d	64	{	7D	¸	B8
3	33	L	4C	e	65	~	7E	¹	B9
4	34	M	4D	f	66	DEL	7F	º	BA
5	35	N	4E	g	67	¡	A1	»	BB
6	36	O	4F	h	68	¢	A2	¼	BC
7	37	P	50	i	69	£	A3	½	BD
8	38	Q	51	j	6A	¤	A4	¾	BE
9	39	R	52	k	6B	¥	A5	¿	BF
!	21	:	3A	S	53	L	6C	¦	A6
"	22	;	3B	T	54	M	6D	§	A7
#	23	<	3C	U	55	N	6E	¨	A8
$	24	=	3D	V	56	O	6F	©	A9
%	25	>	3E	W	57	P	70	ª	AA
&	26	?	3F	X	58	Q	71	«	AB
„	27	@	40	Y	59	R	72	¬	AC
(	28	A	41	Z	5A	S	73	®	A AE
)	29	B	42	[	5B	T	74	¯	AF
*	2A	C	43	]	5C	U	75	°	B0

Tabel 4.6 b Nilai Heksadesimal Untuk Beberapa Kode ASCII 7-bit

Simbol	Kode ASCII	Simbol	Kode ASCII	Simbol	Kode ASCII	Simbol	Kode ASCII	Simbol	Kode ASCII
+	2B	D	44	^	5D	V	76	±	B1
,	2C	E	45	_	5E	W	77	²	B2
-	2D	F	46	„	5F	X	78	³	B3
.	2E	G	47	`	60	Y	79	´	B4
/	2F	H	48	A	61	Z	7A	µ	B5

Tabel 4.7 a Nilai Berbagai Sistem Bilangan dan Kode untuk Bilangan Desimal 0 s.d 15

Desimal	Biner	Oktal	Heksa Decimal	BCD		Gray	Peraga 7-Segmen	
				8421	XS-3		Abcdefg	Display
0	0	0	0	0000			1111110	0
1	1	1	1	0001			0110000	1
2	10	2	2	0010			1101101	2
3	11	3	3	0011			1111001	3
4	100	4	4	0100			0110011	4
5	101	5	5	0101			1011011	5

Tabel 4.7 b Nilai Berbagai Sistem Bilangan dan Kode untuk Bilangan Desimal 0 s.d 15

Desimal	Biner	Oktal	Heksa Decimal	BCD		Gray	Peraga 7-Segmen	
				8421	XS-3		Abcdefg	Display
6	110	6	6	0110			1011111	6
7	111	7	7	0111			1110000	7
8	1000	10	8	1000			1111111	8
9	1001	11	9	1001			1110011	9
10	1010	12	A	0001 0000			1111101	A
11	1011	13	B	0001 0001			0011111	B
12	1100	14	C	0001 0010			0001101	C
13	1101	15	D	0001 0011			0111101	D
14	1110	16	E	0001 0100			1101111	E
15	1111	17	F	0001 0101			1000111	F

SOAL – SOAL LATIHAN

1. Ubahlah ke dalam sistem desimal nilai:
   a. 00110101_2
   b. 101001000.111_2
   c. $15BA_{16}$
   d. $4A4E.E2_{16}$
2. Lakukan konversi ke sistem biner dengan menggunakan metode nilai digit dan metode bagi dua bilangan desimal berikut ini, dan lakukan konversi balik untuk memeriksa kebenaran konversi yang Anda lakukan!
   a. 132_{10}
   b. 434_{10}
   c. 155.375_{10}
   d. 112.125_{10}
3. Lakukan konversi ke nilai desimal:
   a. 000101011000_{BCD}
   b. $0101001110010101 0_{BCD}$
   c. 1001001101011110_{XS-3}
   d. $1011001101011101011_{XS-3}$
4. Susunlah ke dalam kode BCD, Gray, dan XS-3 untuk bilangan desimal
   a. 125_{10}
   b. 1124_{10}
   c. 1211_{10}
   d. 187_{10}

BAB V
GERBANG LOGIKA DASAR

A. Pengertian Gerbang Logika

"Gerbang logika" atau gerbang logik adalah suatu entitas dalam elektronika dan matematika boolean yang mengubah satu atau beberapa masukan logik menjadi sebuah sinyal keluaran logik. Gerbang logika terutama diimplementasikan secara elektronis menggunakan dioda atau transistor, akan tetapi dapat pula dibangun menggunakan susunan komponen-komponen yang memanfaatkan sifat-sifat elektromagnetik (relay). Logika merupakan dasar dari semua penalaran (reasoning). Untuk menyatukan beberapa logika, kita membutuhkan operator logika dan untuk membuktikan kebenaran dari logika, kita dapat menggunakan tabel kebenaran.

Tabel kebenaran menampilkan hubungan antara nilai kebenaran dari proposisi atomik. Dengan tabel kebenaran, suatu persamaan logika ataupun proposisi bisa dicari nilai kebenarannya. Tabel kebenaran pasti mempunyai banyak aplikasi yang dapat diterapkan karena mempunyai fungsi tersebut. Salah satu dari aplikasi tersebut yaitu dengan menggunakan tabel kebenaran kita dapat mendesain suatu rangkaian logika. Dalam makalah ini akan dijelaskan bagaimana peran dan kegunaan tabel kebenaran dalam proses pendesainan suatu rangkaian logika. Tabel kebenaran harus memuat seluruh kemungkinan keadaan input yang ada. Jumlah seluruh kemungkinan keadaan input tergantung pada jumlah variable input atau jumlah saluran input dari suatu rangkaian logika, dan mengikuti rumus :

Jumlah seluruh kemungkinan Input = 2^n

Dengan n merupakan jumlah variable atau saluran input rangkaian

Contoh :

- Rangkaian logika dengan satu variable input

Gambar 5. 1 Diagram blok rangkaian logika dengan satu variable input

Jumlah seluruh kemungkinan input = $2^n = 2^1 = 2$ buah. Tabel kebenaran rangkaian logika dengan satu variable input ditunjukkan pada tabel dibawah ini.

Tabel 5. 1 Tabel kebenaran rangkaian logika dengan satu variable input

INPUT	OUTPUT
A	B
0	.....
1	.....

- Rangkaian logika dengan dua input

Gambar 5. 2 Diagram blok rangkaian logika dengan dua variable input

Jumlah seluruh kemungkinan input = $2^n = 2^2 = 4$ buah.

Tabel kebenaran rangkaian logika dengan satu variable input ditunjukkan pada tabel dibawah ini

Tabel 5. 2 Tabel kebenaran rangkaian logika dengan dua variable input

INPUT		OUTPUT
A	B	Y
0	0	.....
0	1	.....
1	0	.....
0	0	.....

Gerbang yang diterjemahkan dari istilah asing gate, adalah elemen dasar dari semua rangkaian yang menggunakan sistem digital. Semua fungsi digital pada dasarnya tersusun atas gabungan beberapa gerbang logika dasar yang disusun berdasarkan fungsi yang diinginkan. Gerbang -gerbang dasar ini bekerja atas dasar logika tegangan yang digunakan dalam teknik digital.Logika tegangan adalah asas dasar bagi gerbang-gerbang logika. Dalam teknik digital apa yang dinamakan logika tegangan adalah dua kondisi tegangan yang saling berlawanan. Kondisi tegangan "ada tegangan" mempunyai istilah lain "berlogika satu" (1) atau "berlogika tinggi" (high), sedangkan "tidak ada tegangan" memiliki istilah lain "berlogika nol" (0) atau "berlogika rendah" (low). Dalam membuat rangkaian logika kita menggunakan gerbang-gerbang logika yang sesuai dengan yang dibutuhkan. Rangkaian digital adalah sistem yang mempresentasikan sinyal sebagai nilai diskrit. Dalam sebuah sirkuit digital,sinyal direpresentasikan dengan satu dari dua

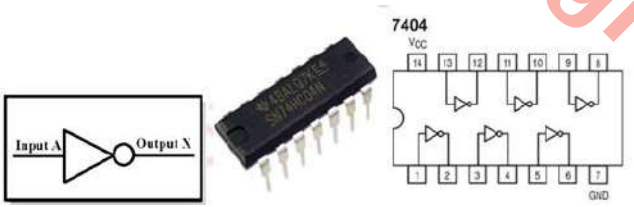
macam kondisi yaitu 1 (high, active, true,) dan 0 (low, nonactive,false).” (Sendra, Smith, Keneth C).

B. Macam – Macam Gerbang Logika

1. Gerbang Logika Dasar

a. Gerbang NOT

Gerbang NOT (NOT GATE) atau juga bisa disebut dengan pembalik (inverter) memiliki fungsi membalik logika tegangan inputnya pada outputnya. Sebuah inverter (pembalik) adalah gerbang dengan satu sinyal masukan dan satu sinyal keluaran dimana keadaan keluaranya selalu berlawanan dengan keadaan masukan. Membalik dalam hal ini adalah mengubah menjadi lawannya. Karena dalam logika tegangan hanya ada dua kondisi yaitu tinggi dan rendah atau “1” dan “0”, maka membalik logika tegangan berarti mengubah “1” menjadi "0” atau sebaliknya mengubah “0” menjadi “1”.

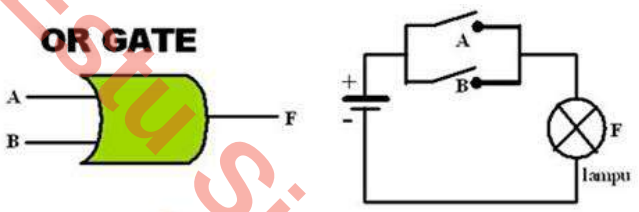


Gambar 5. 3 Simbol gerbang NOT (NOT Gate), Komponen IC NOT, dan rangkaian dalam digitalnya.

b. Gerbang OR

Gerbang OR (OR GATE) berbeda dengan gerbang NOT yang hanya memiliki satu input, gerbang ini memiliki paling sedikit 2 jalur input. Artinya inputnya bisa lebih dari dua, misalnya empat atau delapan. Yang jelas adalah semua gerbang logika selalu

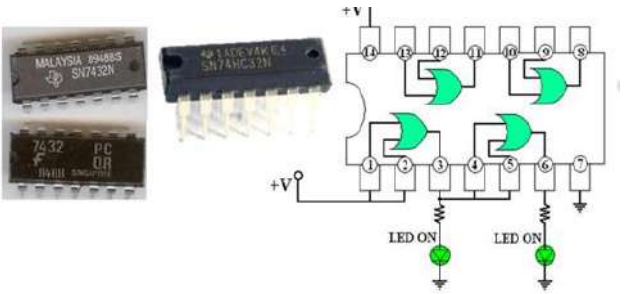
mempunyai hanya memiliki satu output. Gerbang OR akan memberikan sinyal keluaran tinggi atau high jika salah satu atau semua sinyal masukan bernilai tinggi atau high, sehingga dapat dikatakan bahwa gerbang OR hanya memiliki sinyal keluaran rendah jika semua sinyal masukan bernilai rendah.



A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

F = A + B

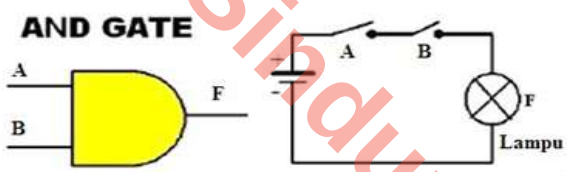
Gambar 5. 4 Simbol gerbang OR (OR Gate), Rangkaian listrik ekuivalen dengan menggunakan saklar dan tabel kebenaran.



Gambar 5. 5 Komponen IC OR, dan rangkaian dalam digitalnya

c. Gerbang AND

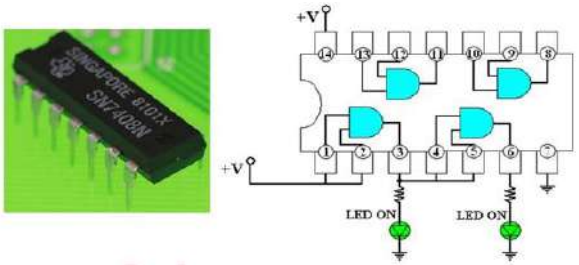
Gerbang AND (AND GATE) atau dapat pula disebut gate AND,adalah suatu rangkaian logika yang mempunyai beberapa jalan masuk (input) dan hanya mempunyai satu jalan keluar (output). Gerbang AND mempunyai dua atau lebih dari dua sinyal masukan tetapi hanya satu sinyal keluaran. Dalam gerbang AND, untuk menghasilkan sinyal keluaran tinggi maka semua sinyal masukan harus bernilai tinggi atau high dan akan menghasilkan keluaran rendah jika ada salah satu input bernilai rendah.



A	B	F
0	0	0
0	1	0
1	0	0
1	1	1

F = A.B

Gambar 5. 6 Simbol gerbang AND (AND Gate), Rangkaian listrik ekuivalen dengan menggunakan saklar dan tabel kebenaran

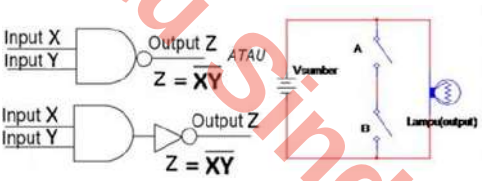


Gambar 5. 7 Komponen IC AND, dan rangkaian dalam digitalnya

2. Gerbang Logika Kombinasi

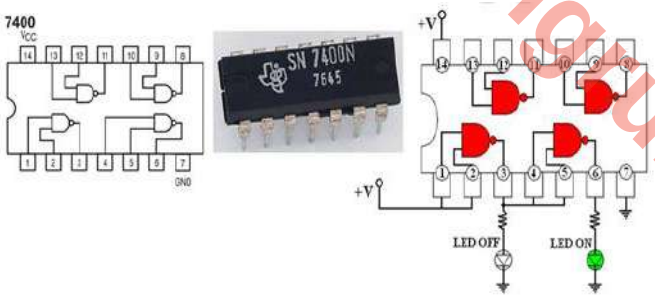
a. Gerbang NAND

Gerbang NAND (NAND GATE) adalah suatu NOT-AND, atau suatu fungsi AND yang dibalikkan. Dengan kata lain bahwa gerbang NAND akan menghasilkan sinyal keluaran rendah jika semua sinyal masukan bernilai tinggi dan akan menghasilkan keluaran tinggi jika salah satu masukan rendah.



X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

Gambar 5. 8 Simbol gerbang NAND (NAND Gate), Rangkaian listrik ekuivalen dengan menggunakan saklar dan tabel kebenaran

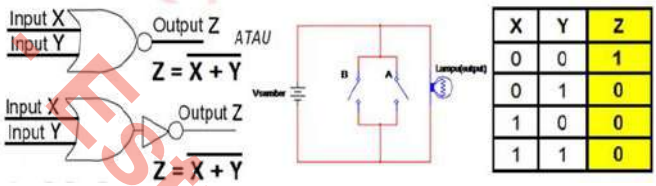


Gambar 5. 9 Komponen IC NAND, dan rangkaian dalam digitalnya.

b. Gerbang NOR

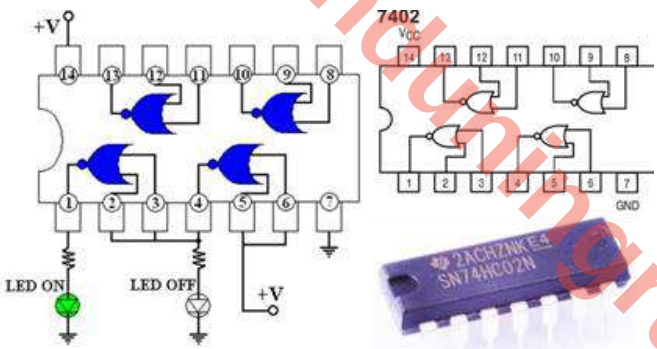
Gerbang NOR (NOR GATE) adalah suatu NOT-OR, atau suatu fungsi OR yang dibalikkan sehingga dapat dikatakan bahwa gerbang NOR akan menghasilkan sinyal keluaran tinggi jika semua sinyal

masukanya bernilai rendah, dan akan menghasilkan sinyal keluaran rendah jika salah satu input bernilai tinggi.



X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

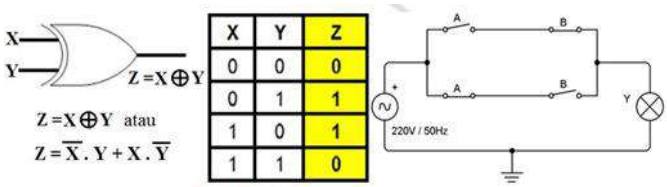
Gambar 5. 10 Simbol gerbang NOR (NOR Gate), Rangkaian listrik ekuivalen dengan menggunakan saklar dan tabel kebenaran



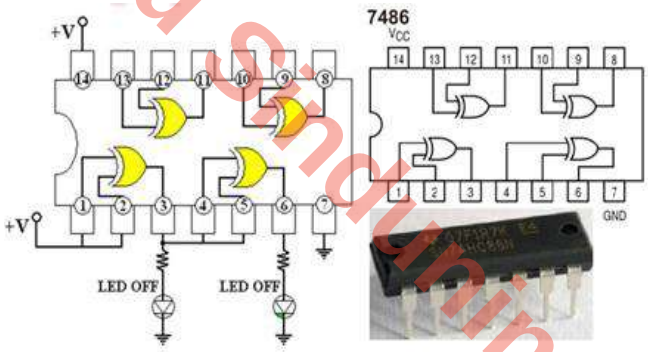
Gambar 5. 11 Komponen IC NOR, dan rangkaian dalam digitalnya.

c. Gerbang X-OR

Gerbang X-OR (X-OR GATE) akan menghasilkan sinyal keluaran rendah jika semua sinyal masukan bernilai rendah atau semua masukan bernilai tinggi atau dengan kata lain bahwa X-OR akan menghasilkan sinyal keluaran rendah jika sinyal masukan bernilai sama semua.



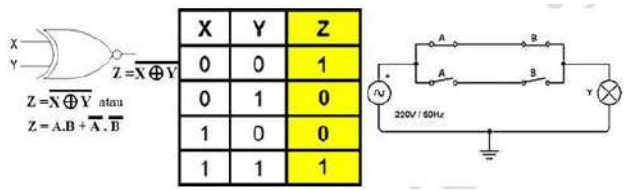
Gambar 5. 12 **Simbol gerbang X-OR (X-OR Gate), Rangkaian listrik ekuivalen dengan menggunakan saklar dan tabel kebenaran**



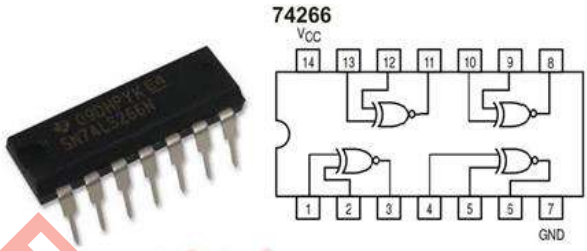
Gambar 5. 13 **Komponen IC X-OR, dan rangkaian dalam digitalnya**

d. Gerbang X-NOR

Gerbang X-NOR (X-NOR GATE) akan menghasilkan sinyal keluaran tinggi jika semua sinyal masukan bernilai sama (kebalikan dari gerbang X-OR).



Gambar 5. 14 **Simbol gerbang X-NOR (X-NOR Gate), Rangkaian listrik ekuivalen dengan menggunakan saklar dan tabel kebenaran**



Gambar 5. 15 Komponen IC X-NOR, dan rangkaian dalam digitalnya

Tabel 5. 3 Gerbang Logika

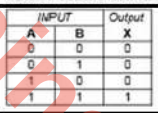
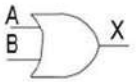
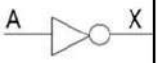
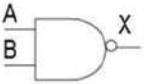
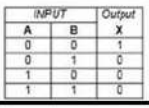
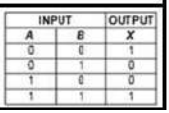
AND (7408)	NAND (7400)	XOR (7486)
0 . 0 = 0	$\overline{0.0} = 1$	$0 \oplus 0 = 0$
0 . 1 = 0	$\overline{0.1} = 1$	$0 \oplus 1 = 1$
1 . 1 = 0	$\overline{1.0} = 1$	$1 \oplus 0 = 1$
1 . 1 = 1	$\overline{1.1} = 0$	$1 \oplus 1 = 0$
OR (7432)	**NOR (7402)**	**XNOR(74266)**
0 + 0 = 0	$\overline{0+0} = 1$	$\overline{0 \oplus 0} = 1$
0 + 1 = 1	$\overline{0+1} = 0$	$\overline{0 \oplus 1} = 0$
1 + 1 = 1	$\overline{1+0} = 0$	$\overline{1 \oplus 0} = 0$
1 + 1 = 1	$\overline{1+1} = 0$	$\overline{1 \oplus 1} = 1$
	NOT (7404)	
	0 = 1	
	1 = 0	

SOAL – SOAL LATIHAN

1. Jelaskan mengenai gerbang logika dasar, dan gerbang logika lanjutan.
2. Perhatikan Inputan dibawah ini, apakah hasil output yang didapatkan

INPUT			OUTPUT
A	B	C	Y = A⊕B⊕C
0	0	1	
1	0	1	
0	1	0	
1	1	0	
1	1	1	

3. Isilah tabel dibawah ini

No	NAMA	TIPE IC	Simbol Logika	Persamaan	Tabel Kebenaran
1		7408		X=A.B	
2				X=A+B	
3	NOT				
4				$X=\overline{A.B}$	
5	NOR				
6	Ex-OR				
7				$X=\overline{A \oplus B}$	

BAB VI
PRAKTEK GERBANG LOGIKA MENGGUNAKAN PROTEUS PROFESIONAL

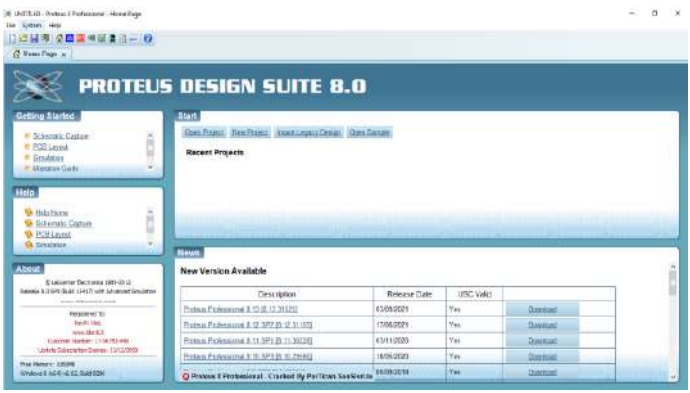
A. Langkah awal Proteus

Keuntungan menggunakan simulasi rangkaian elektronika dengan software proteus sebagai berikut :

- Software ini dapat melakukan simulasi mikrokontroler sampai dengan tingkat program.
- Library yang cukup lengkap.
- Terdapat dua program daam satu softaware yaitu **ISIS** yang digunakan untuk simulasi atau skematik dan **ARES** yang digunakan untuk pembuatan PCB. Jadi apabila rangkaian yang telah kita simulasikan telah berhasil maka tinggal convert file rangkaian ke dalam format PCB-nya.
- Tampilan yang cukup mudah dipahami.

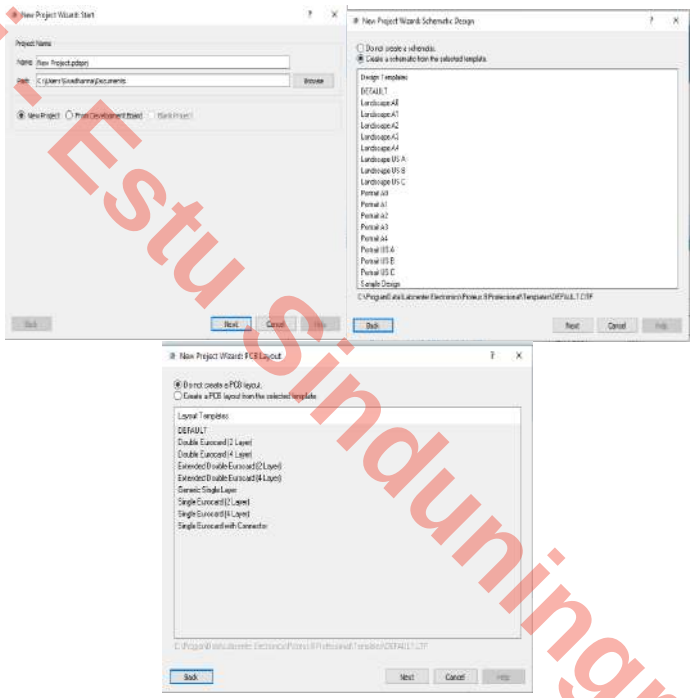
Adapun Langkah-langkah untuk menggunakan Proteus Profesional adalah sebagai berikut:

1. Membuka aplikasi proteus



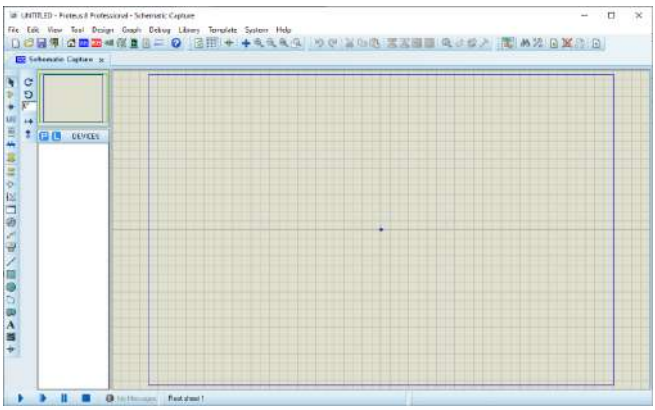
Gambar 6. 1 tampilan awal proteus

2. Pilih New Project – isi nama- next – klik default (karena hanya schematic) – pilih tidak menggunakan layout – next.



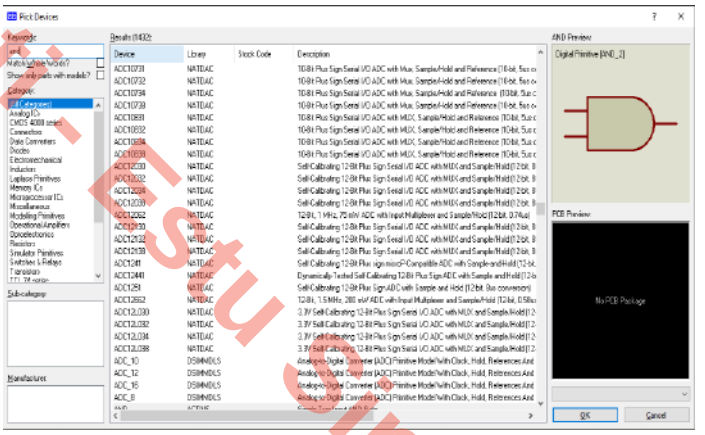
Gambar 6. 2 New project dengan

3. Tampilan halaman utama proteus schematic



Gambar 6. 3 tampilan awal proteus

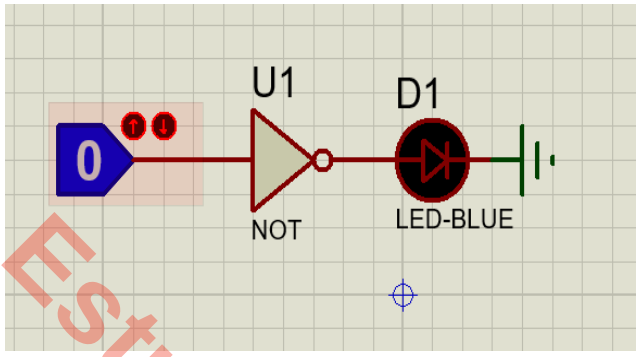
4. Pilih komponen dengan cara klik huruf P



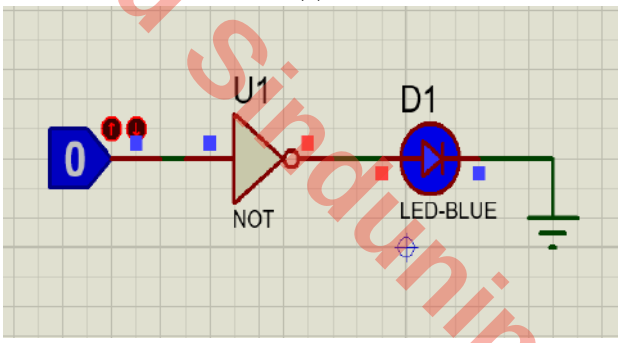
Gambar 6. 4 tampilan awal proteus

B. Percobaan dengan Gerbang Logika

1. Percobaan dengan NOT
   - Pilih komponen yang akan diuji coba
     Gerbang NOT
     Logicstate
     Ground
     Led
   - Letakan sesuai dengan keinginan
   - Untuk menyambungkan atau membuat garis, kita bisa memilih 2D graphics Line Mode atau dengan mendekatkan kursor ke kaki komponen hingga kursor berubah menjadi pensil, kemudian tarikk garis dari satu komponen ke komponen yang lain.
   - Lalu klik run



(a)



(b)

Gambar 6. 5 hasil Run kondisi 0 dan 1 pada input

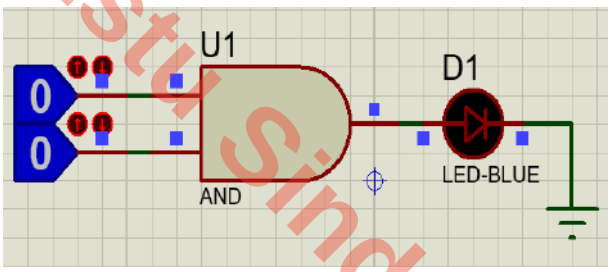
- Sesuai dengan tabel kebenaran

Tabel 6. 1 Tabel Kebenaran Gerbang NOT

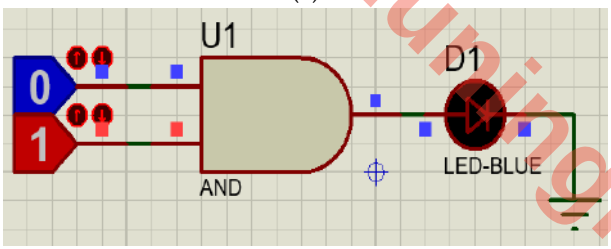
Input	Output
0	1
1	0

2. Percobaan dengan AND
   - Pilih komponen yang akan diuji coba
     - Gerbang AND
     - Logicstate
     - Ground
     - Led

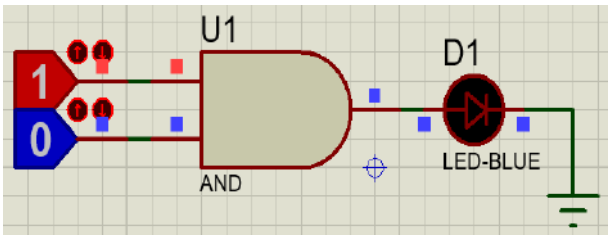
- Letakan sesuai dengan keinginan
- Untuk menyambungkan atau membuat garis, kita bisa memilih 2D graphics Line Mode atau dengan mendekatkan kursor ke kaki komponen hingga kursor berubah menjadi pensil, kemudian tarikk garis dari satu komponen ke komponen yang lain.
- Lalu klik run



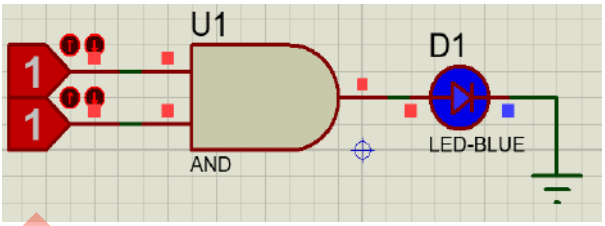
(a)



(b)



(c)



(d)

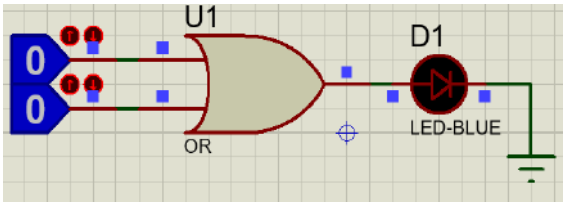
Gambar 6. 6 hasil Run

- Sesuai dengan tabel kebenaran

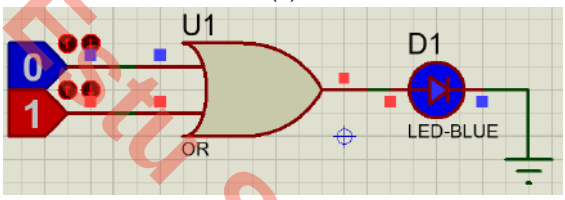
Tabel 6. 2 Tabel Kebenaran Gerbang AND

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

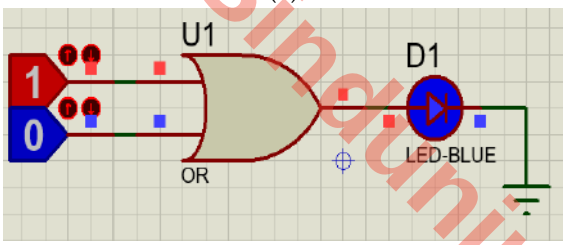
3. Percobaan dengan OR
   - Pilih komponen yang akan diuji coba
     - Gerbang OR
     - Logicstate
     - Ground
     - Led
   - Letakan sesuai dengan keinginan
   - Untuk menyambungkan atau membuat garis, kita bisa memilih 2D graphics Line Mode atau dengan mendekatkan kursor ke kaki komponen hingga kursor berubah menjadi pensil, kemudian tarikk garis dari satu komponen ke komponen yang lain.
   - Lalu klik run



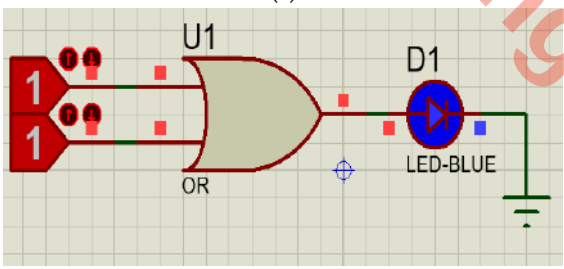
(a)



(b)



(c)



(d)

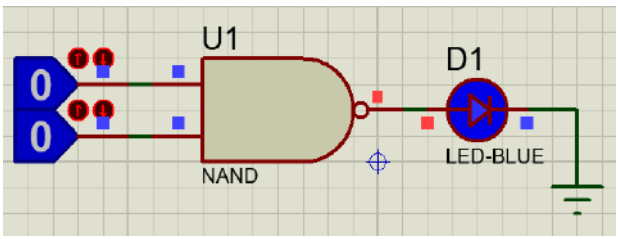
Gambar 6. 7 hasil Run

- Sesuai dengan tabel kebenaran

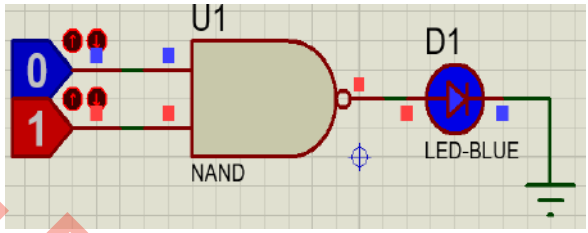
Tabel 6. 3 Tabel Kebenaran Gerbang OR

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

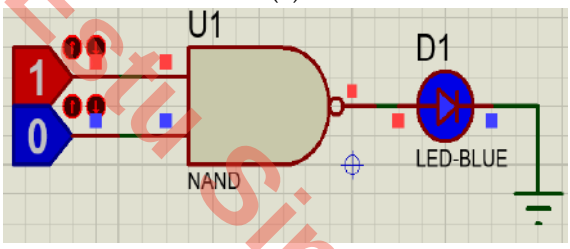
4. Percobaan dengan NAND
   - Pilih komponen yang akan diuji coba
     - Gerbang NAND
     - Logicstate
     - Ground
     - Led
   - Letakan sesuai dengan keinginan
   - Untuk menyambungkan atau membuat garis, kita bisa memilih 2D graphics Line Mode atau dengan mendekatkan kursor ke kaki komponen hingga kursor berubah menjadi pensil, kemudian tarikk garis dari satu komponen ke komponen yang lain.
   - Lalu klik run



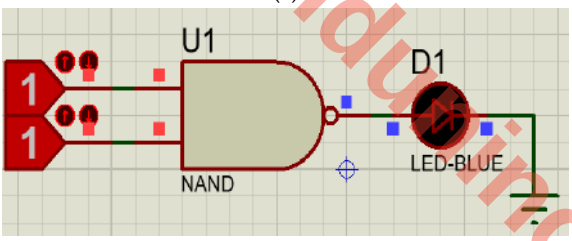
(a)



(b)



(c)



(d)

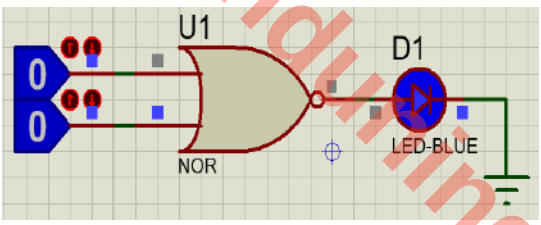
Gambar 6. 8 hasil Run

- Sesuai dengan tabel kebenaran

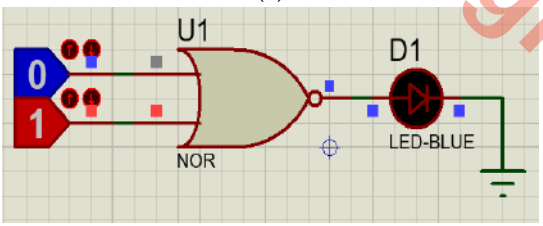
Tabel 6. 4 Tabel Kebenaran Gerbang NAND

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

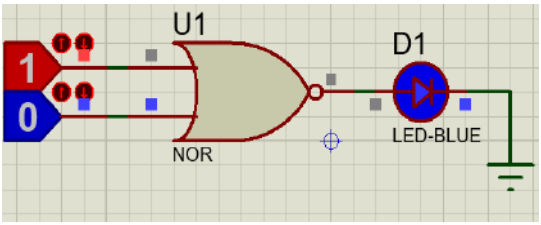
5. Percobaan dengan NOR
   - Pilih komponen yang akan diuji coba
     - Gerbang NOR
     - Logicstate
     - Ground
     - Led
   - Letakan sesuai dengan keinginan
   - Untuk menyambungkan atau membuat garis, kita bisa memilih 2D graphics Line Mode atau dengan mendekatkan kursor ke kaki komponen hingga kursor berubah menjadi pensil, kemudian tarikk garis dari satu komponen ke komponen yang lain.
   - Lalu klik run



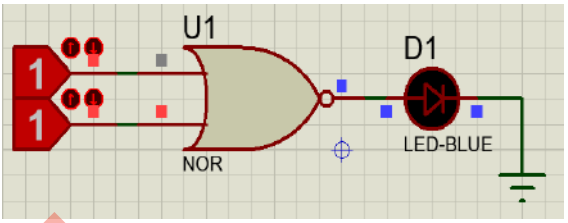
(a)



(b)



(c)



(d)

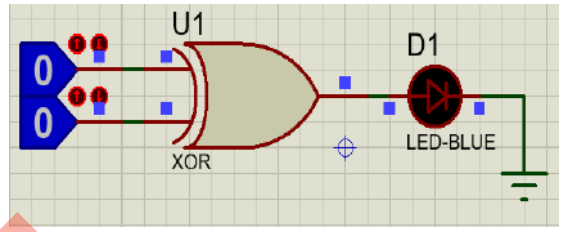
Gambar 6. 9 hasil Run

- Sesuai dengan tabel kebenaran

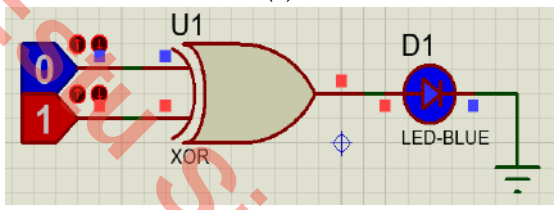
Tabel 6. 5 Tabel Kebenaran Gerbang NOR

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

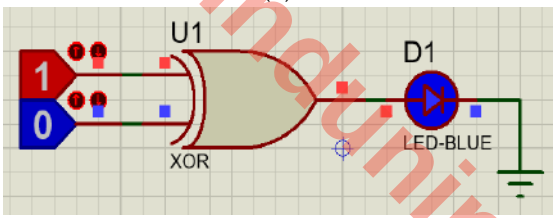
6. Percobaan dengan XOR
   - Pilih komponen yang akan diuji coba
     - Gerbang XOR
     - Logicstate
     - Ground
     - Led
   - Letakan sesuai dengan keinginan
   - Untuk menyambungkan atau membuat garis, kita bisa memilih 2D graphics Line Mode atau dengan mendekatkan kursor ke kaki komponen hingga kursor berubah menjadi pensil, kemudian tarikk garis dari satu komponen ke komponen yang lain.
   - Lalu klik run



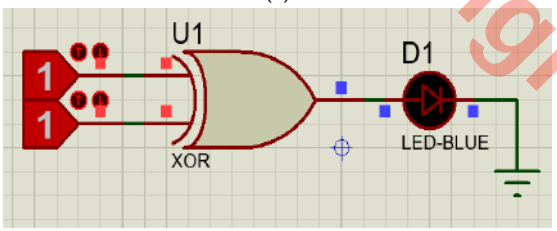
(a)



(b)



(c)



(d)

Gambar 6. 10 hasil Run

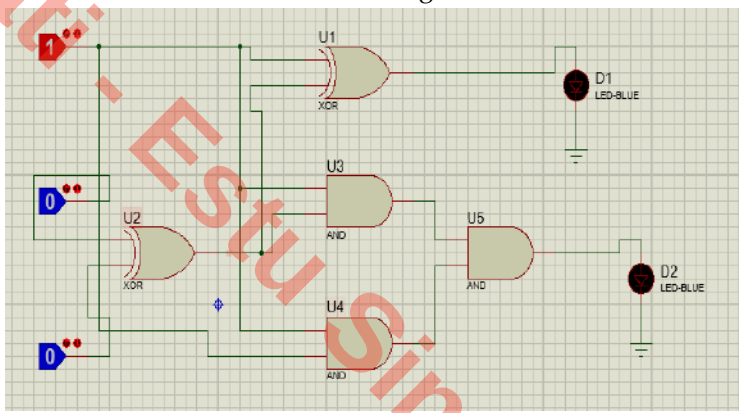
- Sesuai dengan tabel kebenaran

Tabel 6. 6 Tabel Kebenaran Gerbang XOR

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

SOAL – SOAL LATIHAN

Buat lah tabel kebenaran untuk rangkaian dibawah ini :



BAB VII
TEORI RANGKAIAN LOGIKA

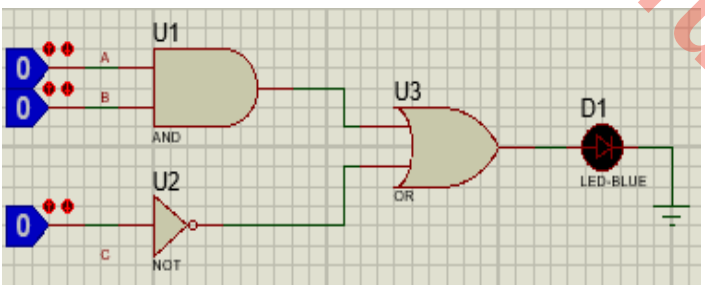
A. Pengertian Rangkaian Logika

Rangkaian logika dapat dideskripsikan dalam dua bentuk yakni dengan menggunakan simbol elemen logika dan menggunakan persamaan logika/ekspresi Boole. "Semua rangkaian logika dapat digolongkan atas dua jenis, yaitu rangkaian kombinasi (*combinational circuit*) dan rangkaian berurut (*sequential circuit*).

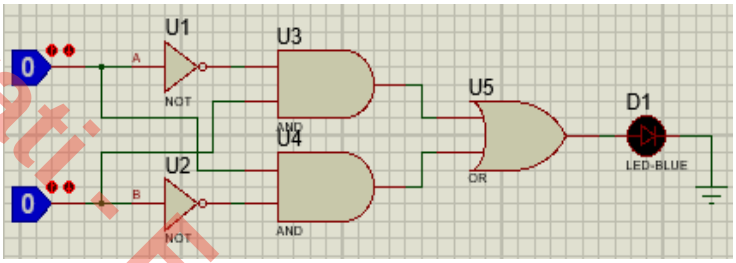
Perbedaan kedua jenis rangkaian ini terletak pada sifat keluarannya. Keluaran suatu rangkaian kombinasi setiap saat hanya ditentukan oleh masukan yang diberikan saat itu.

Contoh:

Pernulisan persamaan logika terhadap operasi yang dilakukan oleh gerbang terakhir akan menghasilkan persamaan logika dari rangkain tersebut.



Gambar 7. 1 Cara mendeskripsikan rangkaian 1 dengan persamaan logika



Gambar 7. 2 Cara mendeskripsikan rangkaian 2 dengan persamaan logika

Tabel 7. 1 Tabel Kebenaran rangkaian 1

Desimal	Input			Output Y
	A	B	C	
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Tabel 7. 2 Tabel Kebenaran rangkaian 2

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

B. Teorema-Teorema Aljabar Boole

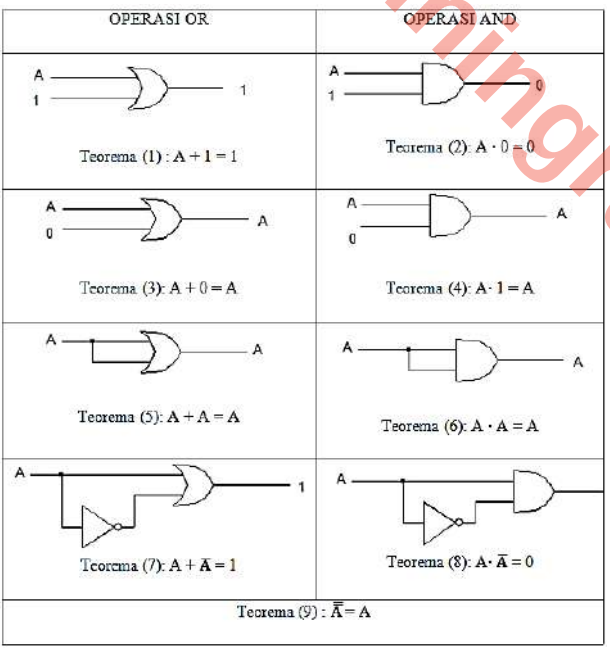
- Aljabar *boole* adalah suatu teknik matematika yang dipakai untuk menyelesaikan masalah-masalah logika.
- Aljabar *boole* mendasari operasi-operasi aritmatika yang dilakukan oleh komputer dan juga bermanfaat untuk

menganalisis dan mendesain rangkaian yang menjadi dasar bagi pembentukkan komputer sendiri

- Aljabar Boolean merupakan cara yang ekonomis untuk menjelaskan fungsi rangkaian digital, bila fungsi yang diinginkan telah diketahui, maka aljabar boolean dapat digunakan untuk membuat implementasi fungsi tersebut dengan cara yang lebih sederhana.
- Terdapat dua jenis teorema dalam aljabar Boole yakni teorema variabel tunggal dan teorema variabel jamak. Setiap teorema baik yang bersifat tunggal maupun jamak selalu memiliki teorema rangkapnya.

1. Teorema Variabel Tunggal

Teorema variabel tunggal aljabar Boole diturunkan dari operasi logika dasar OR, AND, dan NOT. Penurunan teorema variable tunggal ditunjukkan pada gambar dibawah ini.



Gambar 7. 3 Teorema-teorema aljabar Boole untuk variable tunggal

Tabel 7. 3 Teorema-Teorema Boolean dengan satu variable

Teorema	A + 0 = A	A + 0 = A
Satu dan Nol	A + 0 = A	A · 0 = 0
Identitas	A + 1 = 1	A · 1 = A
Idempoten	A + A = A	A · A = A
Komplemen	A + $\bar{A}$ = 1	A · $\bar{A}$ = 0
Involusi	$\bar{\bar{A}}$ = A	

2. Teorema Variabel Jamak

Teorema-teorema variabel jamak aljabar Boole umumnya sama dengan teorema-teorema pada aljabar biasa. Seperti pada aljabar biasa, pada aljabar Boole juga terdapat teorema komulatif, asosiatif, dan distributive. Tabel 7.4, menunjukkan teorema-teorema variable jamak yang berlaku pada aljabar Boole.

Tabel 7. 4 Teorema-teorema aljabar Boole untuk variable jamak

Teorema	Ekspresi	Sifat Rangkap
Komutatif	Teorema (10): A+B = B+A	Teorema (11): AB = BA
Asosiatif	Teorema (12): A+(B+C) = (A+B)+C	Teorema (13): A(BC) = (AB)C
Distributif	Teorema (14): A+BC = (A+B)(A+C)	Teorema (15): A(B+C)=AB+AC
Absorpsi	Teorema (16): A+AB = A Teorema (18): $A + \underline{A}.B = A + B$	Teorema (17): A(A+B) =A Teorema (19): $A(\underline{A} + B) = AB$
De Morgan	Teorema (20): $\underline{A + B} = \underline{A}.\underline{B}$	Teorema (21): $\underline{A.B} = \underline{A} + \underline{B}$

C. Pembuktian teorema Boole secara matematis

Bukti Bukti Teorema(14):
Buktikan: A+BC = (A+B) (A+C)
Bukti: = (A+B)(A+C) = AA + AC + AB +BC
= A+AC+AB+BC ingat teorema (6): AA = A
= (A+AC+AB) + BC
= A. (1+C+B) + BC
= A. 1 + BC ingat teorema (1): A+1 =1
Jadi: A+BC = (A+B) (A+C) **terbukti!**

Bukti Teorema(16):
Buktikan: A+AB = A
Bukti: = A+AB = A (1 + B)
= A. 1 ingat teorema (1): A+1 = 1
Jadi: A+AB = A **terbukti!**

Bukti Teorema(17):
Buktikan: A (A+B) = A
Bukti: = A (A+B) = AA + AB
= A+AB ingat teorema (6): AA = A
= A . (1+B) ingat teorema (1): A+1 = 1 Jadi: A(A+B)
= A **terbukti!**

Bukti Teorema(18):
Buktikan: A + $\bar{A}$B = A + B
Bukti: = A + $\bar{A}$ B = (A+$\bar{A}$) (A+ B)
ingat teorema (14): A+BC = (A+B)(A+C)
= 1 (A+B) ingat teorema (7): A + $\bar{A}$= 1 Jadi: A + $\bar{A}$B
= A + B **terbukti!**

Bukti Teorema(19):
Buktikan: A ($\bar{A}$+B) = A B
Bukti: = A ($\bar{A}$+B) = ($A\bar{A}$) + (A.B)
= 0 + (AB) ingat Teorema (8): A.$\bar{A}$= 0 Jadi: A + $\bar{A}$B
= A + B **terbukti!**

D. Gerbang NOR dan NAND

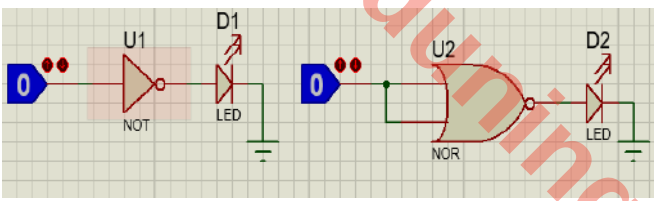
Gerbang NOR dan NAND memiliki sifat universal, artinya semua gerbang logika atau rangkaian logika dapat disusun dengan menggunakan gerbang NOR saja atau NAND saja.

Berikut ini akan ditunjukkan dengan bantuan aljabar Boole bahwa semua gerbang logika dapat disusun hanya dengan gerbang NOR saja atau NAND saja.

1. Gerbang NOT

a. Gerbang NOT dengan NOR

Gerbang NOT dapat dibentuk menggunakan gerbang NOR dengan cara berikut:



Gambar 7. 4 Gerbang NOT (a) Gerbang NOT dengan NOR (b)

Tabel 7. 5 Tabel Kebenaran Gerbang NOT dengan NOR

Gerbang NOT		Gerbang NOT dengan NOR	
Input A	Output Y	Input A	Output Y
0	1	0	1
1	0	1	0

Persamaan logika gerbang NOT pada gambar 28 (a) adalah $Y = \overline{A}$, dan NOT dengan NOR pada gambar 28 (b) adalah:

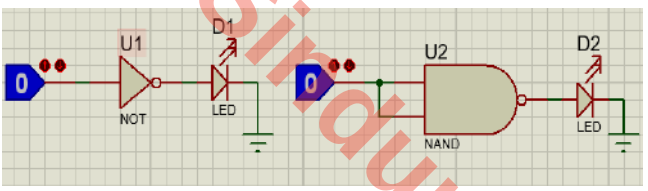
$Y = \underline{A}$

$Y =$ □ ingat teorema De Morgan!
Y = $\bar{A}$ □ inget teorema (6): A.A = A

Jadi, dengan menggunkan teorema de Morgan, dapat ditunjukkan bahwa gerbang NOR yang semua inputnya dijadikan satu memiliki watak yang sama dengan gerbang NOT

b. Gerbang NOT dengan NAND

Pembentukan gerbang NOT dengan gerbang NAND dapat dilakukan dengan cara berikut:



Gambar 7. 5 Gerbang NOT (a) Gerbang NOT dengan NAND (b)

Tabel 7. 6 Tabel Kebenaran Gerbang NOT dengan NAND

Gerbang NOT		Gerbang NOT dengan NAND	
Input A	Output Y	Input A	Output Y
0	1	0	1
1	0	1	0

Persamaan logika gerbang NOT pada gambar 82 (a) adalah Y = $\bar{A}$, dan NOT dengan NAND pada gambar 82 (b) adalah:

$Y = \underline{AA}$

$Y = \underline{A} + \underline{A}$ → ingat teorema De Morgan!
$Y = \underline{A}$ → inget teorema (5) : A + A = A

Jadi, dengan menggunkan teorema de Morgan, dapat ditunjukkan bahwa gerbang NAND yang semua inputnya dijadikan satu memiliki watak yang sama dengan gerbang NOT.

2. Gerbang OR

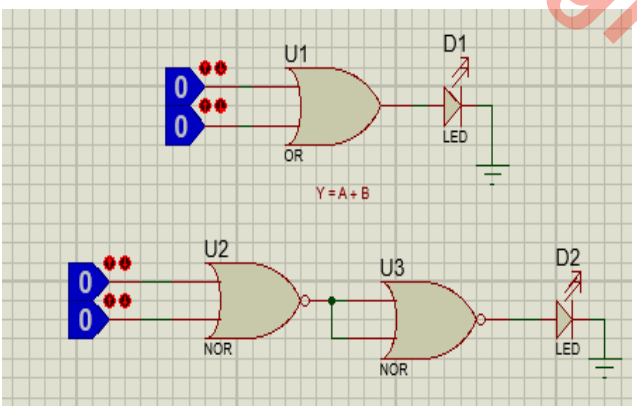
a. Gerbang OR dengan NOR

Persamaan logika gerbang OR adalah Y = A+B, dan dengan memberlakukan operasi komplemen ganda terhadap fungsi OR maka dapat diperoleh bentuk NOR dari gerbang OR, dapat dilakukan dengan cara berikut:

$Y = A + B$

$Y = \underline{\underline{A + B}}$ → operasi komplemen ganda

Persamaan(12) adalah bentuk NOR dari gerbang OR sehingga berdasarkan persamaan(12) dapat diperoleh rangkaian OR menggunakan gerbangNOR seperti gambar 83.



Gambar 7. 6 Gerbang OR (a) Gerbang OR dengan NOR (b)

Tabel 7. 7 Tabel Kebenaran Gerbang OR dengan NOR

Gerbang OR			Gerbang OR dengan NOR		
Input A	Input B	Output Y	Input A	Input B	Output Y
0	0	0	0	0	0
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	1

b. Gerbang OR dengan NAND

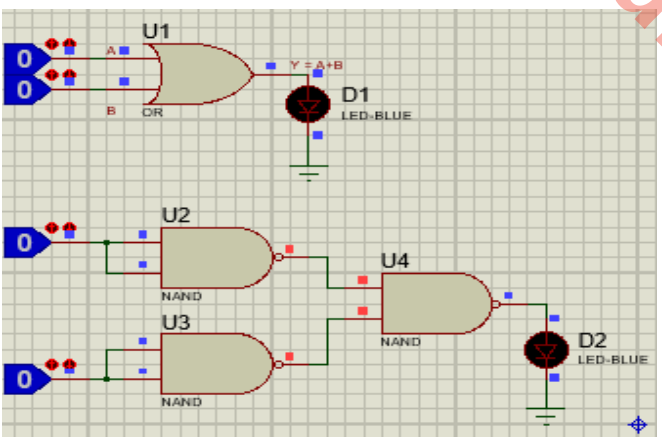
Bentuk NAND untuk gerbang OR dapat diperoleh dengan cara memberlakukan operasi komplemen ganda dan teorema de Morgan terhadap fungsi OR sebagai berikut:

$Y = A + B$

$Y = \underline{\underline{A}} + \underline{\underline{B}} \rightarrow$ Operasi komplemen ganda

$Y = \underline{\underline{AB}} \rightarrow$ Operasi Teorema De Morgan

Persamaan (13) merupakan bentuk NAND dari gerbang OR, sehingga dapat diperoleh rangkaian OR menggunakan gerbang NAND seperti gambar dibawah



Gambar 7. 7 Gerbang OR (a) Gerbang OR dengan NAND (b)

Tabel 7. 8 Tabel Kebenaran Gerbang OR dengan NAND

Gerbang OR			Gerbang OR dengan NAND		
Input A	Input B	Output Y	Input A	Input B	Output Y
0	0	0	0	0	0
0	1	1	0	1	1
1	0	1	1	0	1
1	1	1	1	1	1

3. Gerbang AND

a. Gerbang AND dengan NOR

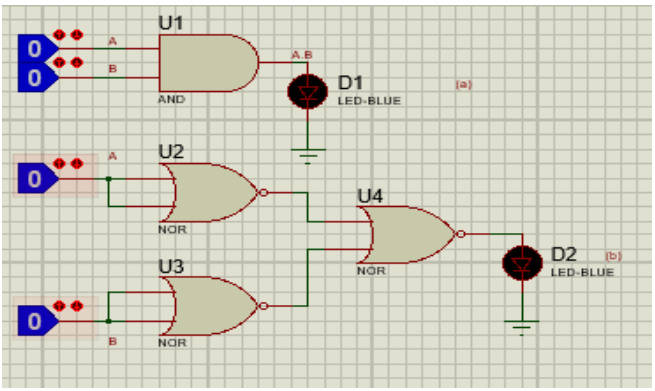
Bentuk NOR untuk gerbang AND dapat diperoleh dengan cara memberlakukan operasi komplemen ganda dan teorema de Morgan terhadap fungsi AND sebagai berikut:

$Y = A.B$

$Y = \overline{\overline{AB}}$ → Operasi Komplemen Ganda

$Y = \bar{A}\bar{B}$ → Operasi Teorema De Morgan

Persamaan (14) merupakan bentuk NOR dari gerbang AND, sehingga dapat diperoleh rangkaian OR menggunakan gerbang NAND seperti gambar dibawah ini



Gambar 7. 8 Gerbang AND (a) Gerbang AND dengan NOR (b)

Tabel 7. 9 Tabel Kebenaran Gerbang AND dengan NOR

Gerbang AND			Gerbang AND dengan NOR		
Input A	Input B	Output Y	Input A	Input B	Output Y
0	0	0	0	0	0
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	1	1	1

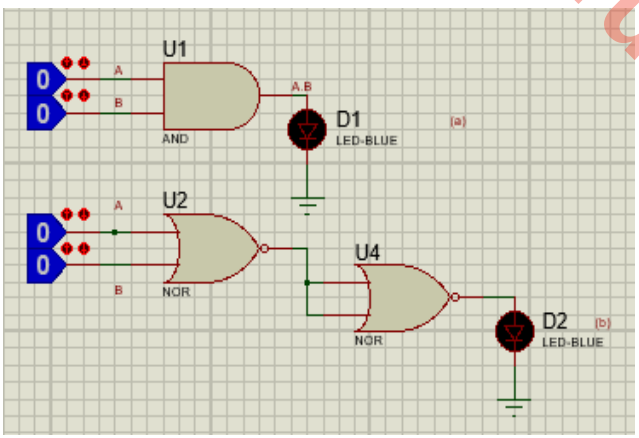
b. Gerbang AND dengan NAND

Dengan memberlakukan operasi komplemen ganda terhadap fungsi AND maka dapat diperoleh bentuk NAND dari gerbang AND sebagai berikut:

$Y = A.B$

$Y = \overline{\overline{AB}}$ Operasi komplemen ganda

Persamaan (15) merupakan bentuk NAND dari gerbang AND, sehingga dapat diperoleh rangkaian seperti gambar dibawah ini.



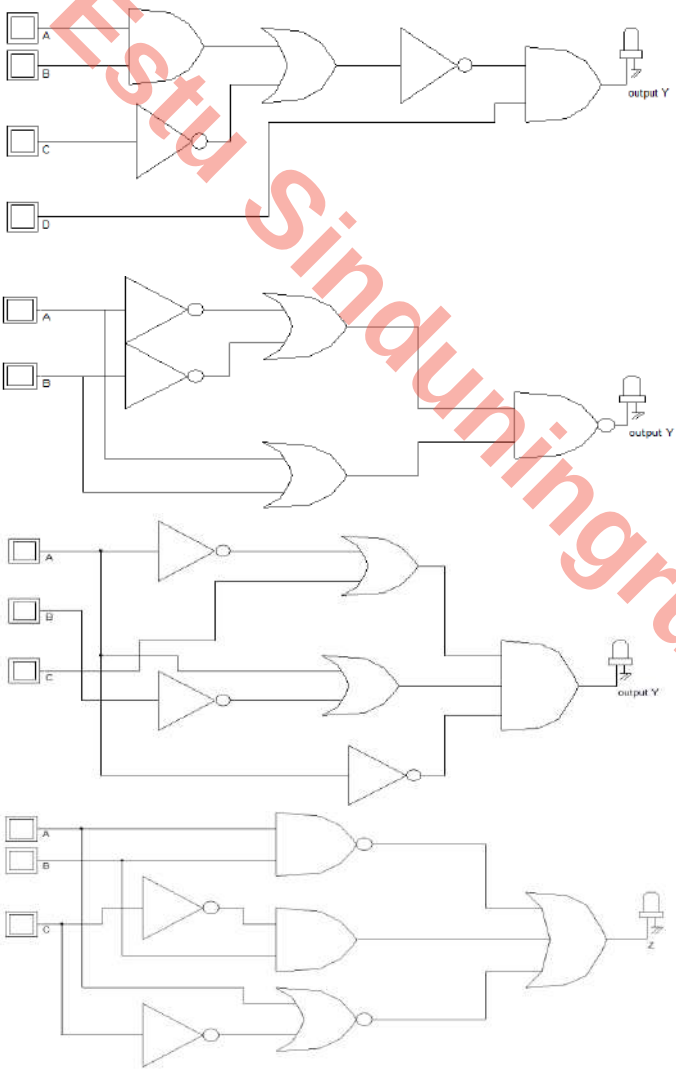
Gambar 7. 9 Gerbang AND (a) Gerbang AND (b) Gerbang NAND

Tabel 7. 10 Tabel Kebenaran Gerbang AND dengan NAND

Gerbang AND			Gerbang AND dengan NAND		
Input A	Input B	Output Y	Input A	Input B	Output Y
0	0	0	0	0	0
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	1	1	1

SOAL – SOAL LATIHAN

1. Coba deskripsikan rangkaian logika berikut ini ke dalam persamaan logika, dan buatlah tabel kebenarannya, serta anggap A adalah MSB!



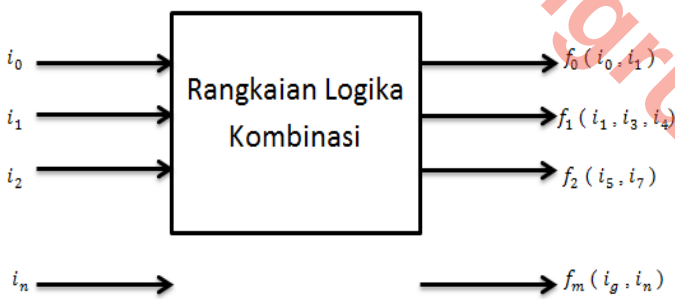
2. Buatlah Pengujian dengan menggunakan tabel kebenaran dan juga dengan menggunakan simulator DSCH2, terhadap teorema-teorema yang lain tidak hanya teorema De-Morgan saja, yaitu:
   a. Teorema (12): A+(B+C) = (A+B)+C
   b. Teorema (13): A(BC) = (AB)C
   c. Teorema (14): A+BC = (A+B)(A+C)
   d. Teorema (15): A(B+C)=AB+AC
   e. Teorema (16): A+AB = A
   f. Teorema (17): A(A+B) =A

BAB VIII
RANGKAIAN LOGIKA KOMBINASI

A. Pengertian Logika Kombinasi

Rangkaian Gerbang Kombinasi adalah :

- Keluaran suatu rangkaian kombinasi setiap saat hanya ditentukan oleh masukan yang diberikan saat itu, atau tidak tergantung pada keadaan output sebelumnya yang tidak tergantung pada waktu.
- Pada Gambar 8.1 diperlihatkan output rangkaian logika kombinasi merupakan fungsi langsung dari input-inputnya. Output f_0 merupakan fungsi dari input i_0 dan i_1 , dalam hal ini f_0 hanya dipengaruhi oleh kombinasi dari i_0 dan i_1 saja. Demikian pula f1 yang merupakan fungsi dari input i_0 ,i_3 dan i_4 keadaan level logikanya hanya tergantung pada ketiga input tersebut



Gambar 8. 1 Diagram blok rangkaian logika kombinasi

B. Bentuk-bentuk Persamaan Logika

1. Bentuk Sum Of Product (SOP)

SOP merupakan persamaan logika yang mengekspresikan operasi OR dan suku-suku berbentuk operasi AND. Secara lebih sederhana dapat dikatakan

bahwa SOP adalah bentuk persamaan yang melakukan operasi OR terhadap AND. SOP bisa disebut juga sebagai gerbang logika Mintrem.

$R = (\underline{A}BC) + (A\underline{B}C) + (AB\underline{C}) + (ABC)$
$S = (\underline{A}B) + (A\underline{B})$
$V = (\underline{A}BC) + (\underline{A}B\underline{C}) + (\underline{A}C) + \underline{A}$
$W = (\underline{A}C) + (B\underline{C}) + (\underline{A}C)$

Tabel 8. 1 Tabel kebenaran fungsi $R = (\underline{A}BC) + (A\underline{B}C) + (AB\underline{C}) + (ABC)$

INPUT			OUTPUT	
A	B	C	R	
0	0	0	0	
0	0	1	1	$m_1 = \underline{A}BC$
0	1	0	0	
0	1	1	0	
1	0	0	1	$m_4 = A\underline{B}C$
1	0	1	0	
1	1	0	1	$m_6 = AB\underline{C}$
1	1	1	1	$m_7 = ABC$

$R = (\underline{A}BC) + (A\underline{B}C) + (AB\underline{C}) + (ABC)$
Misalnya

a. A=0, B=0, C=1
   $R = (\underline{A}BC) + (A\underline{B}C) + (AB\underline{C}) + (ABC)$
   $R = (\underline{0}01) + (0\underline{0}1) + (00\underline{1}) + (001)$
   $R = (111) + (010) + (000) + (001)$
   $R = 1 + 0 + 0 + 0$
   $Maka\ R = 1$
b. A=1, B=0, C=0
   $R = (\underline{A}BC) + (A\underline{B}C) + (AB\underline{C}) + (ABC)$
   $R = (\underline{1}00) + (1\underline{0}0) + (10\underline{0}) + (100)$
   $R = (010) + (111) + (101) + (100)$
   $R = (0) + (1) + (0) + (0)$
   $Maka\ R = 1$

c. A=1, B=1, C=0

$R = (\underline{A}BC) + (\underline{A}B\underline{C}) + (AB\underline{C}) + (ABC)$

$R = (\underline{1}\underline{1}0) + (\underline{1}1\underline{0}) + (11\underline{0}) + (110)$

$R = (000) + (101) + (111) + (110)$

$R = (0) + (0) + (1) + (0)$

$Maka\ R = 1$

d. A=1, B=1, C=1

$R = (\underline{A}BC) + (\underline{A}B\underline{C}) + (AB\underline{C}) + (ABC)$

$R = (\underline{1}\underline{1}\underline{1}) + (\underline{1}\underline{1}\underline{1}) + (11\underline{1}) + (111)$

$R = (001) + (100) + (110) + (111)$

$R = (0) + (0) + (0) + (1)$

$Maka\ R = 1$

Penulisan variabel gerbang logika MINTERM, yaitu :

Cara 1 $R = (\underline{A}BC) + (\underline{A}B\underline{C}) + (AB\underline{C}) + (ABC)$

Cara 2 $R = (A, B, C) = \sum M(1,4,6,7)$

2. Bentuk Product Of Sum (POS)

POS merupakan persamaan logika yang mengekspresikan operasi AND dan suku-suku berbentuk operasi OR, atau dengan kata lain POS adalah bentuk persamaan yang melakukan operasi AND terhadap OR. POS bisa disebut juga sebagai gerbang logika Maxtrem.

Contoh :

$R = (\underline{A} + \underline{B} + C) + (A + \underline{B} + \underline{C}) + (A + B + \underline{C}) + (A + B + C)$

$S = (\underline{A} + \underline{B}) + (A + \underline{B})$

$V = (\underline{A} + \underline{B} + C) + (\underline{A} + B + \underline{C}) + (\underline{A} + \underline{C}) + \underline{A}$

$W = (\underline{A} + C) + (B + \underline{C}) + (\underline{A} + C)$

Misalnya

a. A=0, B=0, C=1

$R = (\underline{A} + \underline{B} + C) + (A + \underline{B} + \underline{C}) + (A + B + \underline{C}) + (A + B + C)$

$R = (\underline{0} + \underline{0} + 1) + (0 + \underline{0} + \underline{1}) + (0 + 0 + \underline{1}) + (0 + 0 + 1)$

$R = (1 + 1 + 1) + (0 + 1 + 0) + (0 + 0 + 0) + (0 + 0 + 1)$

$R = (0) + (1) + (0) + (1)$

$Maka\ R = 0$

b. A=1, B=0, C=0

$R = (\underline{A} + \underline{B} + C) + (A + \underline{B} + \underline{C}) + (A + B + \underline{C}) + (A + B + C)$

$R = (\underline{1} + \underline{0} + 0) + (1 + \underline{0} + \underline{0}) + (1 + 0 + \underline{0}) + (1 + 0 + 0)$

$R = (0 + 1 + 1) + (0 + 0 + 1) + (0 + 0 + 0) + (1 + 0 + 0)$

$R = (0) + (1) + (0) + (1)$

$Maka\ R = 1$

c. A=1, B=1, C=0

$R = (\underline{A} + \underline{B} + C) + (A + \underline{B} + \underline{C}) + (A + B + \underline{C}) + (A + B + C)$

$R = (\underline{1} + \underline{1} + 0) + (1 + \underline{1} + \underline{0}) + (1 + 1 + \underline{0}) + (1 + 1 + 0)$

$R = (1 + 0 + 1) + (0 + 1 + 0) + (1 + 1 + 1) + (1 + 1 + 0)$

$R = (0) + (1) + (1) + (0)$

$Maka\ R = 0$

d. A=1, B=1, C=1

$R = (\underline{A} + \underline{B} + C) + (A + \underline{B} + \underline{C}) + (A + B + \underline{C}) + (A + B + C)$

$R = (\underline{1} + \underline{1} + 1) + (1 + \underline{1} + \underline{1}) + (1 + 1 + \underline{1}) + (1 + 1 + 1)$

$R = (0 + 0 + 1) + (1 + 0 + 0) + (1 + 1 + 0) + (1 + 1 + 1)$

$R = (1) + (1) + (0) + (0)$

$Maka\ R = 0$

Tabel 8. 2 Tabel kebenaran fungsi $\boldsymbol{R = (\underline{A} + \underline{B} + C) + (A + \underline{B} + \underline{C}) + (A + B + \underline{C}) + (A + B + C)}$

INPUT			OUTPUT	
A	B	C	R	
0	0	0	1	
0	0	1	0	$m_1 = A + B + \underline{C}$
0	1	0	0	$m_2 = A + \underline{B} + C$
0	1	1	1	
1	0	0	0	$m_4 = \underline{A} + B + C$
1	0	1	1	
1	1	0	0	$m_6 = \underline{A} + \underline{B} + C$
1	1	1	1	

Penulisan variabel gerbang logika MAXTERM, yaitu:

Cara 1 $R = (\underline{A} + \underline{B} + C) + (A + \underline{B} + \underline{C}) + (A + B + \underline{C}) + (A + B + C)$

Cara 2 $R = (A, B, C) = \prod M\ (1, 2, 4, 6)$

C. Mengubah Fungsi Bentuk Tak Standar Menjadi Bentuk Standar

Jika suatu persamaan logika bentuk SOP Tak Standar ingin diubah menjadi bentuk SOP Standar, maka dapat dilakukan dengan cara seperti contoh berikut ini:

Ubahlah fungsi:

1. $Y = A\underline{BC} + \underline{B}C$
2. $Y = AC + \underline{B}$

Menjadi bentuk standar.

Jawab:

1. $Y = A\underline{BC} + \underline{B}C$

   $Y = A\underline{BC} + \underline{B}C(A + \underline{A}), ingat \ldots . (A + \underline{A}) = 1$

   $Y = A\underline{BC} + A\underline{B}C + \underline{AB}C$

   $Y = (A, B, C) = \sum \quad M = \ (1,4,5)$

2. $Y = AC + \underline{B}$

   $Y = AC(B + \underline{B}) + \underline{B}(A + \underline{A})(C + \underline{C})$

   $Y = ABC + A\underline{B}C + \underline{B}(A + \underline{A})(C + \underline{C})$

   $Y = ABC + A\underline{B}C + A\underline{B} + \underline{AB}(C + \underline{C})$

   $Y = ABC + \frac{A\underline{B}C + A\underline{B}C}{A\underline{B}C} + A\underline{B}C + \underline{AB}C + \underline{AB}C$

   $Y = ABC + A\underline{B}C + A\underline{B}C + \underline{AB}C + \underline{AB}C$

   $Y = (A, B, C) = \sum \quad m = (0,2,3,6,7)$

D. Memperoleh Persamaan Bentuk Standar dari Tabel Kebenaran

Dengan mengingat bahwa nilai fungsi SOP standar selalu bernilal 1 dan nilai fungsi POS standar bernilai 0 untuk input yang memiliki nilai sesuai dengan nama-nama minterm dan maxterm penyusunnya, dan suatu label kebenaran yang diketahui dapat diperoleh persamaan dalam bentuk standar.

Contoh: Tuliskan persamaan logika bentuk SOP Standar dan POS Standar yang diperoleh dan tabel 30, kebenaran berikut ini!

Tabel 8. 3 Tabel kebenaran yang akan ditentukan persamaan logikanya

INPUT		OUTPUT	
A	B	Y	
0	0	0	$m_0 = (A + B)$
0	1	1	$m_1 = (\underline{A} + B)$
1	0	1	$m_2 = (A + \underline{B})$
1	1	0	$m_3 = (\underline{A} + \underline{B})$

Dari tabel 23, persamaan logika yang diperoleh adalah:

1. Bentuk SOP standar :

   $Y = m_1 + m_2 = (\underline{A} + B) + (A + \underline{B})$

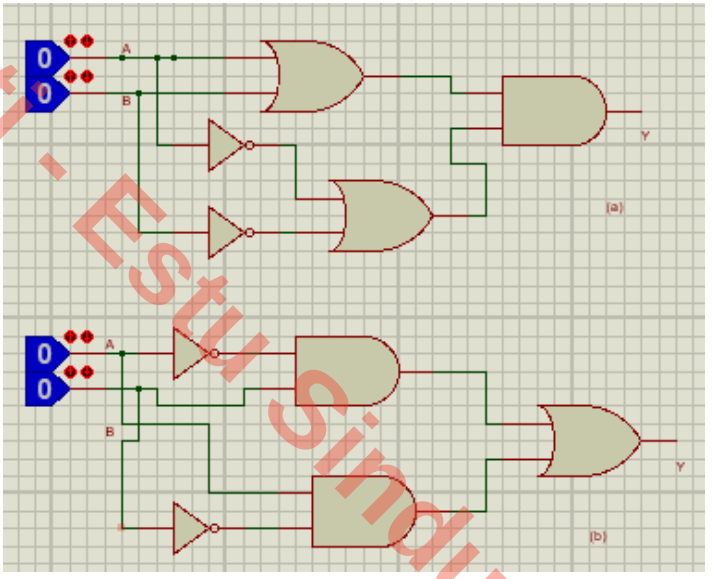
   $Y = A, B = \sum \; m\,(1,2)$

2. Bentuk POS standar:

   $Y = m_0 + m_3 = (A + B) + (\underline{A} + \underline{B})$

   $Y = A, B = \prod \; m(0,3)$

Rangkaian untuk kedua bentuk tersebut adalah:



Gambar 8. 2 Rangkaian untuk (a) Y(A, B) = $\sum m$ (1, 2) dan (b) Y(A, B) = $\prod$ M (0, 3)

E. Penyederhanaan Secara Aljabar

Penyederhanaan secara aljabar atau bisa dikatakan juga penyederhanaan secara matematis.

Contoh Soal :

1. Sederhanakan persamaan $Z = A\underline{B}D + A\underline{B\underline{D}}$

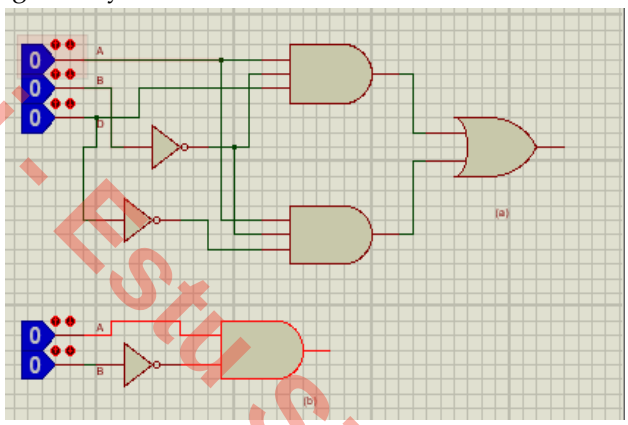
   Jawab:

   $Z = A\underline{B}D + A\underline{B\underline{D}}$

   $Z = A\underline{B}(D + \underline{D})$

   $Z = A\underline{B}, ingat \ldots\ldots D + \underline{D} = 1$

Rangkaiannya adalah



Gambar 8. 3 (a) Rangkaian Z = AB̄D + AB̄D̄ yang belum disederhanakan, (b) Rangkaian yang telah disederhanakan

Sederhanakan persamaan berikut :

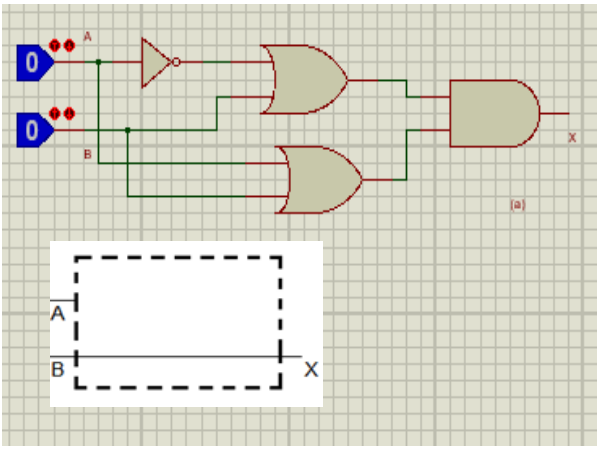
$X = (\underline{A} + B)(A + B)$

$X = (\underline{A}A + \underline{A}B + AB + BB)$

$X = (0 + \underline{A}B + AB + B, ingat \ldots\ldots\ldots A\underline{A} = 0, BB = 1$

$X = B(\underline{A} + A + 1), ingat \ldots\ldots\ldots \underline{A} + A + 1 = 1$

Rangkaiannya adalah :



Gambar 8. 4 (a) Rangkaian X = $\overline{(A}$ + B)(A + B) yang belum disederhanakan, (b) rangkaian yang telah disederhanakan

Contoh Soal Lain :

F_1 = (A′ + B) . A F_2 = A + (A′. B)
= A′. A + A . B = (A + B). (A′ + B)
= 0 + A . B = A + B
= A . B

F_3 = (A + B′) . (A + B) F_4 = A . B + A′. C + B . C
= A. A + A. B + A. B′ + B. B′ = AB + A′C + BC (A + A′)
= A + A. B + A. B′ + 0 = AB + A′C + ABC + A′BC
= A (1 + B + B′) = AB (1 + C) + A′C (1 + B)
= A (1) = AB + A′C
F_3 = (A + B′) . (A + B) = A F_4 = A . B + A′. C + B. C = AB + A′C

F_5 = (A + B) (A + C) $F_6 = \underline{AB} + \underline{CD} + \underline{AC}\underline{D}$
= A.A + A.C + A.B + B.C = $AB\underline{CD} + \underline{A} + \underline{C} + D$
= A + AC + AB + BC = = $AB\left(\underline{C} + \underline{D}\right) + \underline{A} + \underline{C} + D$
= A (1 + C) + AB + BC = $\underline{C} + B\left(A + \underline{D}\right) + (\underline{A} + D)$
= A + AB + BC = $\bar{C}$ + B
= A.(1+B) + BC $F_6 = \underline{AB} + \underline{CD} + \underline{AC}\underline{D} = \underline{C} + B$
= A + BC

F. Peta karnaugh

- Selain dengan teorema boole, salah satu cara untuk memanipulasi dan menyederhanakan fungsi boole adalah dengan teknik peta karnaugh.
- Peta karnaugh merupakan sekumpulan kotak-kotak yang diberi nama sedemikian rupa berdasarkan nama variabelnya dan diletakkan sedemikian rupa pula sehingga dapat mengeliminasi beberapa tabel jika kotak itu digabung. Jumlah kotak tergantung banyaknya variabel input. Jika ada sebanyak n input maka ada 2n kombinasi input, maka sebanyak itu pula kotak yang dibutuhkan.
- Dalam peta karnaugh dikenal istilah tetangga dekat. Yang dimaksud dengan tetangga dekat adalah kotak-kotak yang memiliki satu atau lebih variabel yang sama atau kotak-

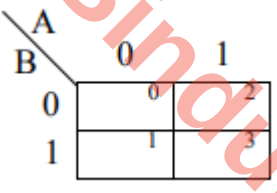
kotak yang terletak dalam satu atau lebih bidang yang sama.

- Yang dimaksud dengan bidang adalah sekumpulan kotak yang sudah diberi nama berdasarkan variabel inputnya.

1. Peta Karnaugh untuk 2 Variabel (A, B)

   Untuk 2 variabel input akan ada sebanyak $2^2 = 4$ kombinasi input

   - Maka banyaknya kotak yang dibutuhkan adalah 4 kotak.
   - Keempat kotak itu diatur sebagai berikut :

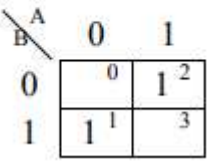


Penggabungan kotak-kotak untuk 2 variabel (A, B)

- Jika ada 2 kotak yang ditandai 1 bertetangga dekat dapat digabung, akan menyatakan 1 variabel tunggal.
- Untuk 1 kotak yang ditandai 1 dan tidak memiliki tetangga dekat, akan menyatakan 2 variabel.

Contoh :

$$y = \bar{A}\,B + A\,\bar{B}$$



Menyatakan 1 tetangga

$$y = \bar{A}\,B + A\,B$$

B \ A	0	1
0		
1	1	1

Menyatakan 2 tetangga sehingga dapat disederhanakan menjadi y = B

$$y = \bar{A}\,\bar{B} + A\,\bar{B} + A\,B$$

B \ A	0	1
0	1	1
1		1

Menyatakan 2 buah 2 tetangga sehingga dapat disederhanakan menjadI $Y = A + \underline{B}$

2. Peta Karnaugh untuk 3 Variabel (A, B, C)

   Untuk 3 variabel input akan ada sebanyak $2^3 = 8$ kombinasi input

   - Maka banyaknya kotak yang dibutuhkan adalah 8 kotak.
   - Kedelapan kotak itu diatur (ada 2 cara) sebagai berikut :

C \ AB	00	01	11	10
0	0	2	6	4
1	1	3	7	5

BC \ A	0	1
00	0	4
01	1	5
11	3	7
10	2	6

Penggabungan kotak-kotak untuk 3 variabel (A, B, C)

- 4 kotak yang bertetangga dekat dapat digabung dan menyatakan 1 variabel unggal.
- 2 kotak yang bertetangga dekat dapat digabung dan menyatakan 2 variabel.
- 1 kotak yang tidak bertetangga dekat akan menyatakan 3 variabel

Contoh:

$Y = AB\underline{C} + \underline{A}BC + ABC + A\underline{B}C$

C \ AB	00	01	11	10
0			1	
1		1	1	1

Menyatakan 3 buah 2 tetangga sehingga dapat disederhanakan menjadi y = AB + BC + AC

$Y = \underline{AB}C + \underline{A}BC + ABC + A\underline{B}C$

C \ AB	00	01	11	10
0				
1	1	1	1	1

Menyatakan 4 tetangga sehingga dapat disederhanakan menjadi y = C

$Y = \underline{AB}\underline{C} + \underline{A}BC + AB\underline{C} + ABC$

C \ AB	00	01	11	10
0		1	1	
1		1	1	

Menyatakan 4 tetangga sehingga dapat disederhanakan menjadi y = B

3. Peta Karnaugh untuk 4 Variabel (A, B, C, D)

Untuk 4 variabel input akan ada sebanyak 24 = 16 kombinasi input

- Maka banyaknya kotak yang dibutuhkan adalah 16 kotak.
- Keenambelas kotak itu diatur sebagai berikut :

CD \ AB	00	01	11	10
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

Penggabungan kotak-kotak untuk 4 variabel (A, B, C, D)

- 8 kotak yang bertetangga dekat dapat digabung dan menyatakan 1 variabel unggal.
- 4 kotak yang bertetangga dekat dapat digabung dan menyatakan 2 variabel tunggal.
- 2 kotak yang bertetangga dekat dapat digabung dan menyatakan 3 variabel.
- 1 kotak yang tidak bertetangga dekat akan menyatakan 4 variabel

Contoh:

$Y = ABCD + ABC\underline{D} + A\underline{B}CD + A\underline{B}C\underline{D}$

CD \ AB	00	01	11	10
00				
01				
11			1	1
10			1	1

Menyatakan 4 tetangga sehingga dapat disederhanakan menjadi y = A C

$Y = \underline{ABCD} + A\underline{BCD} + \underline{ABC}\underline{D} + A\underline{B}C\underline{D}$

CD \ AB	00	01	11	10
00	1			1
01				
11				
10	1			1

Menyatakan 4 tetangga sehingga dapat disederhanakan menjadi $Y = \underline{BD}$

$Y = \underline{A}B + AB\underline{C} + ABCD + ABC\underline{D}$

CD \ AB	00	01	11	10
00		1	1	
01		1	1	
11		1	1	
10		1	1	

Menyatakan 8 tetangga sehingga dapat disederhanakan menjadi y = B

4. Peta Karnaugh untuk 5 Variabel (A, B, C, D, E)

   Untuk 5 variabel input akan ada sebanyak 2^5 = 32 kombinasi input

   - Maka banyaknya kotak yang dibutuhkan adalah 32 kotak.
   - Ketigapuluh dua kotak itu diatur sebagai berikut :

ABC / DE	000	001	011	010
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

100	101	111	110	ABC / DE
16	20	28	24	00
17	21	29	25	01
19	23	31	27	11
18	22	30	26	10

Penggabungan kotak-kotak untuk 5 variabel (A, B, C, D, E)

- 16 kotak yang bertetangga dekat dapat digabung dan menyatakan 1 variabel tunggal.
- 8 kotak yang bertetangga dekat dapat digabung dan menyatakan 2 variabel tunggal.
- 4 kotak yang bertetangga dekat dapat digabung dan menyatakan 3 variabel tunggal.
- 2 kotak yang bertetangga dekat dapat digabung dan menyatakan 4 variabel.
- 1 kotak yang tidak bertetangga dekat akan menyatakan 5 variabel

Contoh :

$Y = \underline{A}B\underline{D} + A\underline{B}\underline{D} + \underline{A}\underline{B}D + A\underline{B}D$

ABC / DE	000	001	011	010
00			1	1
01			1	1
11	1	1		
10	1	1		

100	101	111	110	ABC / DE
		1	1	00
		1	1	01
1	1			11
1	1			10

Menyatakan 2 buah 8 tetangga sehingga dapat disederhanakan menjadi $Y = B\underline{D} + \underline{B}D$

$$Y = \underline{A}BC + \underline{AB}\underline{C} + AB\underline{D} + ABD$$

ABC \ DE	000	001	011	010
00			1	1
01			1	1
11			1	1
10			1	1

100	101	111	110	ABC \ DE
		1	1	00
		1	1	01
		1	1	11
		1	1	10

Menyatakan 16 tetangga sehingga dapat disederhanakan menjadi y = B

5. Peta Karnaugh untuk 6 Variabel (A, B, C, D, E, F)

   Untuk 6 variabel input akan ada sebanyak 26 = 64 kombinasi input

   - Maka banyaknya kotak yang dibutuhkan adalah 64 kotak.
   - Keenampuluh empat kotak itu diatur sebagai berikut :

ABCD \ EF	0000	0001	0011	0010
00	0	4	12	8
01	1	5	13	9
11	3	7	15	11
10	2	6	14	10

0100	0101	0111	0110	ABCD \ EF
16	20	28	24	00
17	21	29	25	01
19	23	31	27	11
18	22	30	26	10

00	32	36	44	40
01	33	37	45	41
11	35	39	47	43
10	34	38	46	42
EF \ ABCD	1000	1001	1011	1010

00	48	52	60	56
01	49	53	61	57
11	51	55	63	59
10	50	54	62	58
EF \ ABCD	1100	1101	1111	1110

Penggabungan kotak-kotak untuk 6 variabel (A, B, C, D, E, F)

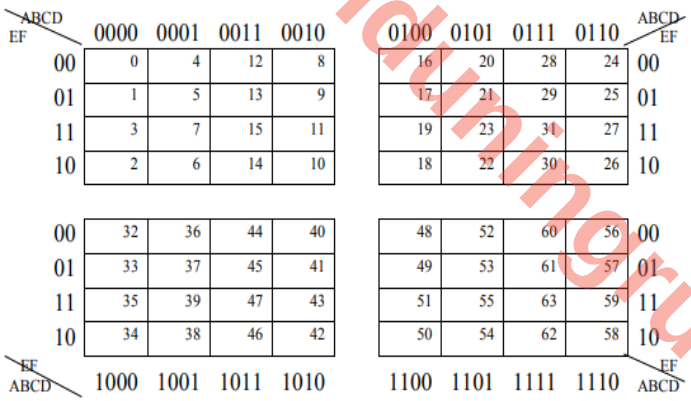
- 32 kotak yang bertetangga dekat dapat digabung dan menyatakan 1 variabel tunggal.
- 16 kotak yang bertetangga dekat dapat digabung dan menyatakan 2 variabel tunggal.

- 8 kotak yang bertetangga dekat dapat digabung dan menyatakan 3 variabel tunggal.
- 4 kotak yang bertetangga dekat dapat digabung dan menyatakan 4 variabel tunggal.
- 2 kotak yang bertetangga dekat dapat digabung dan menyatakan 5 variabel.
- 1 kotak yang tidak bertetangga dekat akan menyatakan 6 variabel

Contoh :

Sederhanakan fungsi boole berikut ini :

Y (A,B,C,D,E,F) = $\sum$ *M*(0,9,11,24,25,27,34,35,38,39,43,47, 51,55,58,59,62,63)



G. Kondisi Diabaikan (Don't Care Condition)

Rangkaian-rangkaian logika yang telah dipelajari sebelumnya hampir semuanya selalu memberikan output 1 atau 0 untuk suatu kombinasi input yang diberikan. Selain itu, terdapat pula rangkaian logika dengan beberapa kombinasi input yang dalam kenyataannya tidak pemah ada. Contoh untuk rangkaian-rangkaian logika dengan input kode BCD, inputnya hanya ada 10 kombinasi yakni 0000 sampai dengan 1001, karena kode BCD memang hanya merepresentasikan bilangan desimal yang memiliki nilai 0

sampai dengan 9.

Dengan kata lain, terdapat 6 buah kombinasi input yang tidak pemah ada yakni 1010 sampai dengan biner 1111. Untuk rangkaian yang input-inputnya tidak pemah ada, outputnya tidak dinyatakan dalam nilai 1 atau 0 melainkan diberi tanda X yang berarti keadaan diabaikan (*don't care condition*). Tabel 29 berikut ini menunjukkan watak rangkaian detektor bilangan prima dengan input kode BCD.

Tabel 8. 4 Contoh Tabel Kebenaran yang mengandung keadaan diabaikan:

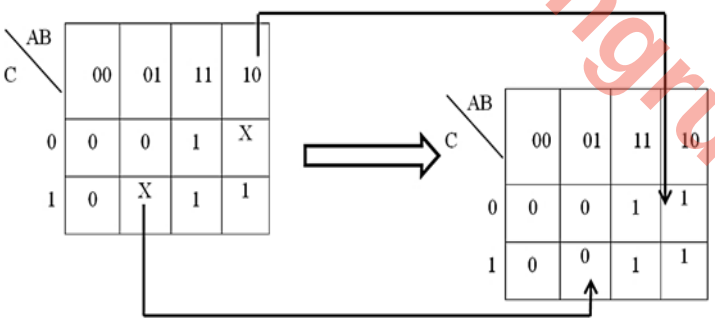
INPUT				OUTPUT
A	**B**	**C**	**D**	**Y**
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Contoh lain dari tabel kebenaran yang mengandung keadaan diabaikan ditunjukkan pada tabel 25 berikut ini

Tabel 8. 5 Contoh Tabel Kebenaran yang mengandung keadaan diabaikan:

INPUT			OUTPUT
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	X
1	0	0	X
1	0	1	1
1	1	0	1
1	1	1	1

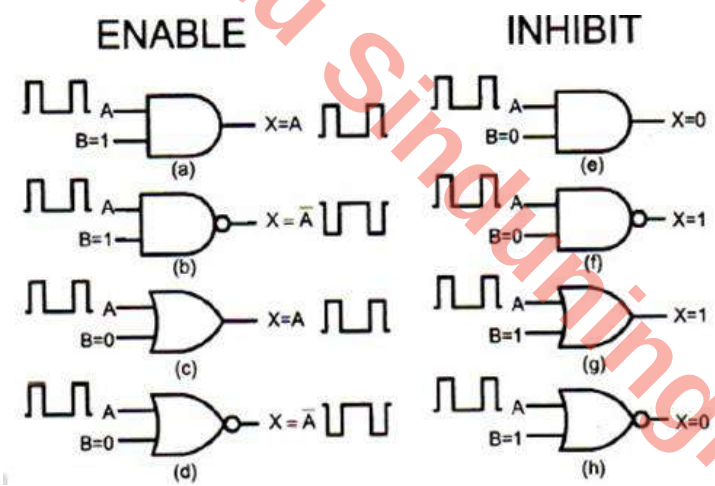
Peta Karnaugh untuk tabel 30 ditunjukkan pada gambar 83. Pada peta Karnaugh yang mengandung kondisi diabaikan, X dapat dipilih bemilai 0 atau 1. Pemilihan nilai X dilakukan sedemikian rupa sehingga dan peta Karnaugh tersebut dapat diperoleh persamaan yang paling sederhana.



Gambar 8. 5 Peta Karnaugh untuk tabel 25

H. Rangkaian Enable dan Inhibit

Rangkaian logika dalam implementasinya, terkadang perlu dilengkapi dengan rangkaian pengontrol yang berfungsi untuk meneruskan atau mengharnbat inputnya. Rangkaian enable adalah susunan gerbang yang memberikan fungsi meneruskan inputnya, dan rangkaian inhibit adalah susunan gerbang yang menyebabkan suatu input tidak dapat diteruskan. Gambar 37 berikut ini menunjukkan susunan gerbang yang memberikan fungsi enable dan inhibit.



Gambar 8. 6 Rangkaian enable dan inhibit

Penjelasan Gambar, yaitu:

1. Pada gambar 37 (a) dan (e) ditunjukkan gerbang AND ditunjukkan gerbang AND sebagai rangkaian enable dan inhibit. Input A akan diteruskan ke outputnya jika pengontrol B bernilai tinggi (B=1), dalam hal ini gerbang AND akan meneruskan inputya (enable) sehingga X=A. Namun, jika pengontrol B bemilai rendah (B=0) maka input akan dihambat (inhibit) dan untuk rangkaian ini outputnya bernilai rendah (X=0).
2. Rangkaian enable dan inhibit juga dapat diimplementasikan dengan gerbang NAND seperti

ditunjukkan pada gambar 37 (b) dan (f). Pada rangkaian ini jika nilai pengontrolnya menyebabkan enable (B=1). maka outputnya merupakan komplemen dan inputnya. Untuk pengontrol yang menyebabkan inhibit (B= 0), outputnya bemilai tinggi (X=1).Selain dengan gerbang AND dan NAND, rangkaian enable dan inhibit juga dapat diimplementasikan dengan gerbang OR dan NOR.

3. Perhatikan gambar 37(c) dan (g)! Pada gerbang OR, keadaan enable terjadi jika pengontrol B bemilai rendah (B=0), dan sebaliknya keadaan inhibit terjadi jika pengontrol B bemilai tinggi (B=1). Untuk keadaan inhibit, gerbang OR memberikan keadaan output tinggi (X=1). Jika rangkaian enable dan inhibit diimplementasikan dengan gerbang NOR, maka untuk pengontrol yang menyebabkan terjadinya keadaan enable, output gerbang merupakan komplemen dan inputnya. Untuk keadaan inhibit, rangkaian dengan NOR membenkan output bernilai rendah.

SOAL – SOAL LATIHAN

1. Jelaskan pengertian *sum of product*, *standard sum of product, product of sum, standard product of sum, minterm* dan *maxterm*! Benkan contoh dari masing-masing pengertian tersebut!

2. Susunlah tabel kebenaran dari persamaan berikut ini:
   a. $A(W_1, W_2, W_3, W_4) = \prod M(2, 3, 8, 9, 12)$
   b. $Z(Y_1, Y_2, Y_3) = \sum m(1, 6, 7)$
   c. $X(A, B, C) = \prod M(0, 3, 8, 7)$

3. Sederhanakan rangkaian logika dibawah ini dengan menggunakan K-Map dan Aljabar Boolean
   a. $E = \underline{A}BC + \underline{AB}C + \underline{A}B$
   b. $E = \underline{ABC} + \underline{ABC} + AB\underline{C} + A\underline{B}C$
   c. $E = \underline{ABC} + \underline{A}B\underline{C} + A\underline{BC} + A\underline{B}C$

4. Ubahlah bentuk persamaan logika berikut ini menjadi bentuk standart!
   a. X = A + BC + AB

5. Tuliskan persamaan logika bentuk SOP standar dan POS standar yang didapatkan dari tabel kebenaran dibawah ini:

Tabel SOP

INPUT		OUTPUT	
A	**B**	**C**	**Y**
0	0	0	**1**
0	0	1	**0**
0	1	0	**1**
0	1	1	**1**
1	0	0	**0**
1	0	1	**1**
1	1	0	**1**
1	1	1	**0**

Tabel POS

INPUT		OUTPUT	
A	**B**	**C**	**Y**
0	0	0	**0**
0	0	1	**1**
0	1	0	**1**
0	1	1	**0**
1	0	0	**0**
1	0	1	**1**
1	1	0	**0**
1	1	1	**1**

BAB IX
PRAKTEK RANGKAIAN LOGIKA KOMBINASI DENGAN PROTEUS

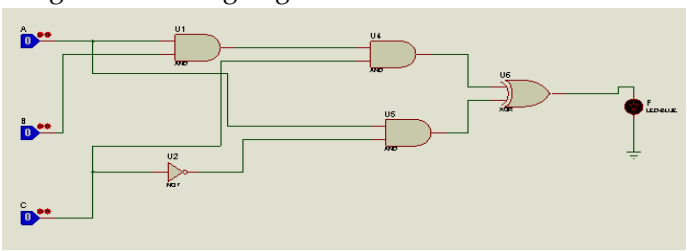
A. Percobaan Dengan Rangkaian Logika Dengan Proteus

Gerbang logika atau gerbang logik adalah suatu entitas dalam elektronika dan matematika Boolean yang mengubah satu atau beberapa masukan logik menjadi sebuah sinyal keluaran logik. Gerbang logika terutama diimplementasikan secara elektronis menggunakan diode atau transistor, akan tetapi dapat pula dibangun menggunakan susunan komponen-komponen yang memanfaatkan sifat-sifat elektromagnetik (relay), cairan, optik dan bahkan mekanik. Terdapat tiga gerbang logika dasar, yaitu : gerbang AND, gerbang OR, gerbang NOT. Ketiga gerbang ini menghasilkan empat gerbang berikutnya, yaitu : gerbang NAND, gerbang NOR, gerbang XNOR.

Gerbang Logika beroperasi berdasarkan sistem bilangan biner yaitu bilangan yang hanya memiliki 2 kode simbol yakni 0 dan 1 dengan menggunakan Teori Aljabar Boolean.

Rumus Gerbang logika Boolean : $F = (A + B)C + \underline{AC}$

1. Rangkaian Gerbang Logika



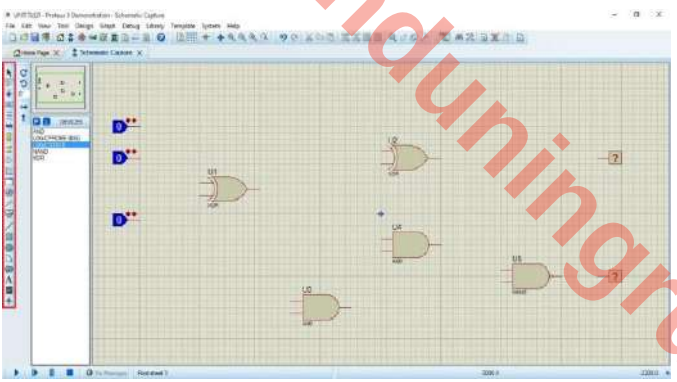
Gambar 9. 1 Rangkaian Gerbang Logika

Untuk membuat rangkaian gerbang logika diatas dibutuhkan komponen – komponen sbb :

- Symbol AND
- Symbol NOT
- Symbol XOR
- Lampu LED sebagai indicator

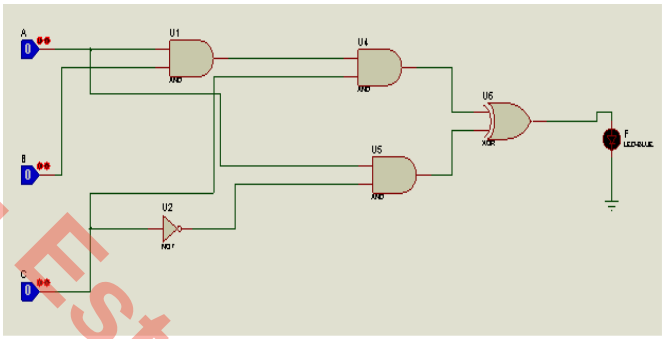
Susunlah komponen-komponen tersebut menjadi seperti rangkaian Gerbang Logika.

- Untuk menyambungkan atau membuat garis, kita bisa memilih "2D Graphics Line Mode" atau dengan mendekatkan kursor ke kaki komponen hinggan kursor berubah menjadi gambar pensil, kemudian tarik garis dari satu komponen ke komponen lain.



Gambar 9. 2 Penempatan Komponen

- Silahkan cek komponen yang sudah disusun, apakah sudah success atau error, dengan cara klik play di pojok kiri bawah. Jika ada berwarna merah berarti komponen yang disusun terjadi error dan sebaliknya jika berwarna hijau berarti komponen yang disusun sudah success.



Gambar 9. 3 simulasi rangkaian

2. Tabel Kebenaran

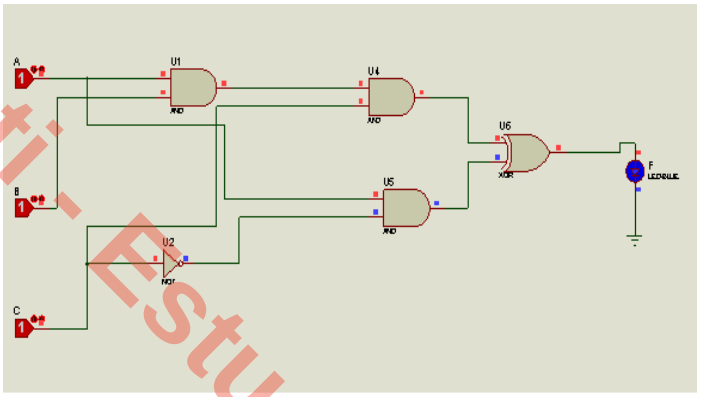
   Berikut adalah tabel kebenaran untuk simulasi diatas:

Tabel 9. 1 Tabel Kebenaran $F = (A + B)C + \underline{AC}$

INPUT			OUTPUT
A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

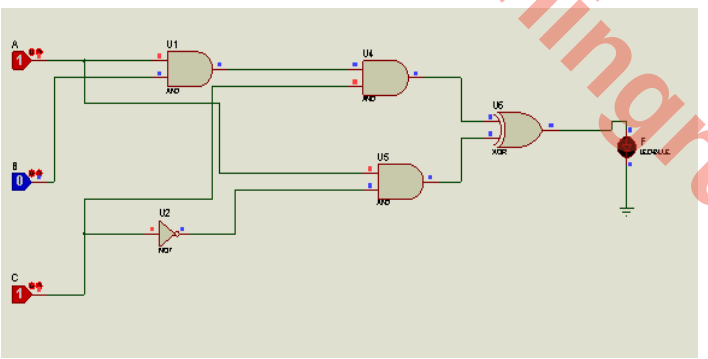
3. Simulasi

   Berdasarkan rangkaian diatas, saya mensimulasikan rangkaian tersebut dengan menggunakan proteus seperti gambar dibawah ini :



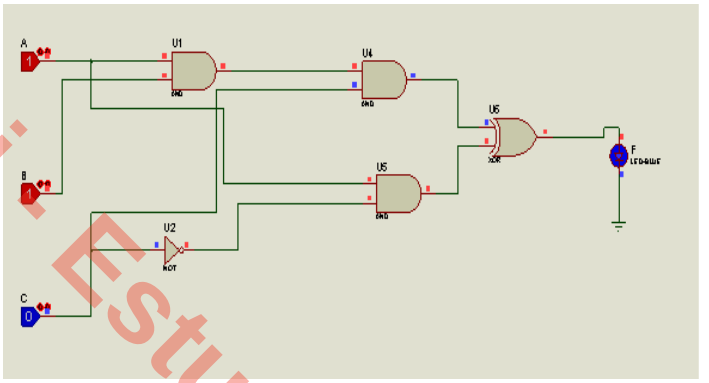
Gambar 9. 4 Rangkaian jika A,B,C berlogika 1

Pada gambar diatas, LED akan menyala jika logic state A,B,C berlogika 1. Dalam hal ini, fungsi logic state adalah sebagai input dari gerbang logika, dan output dari rangkaian tersebut ialah LED.



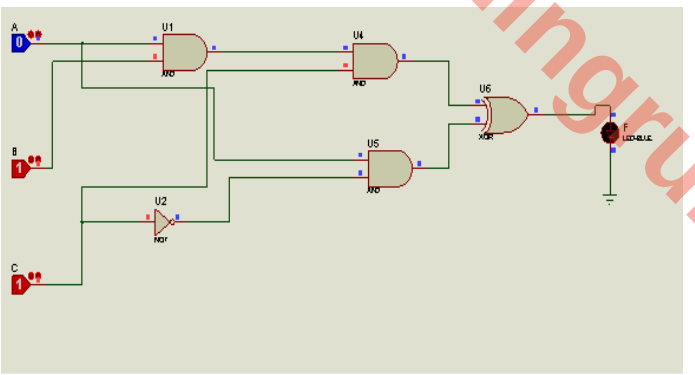
Gambar 9. 5 Rangkaian jika A dan C berlogika 1 dan B berlogika 0

Pada gambar diatas, LED akan mati jika logic state A dan C berlogika 1 sedangkan B berlogika 0. Dalam hal ini, fungsi logic state adalah sebagai input dari gerbang logika, dan output dari rangkaian tersebut ialah LED.



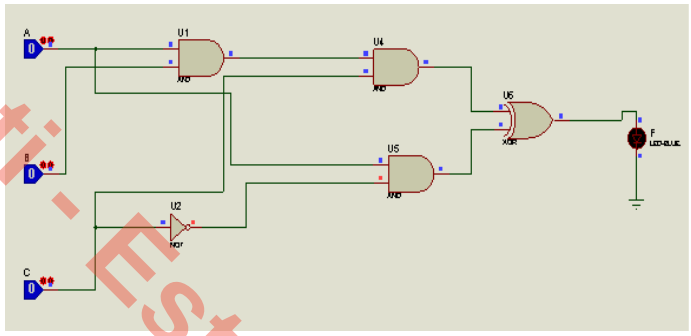
Gambar 9. 6 Rangkaian jika A dan B berlogika 1 dan C berlogika 0

Pada gambar diatas, LED akan menyala jika logic state A dan B berlogika 1 sedangkan C berlogika 0. Dalam hal ini, fungsi logic state adalah sebagai input dari gerbang logika, dan output dari rangkaian tersebut ialah LED.



Gambar 9. 7 Rangkaian jika B dan C berlogika 1 dan A berlogika 0

Pada gambar diatas, LED akan mati jika logic state B dan C berlogika 1 sedangkan A berlogika 0. Dalam hal ini, fungsi logic state adalah sebagai input dari gerbang logika, dan output dari rangkaian tersebut ialah LED.



Gambar 9. 8 Rangkaian jika A, B dan C berlogika 0

Pada gambar diatas, LED akan mati jika logic state A, B dan C berlogika 0. Dalam hal ini, fungsi logic state adalah sebagai input dari gerbang logika, dan output dari rangkaian tersebut ialah LED

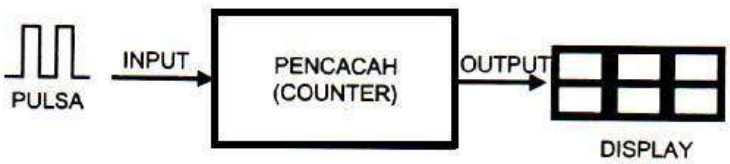
SOAL -SOAL LATIHAN

1. Sederhanakan dan buktikan persamaan $Y = \underline{(A + C)(B + D)}$
2. Sederhanakan dan buktikan persamaan $Q = ABC + A\underline{B}(\underline{AC})$
3. Sederhanakan dan buktikan persamaan $Z = ABC + AB\underline{C} + A\underline{B}C$
4. Sederhanakan dan buktikan persamaan $S = \underline{ABC} + \underline{AB}\underline{C} + \underline{AB}C + A\underline{BC} + \underline{ABC}$
5. Sederhanakan dan buktikan persamaan $X = \underline{ABC} + \underline{AB}\underline{C} + \underline{AB}C + ABC + A\underline{BC}$

BAB X
TEORI COUNTER

A. Pengertian Pencacah (Counter)

Pencacah atau *counter* merupakan rangkaian logika sekuensi yang berfungsi mencacah atau menghitung jumlah pulsa clock yang masuk. Menurut jumlah pulsa yang dapat dicacah, terdapat jenis pencacah modulo 2^n (n=1, 2, 3, 4, ...), contoh pencacah modulo-4, pencacah modulo-8, dan pencacah modulo-16. Jika clock ke-0 dinyatakan sebagai keadaan awal pencacah, jumlah pulsa yang dapat dicacah oleh pencacah modulo-4 adalah 4 buah yakni pulsa ke-0, ke-1, ke-2, ke-3, dan pada pulsa clock ke-4, output pencacah ini akan reset kembali ke 0. Pada pencacah modulo-8, output akan reset pada clock ke-8 sehingga pencacah ini hanya mampu mencacah pulsa clock ke-0 sampai dengan pulsa clock ke-7. Selain pencacah modulo 2 terdapat pula pencacah seperti modulo-5, modulo-6, dan modulo-10. Diagram blok pencacah ditunjukkan pada gambar 38



Gambar 10. 1 Diagram Block Pencacah

Sedangkan menurut pengaktifan elemen penyimpanannya dan dalam hal ini elemen penyimpan pencacah adalah flip-flop, terdapat pencacah jenis tak serempak atau pencacah tak sinkron (*asynchronous counter*), dan pencacah jenis serempak atau pencacah sinkron (*synchronous counter*). Pada pencacah tak serempak, elemen-

elemen penyusunnya yakni flip-flop bekerja secara tidak serempak ketika pencacah tersebut diberi input pulsa, dan pada pencacah serempak elemen-elemen penyusunnya bekerja secara bersama- sama ketika ada pulsa masuk ke inputnya. Prosedur perancangan kedua jenis pencacah tersebut agak berbeda. Untuk pencacah serempak prosedur perancangannya sama dengan prosedur perancangan rangkaian sekuensial seperti telah dijelaskan dimuka. Sedangkan untuk rangkaian pencacah tak serempak prosedur perancangannya lebih sederhana dan akan dijelaskan terlebih dahulu.

B. Pencacah Tak Serempak (Asyncronous Counter)

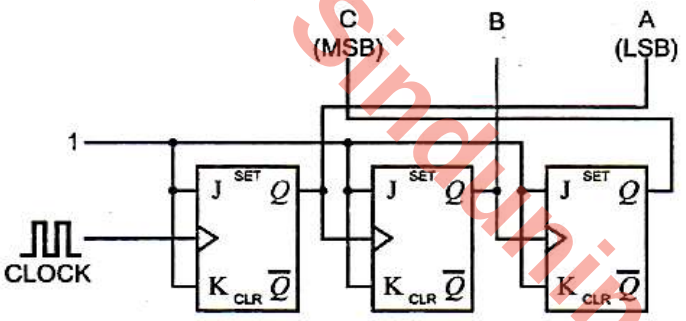
Pencacah Tak serempak tersusun atas flip-flop yang dihubungkan seri dan pemicunya tergantung dari flip-flop sebelumnya, kemudian menjalar sapai flip-flop MSB-nya. Karena itulah sering disebut juga sebagai *ripple- through counter*.

Untuk merancang pencacah tak serempak, perlu ditetapkan terlebih dahulu modulo dan pencacah yang akan dirancang. Untuk modulo-2^n, prosedur perancangannya mengikuti urutan sebagai berikut:

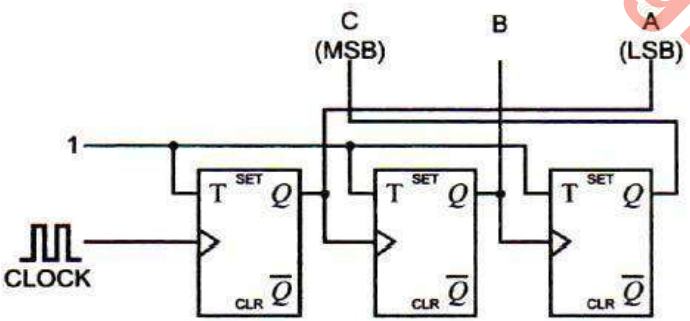
1. Tetapkan modulo pencacah yang akan dirancang, misalnya akan dirancang pencacah tak serempak modulo-8 atau modulo-2^ndengan n= 3.
2. Tentukan jumlah dan jenis flip-flop yang akan digunakan. Jumlah flip- flop yang digunakan adalah n buah. Jika akan digunakan flip-flop J-K, maka sediakan flip-flop J-K sebanyak n buah, dalam hal ini 3 buah.
3. Lakukan pengaturan input-input flip-flop yang digunakan. Untuk flipflop J-K, hubungkan semua input J dan input K dengan level logika 1. Untuk flip-flop T, hubungkan semua input T dengan level logika 1, dan untuk flip-flop D, hubungkan tiap input D dengan komplemen outputnya.
4. Berikan input pencacah ke input clock flip-flop paling kiri.

5. Hubungkan output flip-flop paling kiri dengan input clock flip-flop disebelah kanannya dan seterusnya.
6. Ambil output pencacah melalui setiap output flip-flop. Ingat: output flip- flop paling kiri adalah LSB dan ouput flop paling kanan adalah MSB.

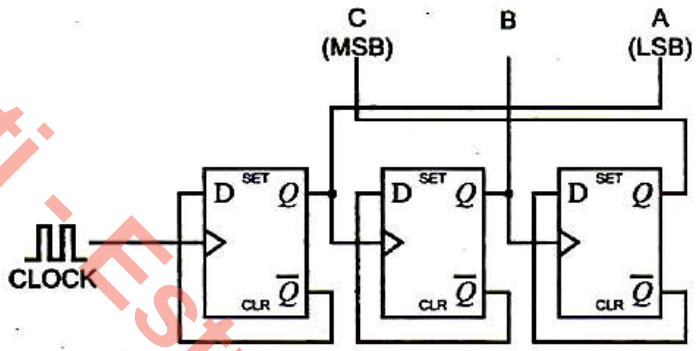
Melalui prosedur tersebut, dapat disusun rangkaian pencacah tak serempak modulo-8 dengan flip-flop J-K, flip-flop T maupun flip-flop D seperti ditunjukkan pada gambar 39.



(a)



(b)



(c)

Gambar 10. 2 Pencacah modulo-8 dengan flip-flop (a) JK-FF, (b) T-FF, dan (c) D-FF

Tabel 10. 1 Tabel Kebenaran pencacah tak serempak modulo-8

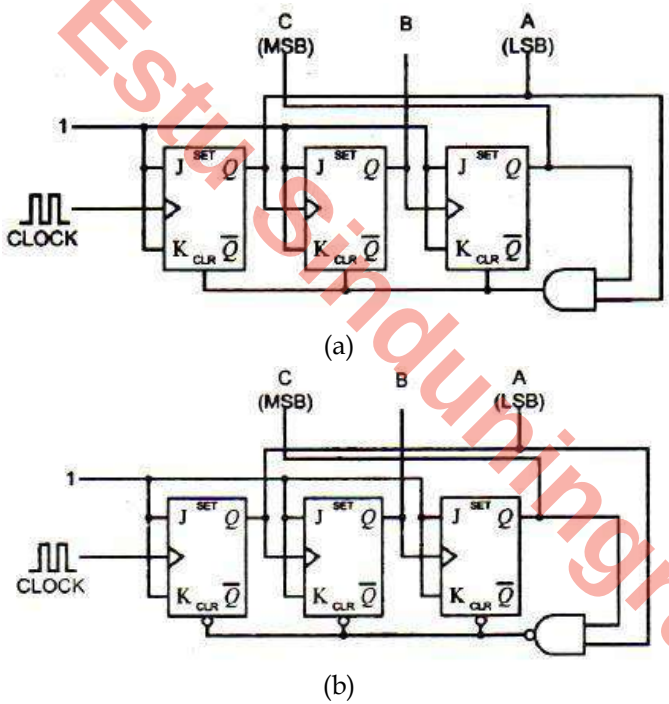
CACAH (COUNT)	OUTPUT		
	C	B	A
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	0	0	0

Tabel 10. 2 Tabel Kebenaran pencacah tak serempak modulo-5

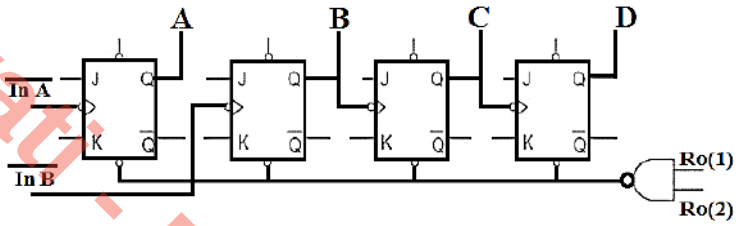
CACAH (COUNT)	OUTPUT		
	C	B	A
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1
8	0	0	0

1. Penjelasan dari Tabel kebenaran dari tabel kebenaran terlihat bahwa ketika pencacah memberikan output bernilai 5 desimal atau CBA=101 biner, karena harus reset yakni semua outputnya 0 maka pada dock-5 output C dan A keduanya harus berubah dan 1 ke 0. Perubahan tersebut dapat dilakukan dengan menghubungkan clear setiap flip-flop dengan suatu gerbang. Gerbang harus dapat membangkitkan sinyal yang diperlukan untuk clear jika inputnya C=1 dan A=1.
2. Jika jenis clear pada flip-flop adalah active-low (clear jika diberi 0), maka gerbang yang digunakan harus dapat mernbangkitkan sinyal 0 jika kedua inputnya 1 yakni berasal dan C dan A. Untuk keadaan ini gerbang yang digunakan untuk melakukan clear adalah NAND.

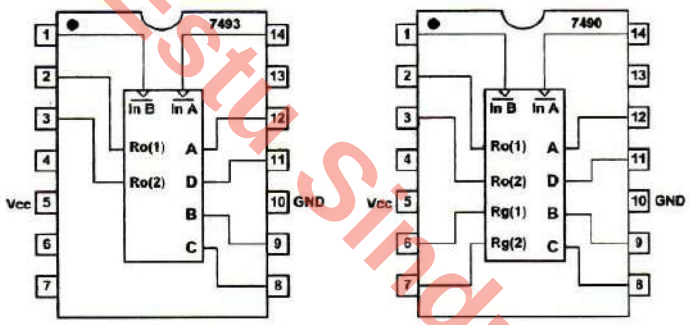
3. Jika jenis clear pada flip-flop adalah active-high (clear jika diberi 1), maka gerbang yang digunakan harus dapat rnembangkitkan sinyal 1 jika kedua inputnya 1 yakni berasal dan C dan A. Untuk keadaan ini gerbang yang digunakan untuk melakukan clear adalah AND.



Gambar 10. 3 Rangkaian pencacah modulo-5 dengan flip-flop J-K : (a) clear jenis active-high, dan (b) clear jenis active-low



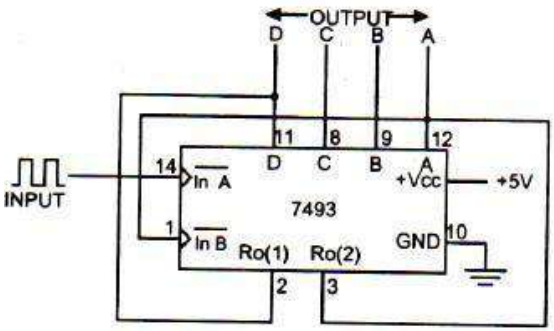
(a)



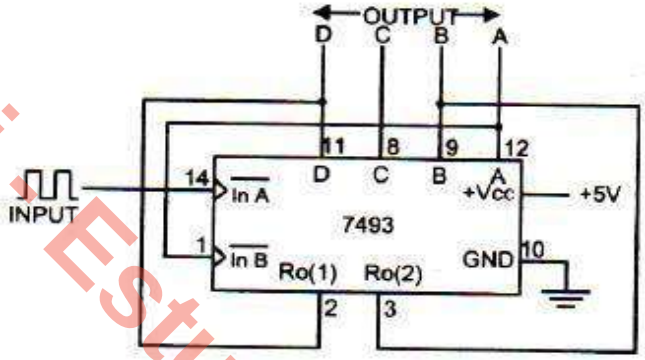
(b)

Gambar 10. 4 IC Pencacah tak sinkron: (a) rangkaian internal IC 7493, (b)spesifikasi pin IC 7493, dan spesifikasi

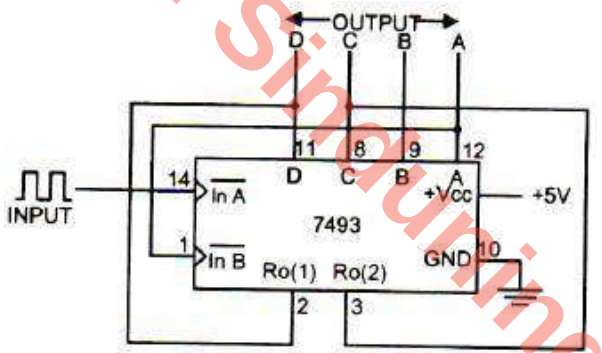
Dengan menggunakan IC 7493 dapat disusun rangkaian pencacah sampai dengan modulo-16



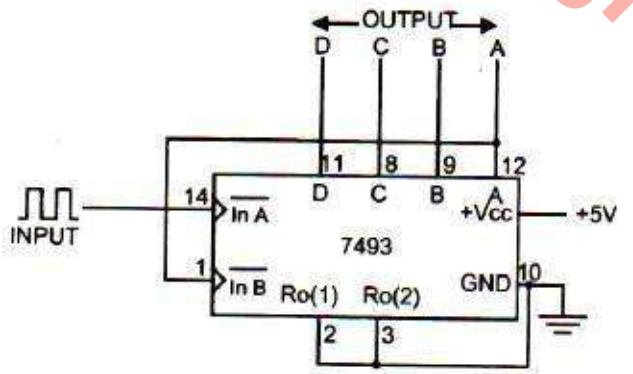
(a)



(b)



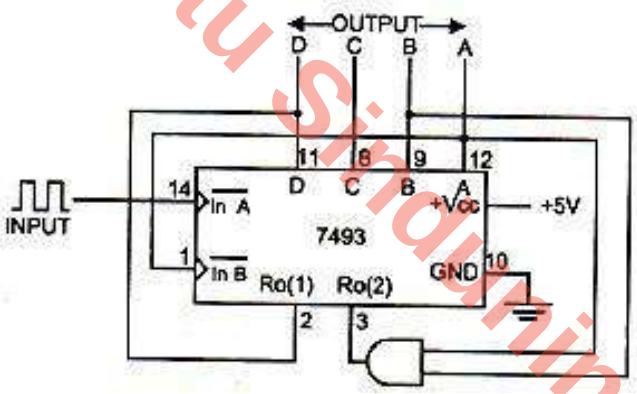
(c)



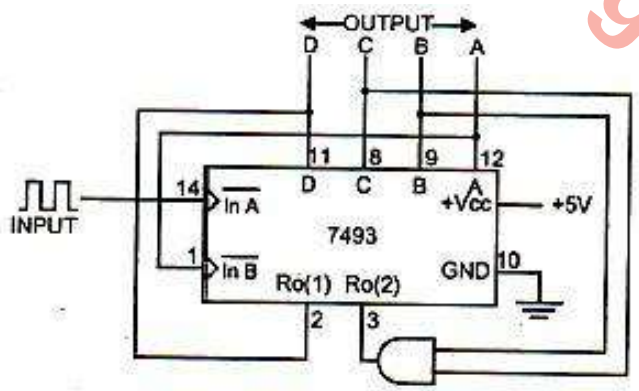
(d)

Gambar 10. 5 **IC 7493 sebagai pencacah tak serempak: (a) modulo-9, (b) modulo-10, (c) modulo-12), dan (d) modulo-16**

Karena gerbang NAND sebagai fasilitas clear yari g ada di dalam rangkaian internal 7493 hanya memiliki dua buah input, maka untuk pencacah yang memerlukan clear terhadap 3 buah outputnya seperti pencacah modulo- 11, modulo-13, modulo-14, dan modulo-15 diperlukan rangkaian luar atau rangkaian tambahan. Gambar 43 menunjukkan rangkaian tambahan yang diperlukan untuk membangun pencacah serempak modulo-11, dan modulo- 14.

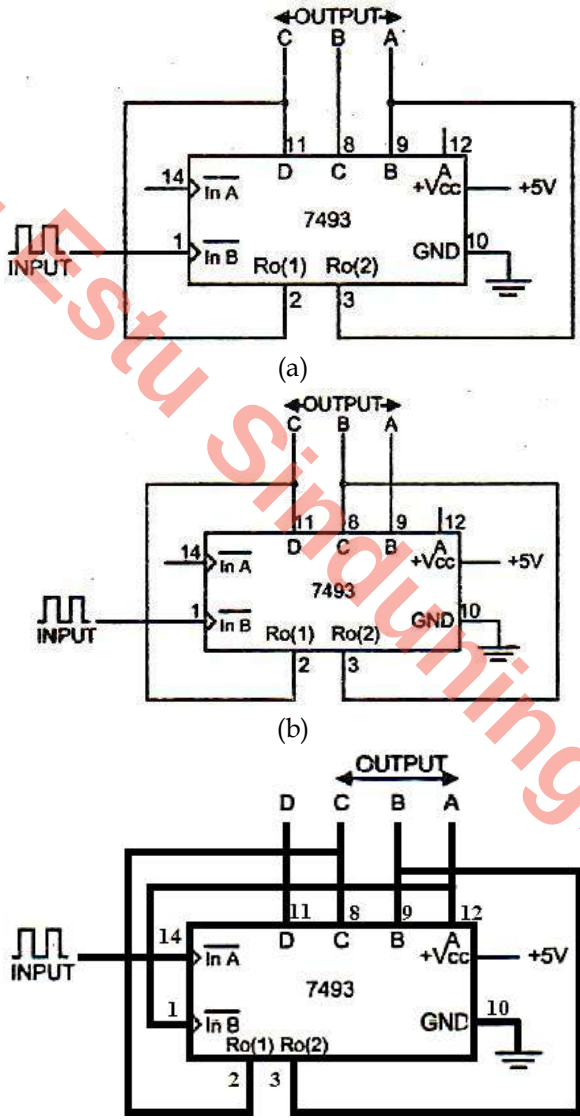


(a)



(b)

Gambar 10. 6 IC 7493 sebagai pencacah tak serempak (a) modulo-11, dan (b)modulo-14



Gambar 10. 7 IC 7493 sebagi pencacah tak serempak: (a) modulo-5, (b) dan (c) modulo 6

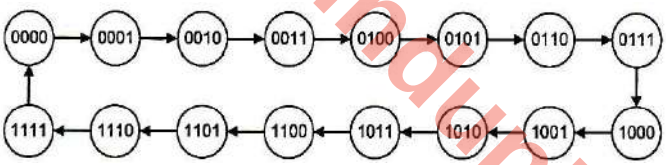
C. Pencacah Serempak

Pencacah serempak merupakan rangkaian sekuensial serempak, sehingga perancangannya dilakukan dengan menggunakan prosedur seperti pada perancangan rangkaian sekuensial serempak.

Contoh 1 : Rancang pencacah serempak modulo-16

Jawab :

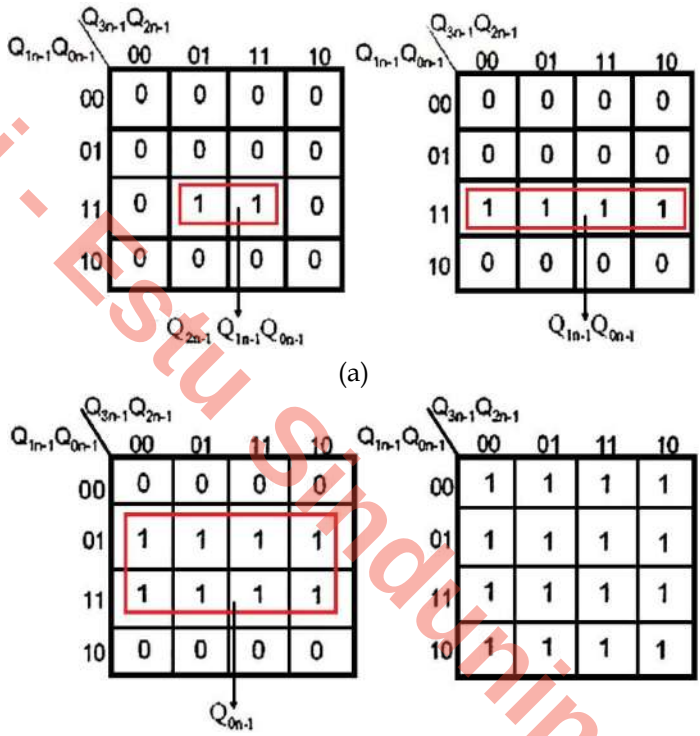
1. Merancang rangkaian ini adalah menyusun diagram transisi keadaan berdasarkan definisi watak. Pencacah serempak moduk-16 adalah pencacah yang keadan outputnya akan reset pada clock ke- 16.



Gambar 10. 8 Diagram transisi keadaan pencacah modulo-16

Pada unit rangkaian kombinasi pada rangkaian pencacah serempak tidak memiliki input, dan output rangkaian diambil dan output unit penyimpannya maka diagram transisinya hanya menggambarkan keadaan transisi dan output elmen-elemen penyimpannya saja.

2. Menyusun tabel keadaan berdasarkan diagram transisi keadaan yang diperoleh. Dan gambar 106, dapat disusun tabel keadaan dan rangkaian pencacah serempak modolo-16 seperti ditunjukkan pada tabel 28
3. Selanjutnya, berdasarkan table 29 eksitasi tersebut disusun peta Karnaugh untuk menentukan fungsi dan masing-masing input pada flip-flop yang digunakan



(a)

(b)

Gambar 10. 9 Peta Karnaugh tabel 37 untuk: (a) T_3, (b) T_2, (c) T_1, dan (d)T_0

Tabel 10. 3 Tabel keadaan rangkaian pencacah serempak modulo-16

KEADAAN SEBELUMNYA				KEADAAN SEKARANG			
Q3(n−1)	Q2(n−1)	Q1(n−1)	Q0(n−1)	Q3(n)	Q2(n)	Q1(n)	Q0(n)
0	0	0	0	0	0	0	1
0	0	0	1	0	0	1	0
0	0	1	0	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	0	0	1	0	1
0	1	0	1	0	1	1	0
0	1	1	0	0	1	1	1
0	1	1	1	1	0	0	0
1	0	0	0	1	0	0	1
1	0	0	1	1	0	1	0
1	0	1	0	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	0	1	1	0	1
1	1	0	1	1	1	1	0
1	1	1	0	1	1	1	1
1	1	1	1	0	0	0	0

Tabel 10. 4 Tabel ekstitasi flip-flop T

KEADAAN SEBELUMNYA				KEADAAN SEKARANG				KEADAAN INPUT FLIP-FLOP			
Q3(n−1)	Q2(n−1)	Q1(n−1)	Q0(n−1)	Q3(n)	Q2(n)	Q1(n)	Q0(n)	T3	T2	T1	T0
0	0	0	0	0	0	0	1	0	0	0	1
0	0	0	1	0	0	1	0	0	0	1	1
0	0	1	0	0	0	1	1	0	0	0	1
0	0	1	1	0	1	0	0	0	1	1	1
0	1	0	0	0	1	0	1	0	0	0	1
0	1	0	1	0	1	1	0	0	0	1	1
0	1	1	0	0	1	1	1	0	0	0	1
0	1	1	1	1	0	0	0	1	1	1	1
1	0	0	0	1	0	0	1	0	0	0	1
1	0	0	1	1	0	1	0	0	0	1	1
1	0	1	0	1	0	1	1	0	0	0	1
1	0	1	1	1	1	0	0	0	1	1	1
1	1	0	0	1	1	0	1	0	0	0	1
1	1	0	1	1	1	1	0	0	0	1	1
1	1	1	0	1	1	1	1	0	0	0	1
1	1	1	1	0	0	0	0	1	1	1	1

Dari peta Karnaugh gambar 107 tersebut dapat diturunkan fungsi input setiap flip-flop seperti ditunjukkan pada persamaan dibawah ini.

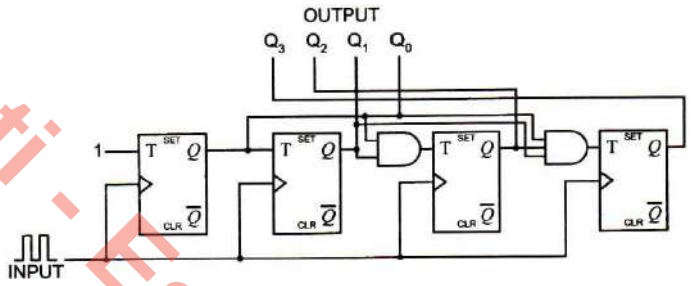
$T3 = O2n-1O1n-1O0n-1$

$T2 = O1n-1O0n-1$

$T1 = O0n-1$

$T_0 = 1$

Berdasarkan persamaan diatas dapat disusun rangkaian pencacah serempak modulo-16.



Gambar 10. 10 Rangkaian pencacah serempak modulo-16 dengan flip-flop T

D. Pencacah Naik-Turun

Pencacah naik-turun (*up-down counter*) merupakan pencacah yang urutan transisi keadaannya dapat diatur naik (*count-up*) atau turun (*count- down*). Contoh pencacah up-down modulo-10 jika diatur agar mencacah naik maka urutan transisi keadaannya adalah 0, 1, 2, 3,4, 5,6, 7, 8, 9, dan kembali ke urutan semula, dan jika urutannya diatur turun maka transisi keadaannya adalah 9,8,7,6,5,4,3,2, 1,0, dan kembali ke urutan semula. Implementasi pencacah ini dapat dilakukan dalam berbagai bentuk antara lain menggunakan pencacah serempak.

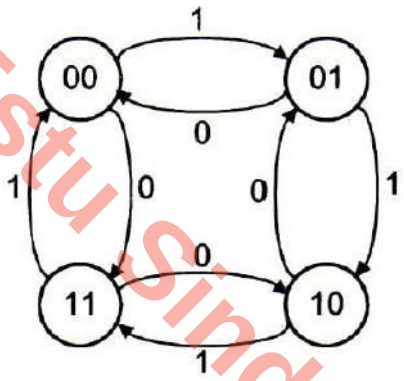
Contoh soal :

Rancang pencacah naik-turun modulo-4 yang memiliki input pengontrol urutan R. Jika R=1 output pencacah memberikan urutan naik dan apabila R=0 output pencacah memberikan urutan turun.

Jawab:

Tahap pertama perancangan pencacah ini adalah dengan menyusun terlebih dahulu diagram transisi keadaan. Oleh karena rangkaian pencacah serempak merupakan rangkaían sekuensial serempak yang outputnya diarnbil dan output flip-flop penyusunnya, atau dengan kata lain pada bagian rangkaian kombinasinya tidak memiliki output, maka pada diagram transisi keadaan hanya dicantumkan notasi

inputnya saja untuk setiap terjadinya transisi keadaan. Berdasarkan definisi wataknya, dapat disusun diagram transisi keadaan pencacah naik-turun modulo-4 seperti ditunjukkan pada gambar 48.



Gambar 10. 11 Diagram transisi keadaan pencacah naik-turun modulo-4.

Dan diagram transisi keadaan pada gambar 48, dapat disusun tabel keadaan seperti ditunjukkan pada tabel 30.

Tabel 10. 5 Keadaan rangkaian pencacah naik-turun modulo-4.

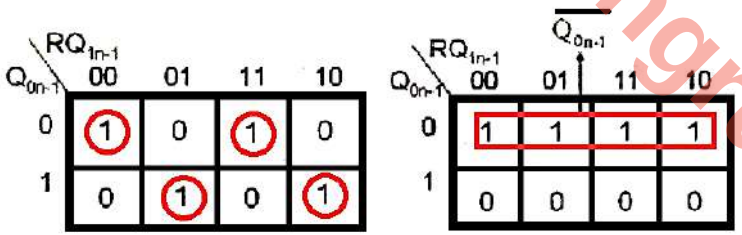
KEADAAN SEBELUMNYA		KEADAAN SEKARANG			
		R=0		R=1	
Q1(n−1)	Q0(n−1)	Q1(n)	Q0(n)	Q1(n)	Q0(n)
0	0	1	1	0	1
0	1	0	0	1	0
1	0	0	1	1	1
1	1	1	0	0	0

Untuk memperoleh fungsi input dari flip-flop penyusun rangkaian pencacah ini perlu disusun terlebih dahulu tabel eksitasi. Karena tabel eksitasi menunjukkan pengaruh eksitasi pada input flip-flop terhadap outputnya,

maka perlu ditetapkan terlebih dahulu jenis flip-flop yang digunakan. Tabel 29 menunjukkan tabel eksitasi untuk flip-flop D. Karena output flip-flop D sama dengan inputnya, maka keadaan input flip-flop D diisi sama dengan output keadaan sekarang

Tabel 10. 6 Tabel eksitasi flip-flop D untuk tabel 30

INPUT	KEADAAN SEBELUMNYA		KEADAAN SEKARANG		KEADAAN INPUT FLIP-FLOP	
R	Q1(n−1)	Q0(n−1)	Q1(n)	Q0(n	D1	D0
0	0	0	1	1	1	1
0	0	1	0	0	0	0
0	1	0	0	1	0	1
0	1	1	1	0	1	0
1	0	0	0	1	0	1
1	0	1	1	0	1	0
1	1	0	1	1	1	1
1	1	1	0	0	0	0



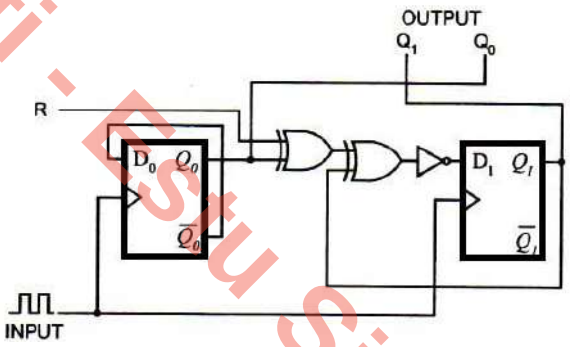
Gambar 10. 12 Peta Karnaugh tabel 29 untuk (a) D_1**, (b)** D_0

Berdasarkan peta Karnaugh pada gambar 108, terlihat bahwa semua *minterm* untuk D1 terisolasi, namun demikian karena peta Karnaugh untuk D_1 sama dengan peta Karnaugh rangkaian *exclusive*-NOR (XNOR), maka D_1 merupakan fungsi XNOR atau komplemen dari XOR. Dengan demikian dapat diperoleh fungsi D_1 dan D_0 seperti ditunjukkan pada persamaan dibawah.

$D_1 = \underline{R \oplus Q_{1n-1} \oplus Q_{0n-1}}$

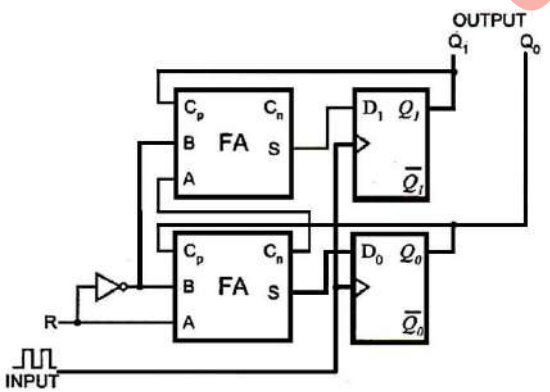
$D_0 = Q_{1n-1}$

Dan persamaan diatas dapat digambar rangkaian pencacah naik-turun modulo-4 seperti pada gambar 50



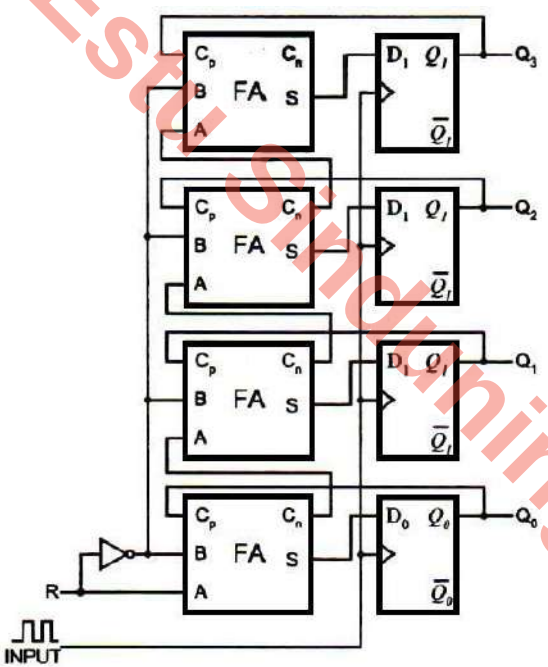
Gambar 10. 13 Rangkaian pencacah naik-turun serempak modulo-4

Selain dapat diimplementasikan menggunakan rangkaian kombinasi dan flip-flop D seperti yang telah dibahas di muka, pencacah naik-turun juga dapat dümplementasikan dengan menggunakan rangkaian kombinasi full-adder dan flip-flop D seperti ditunjukkan pada gambar 51.



Gambar 10. 14 Implementasi pencacah naik-turun serempak modulo 4 dengan full-adder

Pada gambar 51 terlihat bahwa untuk pencacah modulo-4 memerlukan full adderdan flip-flop D masing-masing 2 buah. Dengan demikian jika diinginkan pencacah serempak naik-turun dengan modulo yang lebih tinggi misalnya modulo-16, diperlukan full adder dan flip-flop D masing-masing 4 buah.



Gambar 10. 15 Pencacah serempak naik-turun modulo-16 dengan full- adder

E. IC Pencacah Serempak 74193

Salah satu seri IC yang menyediakan fungsi pencacah serempak adalah 74193. IC ¡ni dilengkapi dengan fasilitas preset dan kendali naik-turun dengan kemampuan pencacahan sampai modulo-16. Karena dilengkapi dengan fasilitas-fasilitas tersebut, IC ini dinamakan presettable up/down counter. Pada pencacah jenis ini, sinyal preset diberikan melalui suatu input paralel dan untuk memindah

data preset tersebut ke output pencacah digunakan pengendali *parallel load* (PL). Rangkaian pencacah naik modulo-16 yang dilengkapi dengan preset dan untuk pencacah turun. Untuk memberikan preset pada pencacah tersebut dilakukan dengan cara memasang sinyal preset terlebih dahulu pada input P_3, P_2 ,P_1 dan P_0. Selanjutnya diberikan sinyal $\underline{PL}$

SOAL – SOAL LATIHAN

1. Gambarkanlah diagram pewaktu (bentuk gelombang) pada pencacah biner 5 bit naik untuk 15 detak masukan.
2. Rancanglah suatu rangkaian pencacah riak (tak sinkron) yang dapat menyalakan LED selama 40 ms dan mematikannya selama 20 ms. Frekuensi detak yang dikenakan pada pencacah tersebut sebesar 100 Hz.
3. Rancanglah suatu rangkaian pencacah tak sinkron yang dapat mencacah 0-1-2-3-4-5-6-7-8-9-10-11 kemudian berhenti untuk menyalakan LED sebagai tanda bahwa proses pencacahan berhenti. Proses pencacahan tersebut dimulai dengan cara menekan suatu tombol.

BAB XI
RANGKAIAN ARITMATIKA

A. Operasi dasar aritmetika bilangan biner

1. 1's *complement*, beserta contoh

   Komplemen satu merupakan suatu sistem penomoran yang diterapkan dalam beberapa jenis komputer untuk merepresentasikan nilai-nilai negatif. Pada cara ini terdapat aturan bahwa nilai 0 (nol) akan direpresentasikan dengan dua buah nilai, yaitu +0 (positif nol) dan -0 (negatif nol).

   Komplemen 1 di sistem bilangan binari dilakukan dengan mengurangkan setiap bit (digit) dari nilai 1, atau dengan mengubah setiap bit 0 menjadi 1 dan bit 1 menjadi 0. Dengan komplemen 1, hasil digit paling kiri dipindahkan untuk ditambahkan pada bit paling kanan.

2. 2's *complement,* beserta contoh

   Komplemen dua mirip dengan komplemen satu, hanya saja dalam proses negasinya semua bit juga akan dibalik, sehingga tidak ada lagi rasa "bingung" merepresentasikan nilai +0 dan -0, karena hanya ada satu nilai 0 (nol), seperti berikut:

```
000...00011 = +3
000...00010 = +2
000...00001 = +1
000...00000 = +0
111...11111 = -0
111...11110 = -1
111...11101 = -2
111...11100 = -3
```

Gambar 11. 1 2's complement

Komplemen 2 pada sistem bilangan binari adalah hasil dari komplemen 1 ditambah 1, misalnya komplemen 2 dari binari 10110 adalah 01010 (dari komplemen 1 yaitu 01001 ditambah 1). Dengan komplemen 2, hasil digit paling kiri dibuang (tidak digunakan).

Sedangkan contoh pengurangan dengan komplemen 1 pada sistem bilangan binari adalah sebagai berikut :

1. Operasi penjumlah setengah/*half adder* dan penjumlah penuh/*full adder*
   a. Pengertian

      Half Adder adalah untai logika yang keluarannya merupakan jumlah dari dua bit bilangan biner. Berdasarkan dua input, yaitu A dan B, maka outpunya adalah S(sum), S atau sum ini akan dihitung berdasarkan implementasi operasi logika XOR dari A dan B. Selain Output S(sum), masih ada lagi output lain yang kita kenal dengan C(carry), nah sedangkan output C(carry) ini dihasilkan dari implementasi operasi logika AND.

      Full Adder adalah untai logika yang keluarannya merupakan hasil dari 3 bit bilangan biner. maka prinsip kerjanya juga sama seperti half-adder, hanya saja Full-adder mampu menampung carry dari hasil penjumlahan sebelumnya. Sehingga dengan adanya carry tersebut, maka jumlah inputnya sewaktu-waktu bisa jadi 3 (tergantung kondisi carrynya, apakah aktif/tidak)[1].

   b. Tabel Kebenaran
      1) *Half Adder*

         Untuk penjumlahan 2 Bit Biner (LSB/*Least Sigbificant Bit*)

Tabel 11. 1 Tabel kebenaran *Half Adder*

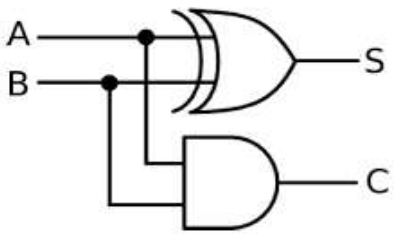
A_o	B_o	=	Σ_o	C_{out}
0	0	=	0	0
0	1	=	1	0
1	0	=	1	0
1	1	=	0	1

2) *Full Adder*
Untuk penjumlahan 3 Bit Biner

Tabel 11. 2 Tabel kebenaran *Full Adder*

A_i	B_i	C_i	=	Σ_i	C_{out}
0	0	0	=	0	0
0	0	1	=	1	0
0	1	0	=	1	0
0	1	1	=	0	1
1	0	0	=	1	0
1	0	1	=	0	1
1	1	0	=	0	1
1	1	1	=	1	1

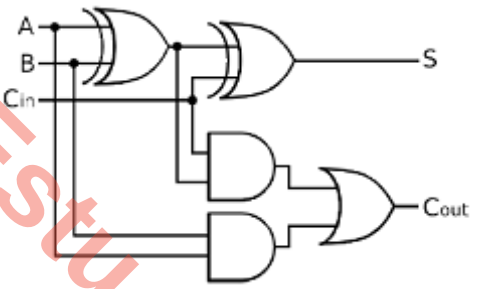
c. Rangkaian Logika
1) *Half Adder*
Untuk penjumlahan 2 Bit Biner



Rangkaian Half Adder

Gambar 11. 2 Rangkaian *Half Adder*

2) *Full Adder*
Untuk penjumlahan 3 Bit Biner



Gambar 11. 3 Rangkaian *Full Adder*

2. Operasi pengurang setengah/*half subtractor* dan pengurang penuh/*full subtractor*
   a. Pengertian

      Half subtractor adalah suatu rangkaian yang dapat digunakan untuk melakukan operasi pengurangan data-data bilangan biner hinga 1 bit saja. *Half Subtractor* mempunyai karakteristik : 2 masukan yaitu *input* A dan B serta 2 *output* yaitu *Diference* (Dif) dan *Borow* (Br)[2].

      Full subtractor adalah rangkaian yang digunakan untukn pengurangan bilangan-bilangan biner yang lebih dari 1 bit. rangkaian ini terdiri dari 3 terminal *input* (a, b, dan carry-in) dan 2 terminal *output* (*difference* dan *borrow*). rangkaian *full subtractor* dibentuk dari 2 buah rangkaian *half subtractor.*

b. Tabel Kebenaran

1) *Half Subractor*

Tabel 11. 3 Tabel kebenaran

A_0	B_0	R_0	B_{out}
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

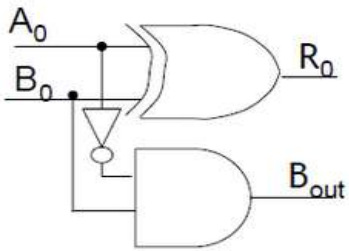
2) *Full Subractor*

Tabel 11. 4 Tabel kebenaran

A	B	Bor_{in}	Dif	Bor_{out}
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	1	0	0	0
1	1	1	1	1

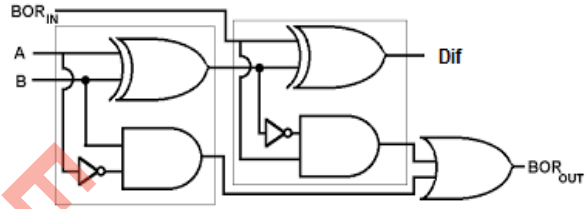
c. Rangkaian Logika

1) *Half Subractor*



Gambar 11. 4 *Half Subractor*

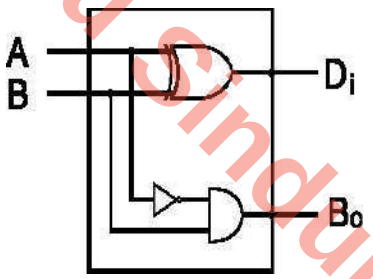
2) *Full Subractor*



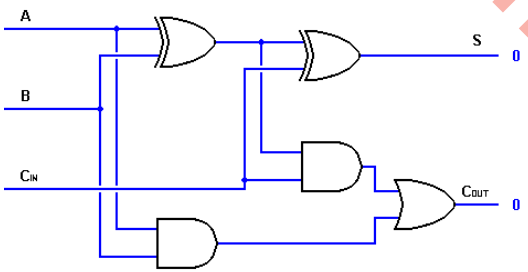
Gambar 11. 5 ***Full subractor***

SOAL – SOAL LATIHAN

1. Hitunglah operasi di bawah ini menggunakan 1's *comp* dan 2's *comp*
   a. 10 + 4
   b. 8 – 6
2. Buatlah rangkaian di bawah ini pada simulator hanya dengan menggunakan gerbang dasar
   a. Rangkaian *half subtractor*

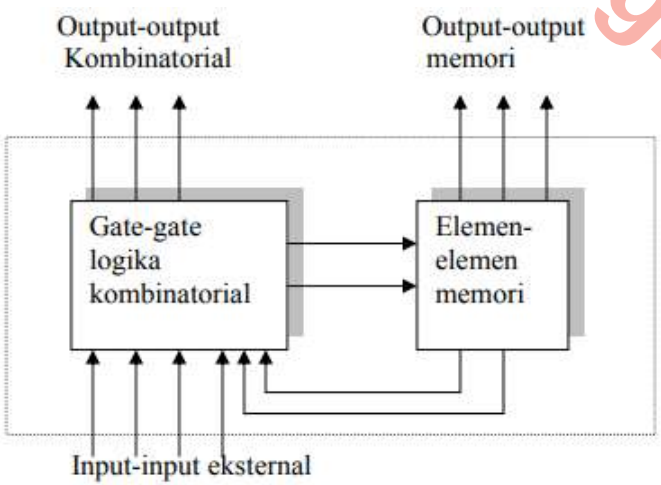


   b. Rangkaian *full adder*



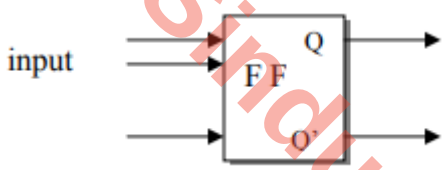
BAB XII
RANGKAIAN FLIP FLOP

Sejauh ini rangkaian logika yang telah dibahas adalah rangkaian logika kombinatorial yang level-level outputnya pada setiap saat tertentu tergantung kepada level-level yang terdapat pada input-inputnya pada saat itu. Keadaan level input yang terdahulu tidak mempunyai pengaruh terhjadap output-output yang kemudian karena rangkaian kombinatorial tidak mempunyai memori. Sistem-sistem digital kebanyakan terbuat dari dua-duanya, rangkaian-rangkaian kombinatorial dan elemen-elemen memori. Gambar 12.1 menunjukkan diagram blok dari suatu system digital umum yang menggabungkan gate-gate logika dengan elemen-elemen memori. Blok kombinatorial menerima sinyal-sinyal logika dari input-input luar dan dari output-output elemen-elemen memori.



Gambar 12. 1 Diagram system digital umum

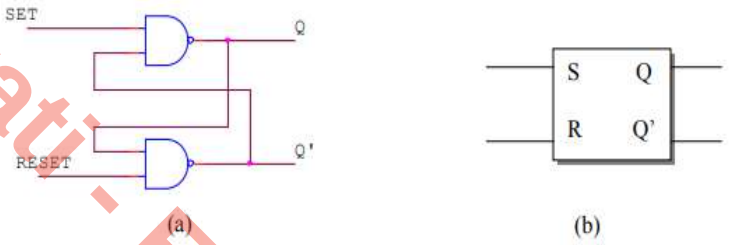
Elemen memori yang paling banyak digunakan adalah Flip-Flop (FF). FF adalah suatu rangkaian logika dengan dua output yang saling berlawanan. Gambar 10.2 menunjukkan symbol flip-flop secara umum. Untuk FF ada dua keadaan kerja yang mungkin : (1) Q = 0, Q'=1 : dan (2) Q = 1, Q' = 0 . FF mempunyai satu input atau lebih yang digunakan untuk mengoperasikan FF bolak-balik antara dua keadaan tersebut. Sekali sebuah sinyal input mengoperasikan FF menuju suatu keadaan tertentu, FF tersebut akan tetap berada pada keadaan itu meskipun setelah sinyal inputnya terputus. Ini adalah karakteristik memori dari rangkaian FF.



Gambar 12. 2 Simbol FF secara umum

A. NAND Gate Latch

Rangkaian dasar Flip-Flop dapat disusun dari dua buah NAND gate atau NOR gate. Apabila disusun dari NAND gate, disebut dengan NAND gate latch atau secara sederhana disebut latch, seperti ditunjukkan pada gambar 10.3 (a). Dua buah NAND gate disilangkan antara output NAND gate-1 dihubungkan dengan salah satu input NAND gate-2, dan sebaliknya. Output gate (output latch) diberi nama Q dan Q'. Pada kondisi normal kedua output tersebut saling berlawanan. Input latch diberi nama SET dan RESET. Gambar 10.3 (b) menunjukkan symbol dari NAND gate latch.



Gambar 12. 3 NAND Gate Latch

Tabel 12. 1 Tabel kebenaran

Set	Reset	Keluaran FF
1	1	Q (tak berubah)
0	1	Q = 1 ; Q′ = 0
1	0	Q = 0 ; Q′ = 1
0	0	Tak Tertentu

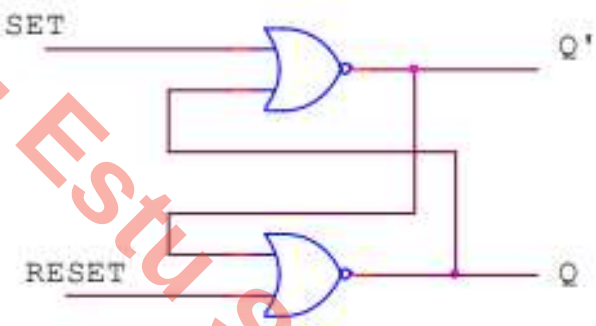
Ikhtisar dari NAND gate latch :

1. SET = 0; RESET = 1 selalu menghasilkan Q=1, tanpa mempedulikan keadaan output FF sebelumnya. Ini disebut mengeset atau setting FF pada keadaan 1 atau keadaan tinggi
2. SET = 1, RESET = 0 selalu menghasilkan Q= 0, tanpa mempedulikan keadaan output FF sebelumnya. Ini disebut mereset FF pada keadaan 0 atau keadaan rendah.
3. SET = 1, RESET = 1 tidak mempengaruhi keadaan FF. FF tetap berada pada keadaan sebelumnya.
4. SET = 0, RESET = 0 adalah keadaan tak menentu dan tidak seharusnya digunakan.

B. NOR Gate Latch

Dua buah NOR gate yang saling disilangkan dikenal sebagai NOR gate latch, dengan dua buah output Q dan Q′ yang saling berlawanan serta dua buah input SET dan RESET, seperti ditunjukkan pada gambar 7.4. Jika logika 1 diberikan pada input S, maka kondisi ini menyebabkan FF di set ke 1

(Q=1). Jika logika 1 diberikan ke input R, maka kondisi ini menyebabkan FF di reset ke 0 (Q=0).



Gambar 12. 4 NOR gate Latch

Tabel 12. 2 tabel kebenaran

Set	Reset	Keluaran FF
1	1	Q (tak berubah)
0	1	Q = 1 ; Q′ = 0
1	0	Q = 0 ; Q′ = 1
0	0	Tak Tertentu

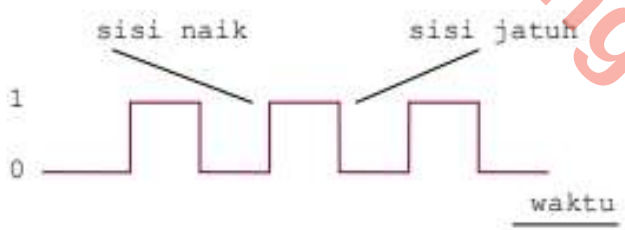
Ikhtisar dari NOR gate latch

1. SET = 1, RESET = 0 selalu menghasilkan Q = 1, tanpa mempedulikan keadaan output FF sebelumnya. Ini disebut mengeset atau stting FF pada keadaan 1 atau keadaan tinggi.
2. SET = 0, RESET = 1 selalu menghasilkan Q = 0, tanpa mempedulikan keadaan output FF sebelumnya. Ini disebut mereset FF pada keadaan 0 atau keadaan rendah.
3. SET = 0, RESET = 0 tidak mempengaruhi keadaan FF. FF tetap berada pada keadaan sebelumnya.
4. SET = 1 , RESET = 1 adalah keadaan tak menentu dan tidak seharusnya digunakan.

5. harga 1 pada SET atau RESET, yang digunakan untuk mengubahnkeadaan FF, dapat merupakan suatu tegangan DC atau pulsa sesaat.

C. Pulsa Clock (Sinyal jalan)

Hampir semua system digital beroperasi sebagai system-sistem urutan sinkron atau synchronous sequential system. Yang dimaksud adalah bahwa urutan operasi disinkronisasikan oleh suatu pulsa yang disebut pulsa clock. Pulsa clock yaitu pulsa-pulsa periodik yang biasanya berbentuk bujur sangkar (duty cycle 50%), seperti yang ditunjukkan pada gambar 10.5. Operasi-operasi yang terjadi di dalam system digital diusahakan terjadi pada waktu-waktu pulsa clock bertransisi dari 0 ke 1 atau dari 1 ke 0. Waktu-waktu transisi ini ditunjukkan pada gambar 10.5. Transisi 0-ke-1 disebut sisi naik (rising edge) atau sisi menuju positip, transisi dari 1-ke-0 disebut sisi jatuh (falling edge) atau sisi menuju negatip.



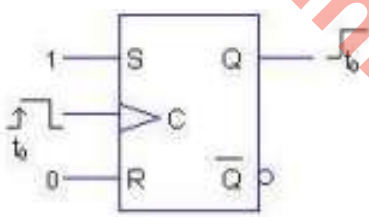
Gambar 12. 5 Pulsa Clock (Sinyal jam)

Pulsa clock ini digunakan pada Flip-Flop untuk mengubah keadaan-keadaan pada salah satu sisi naik atau sisi turun dari pulsa clock. Dengan kata lain pulsa clock FF akan mengubah keadaan-keadaan pada transisi clock yang sesuai dan akan diam/istirahat (rest) antara pulsa-pulsa clock yang berurutan. Frekuensi dari pulsa-pulsa clock biasanya ditentukan oleh berapa lama waktu yang dibutuhkan FF dan gate-gate di dalam rangkaian untuk memberikan respond

terhadap level perubahan-perubahan yang dikomando oleh pulsa clock. Pada sub-bab 7.4 akan dimulai dipelajari berbagai macam clocked flip-flop yang begitu luas penggunaannya di hampir semua sistem-sistem digital.

D. Clocked SR Flip Flop

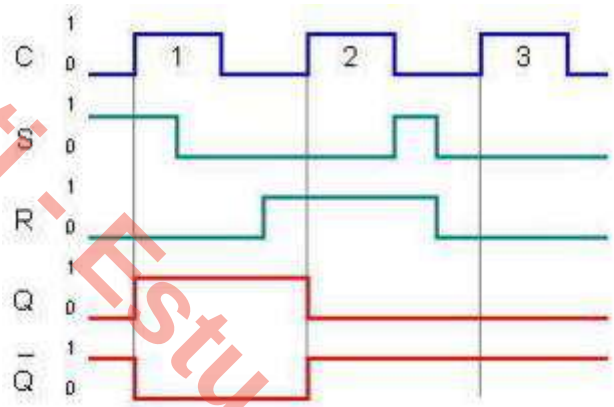
Gambar 12.6 menunjukkan sebuah clocked SR flip-flop yang dikomando oleh sisi menuju positip dari pulsa clock. Ini berarti bahwa FF akan mengubah keadaan hanya apabila suatu sinyal diberikan kepada clock inputnya (disingkat CLK atau C) melakukan suatu transisi dari 0 ke 1. Input-input S dan R mengontrol keadaan FF dengan cara yang sama seperti yang diuraikan pada SR FF dasar (tanpa clock), tetapi FF tersebut tidak akan memberikan respon kepada input-input ini sampai saat terjadinya transisi sisi naik dari pulsa clock. Ini ditunjukkan oleh bentuk gelombang pada gambar 12.7.



Gambar 12. 6 clocked SR Flip - Flop dengan pulsa clock aktif tinggi

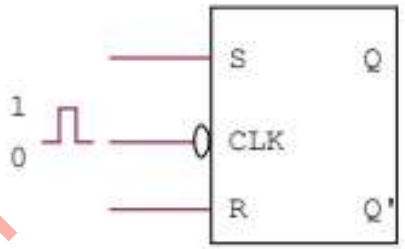
Tabel 12. 3 kebenaran

Input			Output		
S	R	C	Q	$\overline{Q}$	Comments
0	0	↑	Q	$\overline{Q}$	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	↑	?	?	Invalid



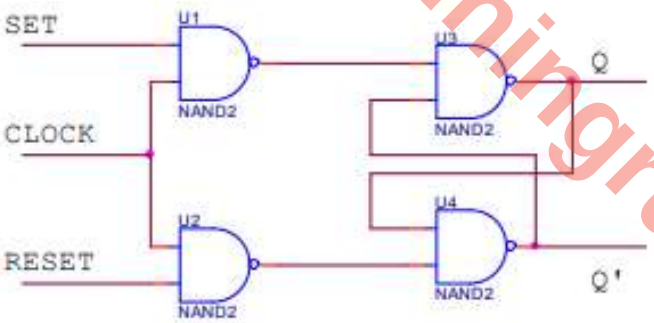
Gambar 12. 7 Bentuk – bentuk gelombang

Dari gambar 12.7 terlihat bahwa output FF tidak terpengaruh oleh sisi menuju negatip dari pulsa clock. Juga perhatikan bahwa level-level S dan R tidak mempunyai pengaruh terhadap FF kecuali pada saat terjadi transisi menuju positip dari pulsa clock. Input-input S dan R pada hakekatnya adalah input-input pengontrol, yang mengontrol ke keadaan mana output FF apabila terjadi pulsa clock. Clock input adalah trigger input, yang sesungguhnya menyebabkan berubahnya keadaan FF sesuai dengan level dari input-input S dan R. Gambar 12.8 menunjukkan symbol untuk sebuah Clocked SR FF yang CLK inputnya mendapat trigger pada saat transisi menuju negatip. Lingkaran kecil yang digambar pada CLK input menunjukkan bahwa FF ini akan mendapat trigger pada saat CLK berubah dari 1 ke 0.



Gambar 12. 8 Clocked SR Flip – Flop dengan pulsa clock aktif rendah

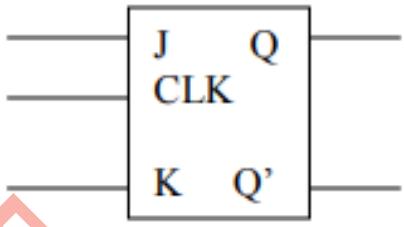
Rangkaian internal Clocked SR FF dalam kenyataannya sudah ada dalam bentuk IC, rangkaiannya terdiri dari dua bagian yaitu : 1. NAND latch yang disusun oleh NAND-3 dan NAND-4 86 2. Rangkaian pulsa yang disusun oleh NAND-1 dan NAND-2



Gambar 12. 9 Rangkaian Clocked SR Flip Flop

E. Clocked JK Flip Flop

Gambar 12.10 menunjukkan sebuah clocked JK FF yang ditrigger oleh sisi menuju positip dari pulsa clock. Input-input J dan K mengontrol keadaan FF dengan cara yang sama seperti input-input S dan R kecuali satu perbedaan utama : keadaan J = K = 1 tidak menghasilkan suatu output yang tidak menentu. Untuk keadaan ini FF akan selalu berada dalam keadaan yang berlawanan.



Gambar 12. 10 Clocked JK Flip Flop

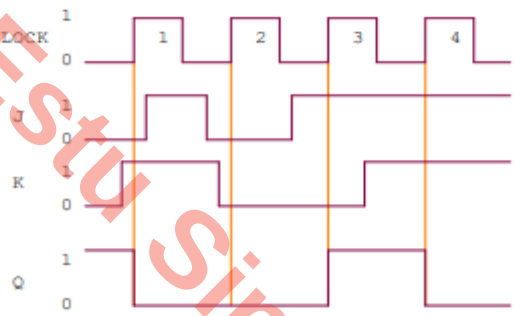
Tabel 12. 4 Tabel kebenaran

Input			Output		
S	R	C	Q	$\underline{Q}$	Comments
0	0	↑	Q	$\underline{Q}$	No change
0	1	↑	0	1	RESET
1	0	↑	1	0	SET
1	1	→	$\underline{Q}$	Q	Invalid

Bekerjanya FF ini ditunjukkan oleh bentuk gelombang pada gambar 10.11, yang dapat dianalisa sebagai berikut :

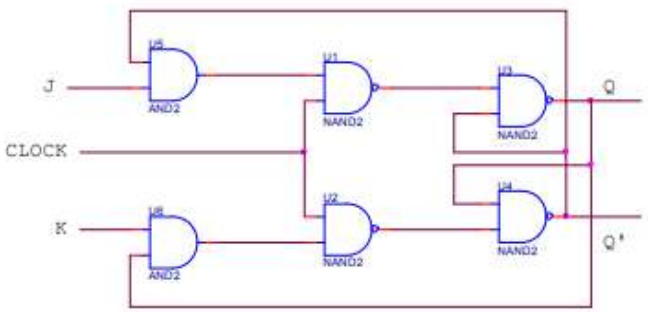
1. Mula-mula semua input adalah 0 dan output Q sama dengan 1.
2. Apabila terjadi sisi menuju positip dari pulsa clock pertama berlangsung pada kondisi J=0 dan K=1, maka output Q=0
3. Pulsa clock kedua mendapatkan J=0 dan K=0 pada saat melakukan transisi positipnya, ini menyebabkan output Q tetap pada kondisi sebelumnya yaitu Q=0. J Q CLK K Q′ Q' 87
4. Pulsa clock ketiga mendapatkan J=1 dan K=0 pada saat melakukan transisi positipnya, ini menyebabkan output Q=1.

5. Pulsa clock keempat mendapatkan J=1 dan K=1 pada saat melakukan transisi positipnya, ini menyebabkan FF toggle sehingga output Q berlawanan dari kondisi sebelumnya , yaitu menjadi Q=0.



Gambar 12. 11 Bentuk gelombang

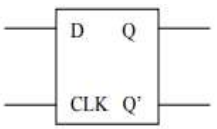
Dari bentuk gelombang ini hendaknya diperhatikan bahwa FF tidak terpengaruh oleh sisi menuju negatip dari pulsa clock. JK FF adalah jauh lebih baik dari pada SRFF karena tidak mempunyai keadaan kerja yang tidak menentu. Keadaan J=K=1, yang menghasilkan operasi toggle, sangat banyak ditemukan pemakaiannya di dalam semua jenis alat hitung biner. Oleh Karena itu, JKFF digunakan secara luas pada hampir semua sistem-sistem digital.



Gambar 12. 12 Rangkaian JK FF

F. Clocked D Flip Flop

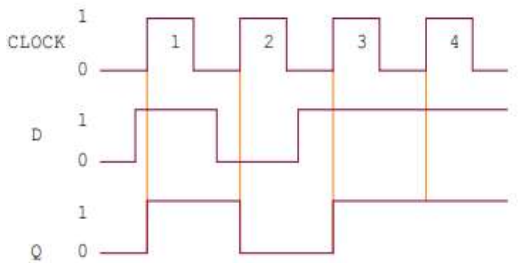
Gambar 12.13 menunjukkan symbol dari sebuah clocked D FF yang mendapat trigger dari transisi positip pada CLK inputnya. D input adalah suatu input pengontrol tunggal yang menentukan keadaan kerja FF sesuai dengan tabel kebenaran. Pada hakekatnya, ouput Q FF akan memasuki keadaan kerja yang sama dengan yang terdapat pada D input apabila terjadi suatu transisi positip pada CLK input. Perhatikanlah bahwa setiap terjadi transisi positip pada CLK inputnya, output Q memiliki harga yang sama seperti pada yang terdapat pada level D input. Transisi negatip pada CLK input tidak mempunyai pengaruh.



(a)

D	Q
0	Q = 0
1	Q = 1

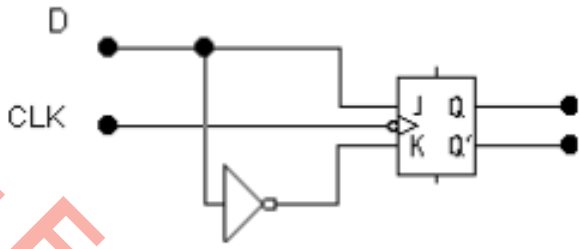
(b)



(c)

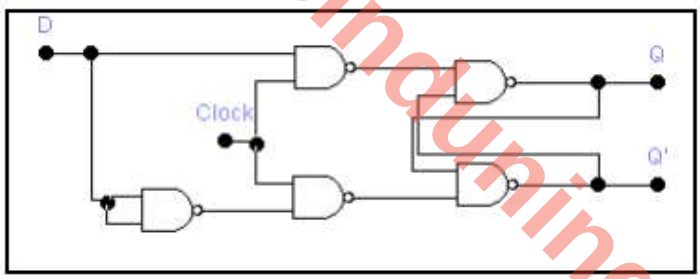
Gambar 12. 13 D FF yang ditrigger pada transisi menuju positif

D FF pada prinsipnya digunakan pada transfer data biner. SR FF dan JK FF dengan mudah dapat dimodifikasi untuk beroperasi sebagai D FF seperti ditunjukkan pada gambar 12.14



Gambar 12. 14 susunan JK FF yang bekerja sebagai D FF

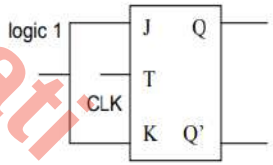
D FF juga dapat dibentuk dari NAND gate seperti ditunjukan pada gambar 12.15 berikut:



Gambar 12. 15 D FF yang disusun dari NAND gate

G. T Flip Flop

T FF dapat dibentuk dari modifikasi Cloked SR FF, D FF, maupun JK FF. Pada gambar di bawah ditunjukkan modifikasi JK FF yang digunakan sebagai T FF. Masukan J dan K pada JK FF dihubungkan dengan logika "1" atau dalam praktek dihubungkan dengan VCC +5 Volt, sedangkan sebagai masukan T FF adalah clock pada JK FF. Keadaan output akan Q berubah setiap ada pulsa clock



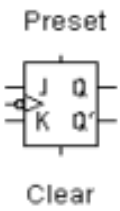
Tabel Kebenaran T FF

Input T	Present State Q	Next State Q*
0	0	0
1	0	1
0	1	1
1	1	0

Gambar 12. 16 T Flip Flop

H. Flip Flop Input Sinkron dan Asinkron

Untuk clocked flip-flop yang telah dipelajari, S, R, J, K, dan D telah dipandang sebagai input pengontrol. Input-input ini juga disebut input-input sinkron, karena pengaruhnya pada output FF disinkronkan dengan pulsa clock input. Hampir semua clocked FF juga mempunyai satu atau lebih input asinkron yang bekerja secara bebas dari input-input sinkron dan pulsa clock. Input-input asinkron ini dapat digunakanuntuk mengeset FF menuju keadaan 1 atau meng-clear FF menuju keadaan 0 pada setiap saat, tanpa mempedulikan keadaan pada input-input yang lain. Dengan kata lain, input-input asinkron tersebut merupakan input-input override (berkuasa) , yang dapat digunakan untuk melampaui input-input yang lain dengan maksud untuk menempatkan FF pada satu keadaan atau keadaan yang lain. Gambar 12.17 menunjukkan sebuah clocked JK FF dengan input Preset dan Clear. Input-input asinkron ini diaktifkan oleh suatu level 1, tabel kebenaran menunjukkan bagaimana bekerjanya input-input asinkron ini



Preset	Clear	FF Respons
0	0	Tak ada pengaruh pada FF
0	1	Meng-clear Q = 0
1	0	Mengeset Q = 1
1	1	Tak menentu

Gambar 12. 17 Clocked JK FF dengan input – input asinkron

SOAL – SOAL LATIHAN

1. Jelaskan perbedaan utama antara SR FF dengan JK FF
2. Jelaskan perbedaan atara input FF sinkron dengan asinkron
3. Terapkanlah bentuk gelombang J, K, dan CLK dari gambar 7.11 pada JK FF yang
4. bekerja saat pulsa clock bertransisi menuju negatip (aktif low).

BAB XIII
TEORI DAN PRAKTEK RANGKAIAN FLIP FLOP MENGGUNAKAN PROTEUS PROFESIONAL

A. RS FLIP – FLOP

Flip-flop adalah nama lain bagi multivibrator bistabil, yakni multivibrator yang keluarnya adalah suatu tegangan rendah atau tinggi 0 atau 1. Keluaran ini tetap rendah atau tinggi dan untuk mengubahnya, rangkaian yang bersangkutan harus didrive oleh suatu masukan yang disebut (trigger). Sampai datangnya pemicu, tegangan keluaran tetap rendah atau tinggi untuk selang waktu yang tak terbatas.

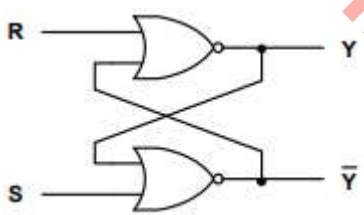
Tabel 13.1 meringkaskan kemungkinan-kemungkinan masukan/keluaran bagi flip-flop RS (Reset-Set) :

- Kondisi masukan yang pertama adalah RS = 0-0, Ini berarti tidak diterapkan pemicu. Dalam hal ini keluaran Y mempertahankan nilai terakhir yang dimilikinya.
- Kondisi masukan yang kedua adalah RS = 0-1 berarti bahwa suatu pemicu diterapkan pada masukan S (Set). Seperti kita ketahui, hal ini mengeset flip-flop dan menghasilkan keluaran Y bernilai 1.
- Kondisi masukan yang ketiga adalah RS = 1-0 ini menyatakan bahwa suatu pemicu diterapkan pada masukan R (Reset). Keluaran Y yang dihasilkan adalah 0.
- Kondisi masukan RS = 1-1 merupakan masukan terlarang. Kondisi ini berarti menerapkan suatu pemicu pada kedua masukan S dan R pada saat yang sama. Hal ini merupakan suatu pertentangan karena mengandung pengertian bahwa kita berupaya untuk memperoleh keluaran Y yang secara serentak sama dengan 1 dan sama dengan 0.

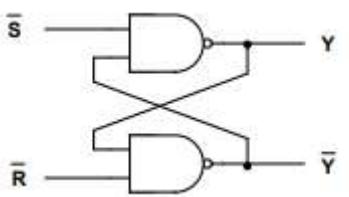
Tabel 13. 1 RS FLIP-FLOP

R	S	Y
0	0	Nilai terakhir
0	1	1
1	0	0
1	1	Terlarang

Keluaran masing-masing gerbang NOR mendrive salah satu masukan pada gerbang NOR yang lain. Demikian pula, masukan-masukan S dan R memungkinkan kita mengeset atau mereset keluaran Y. Seperti sebelumnya, masukan S yang tinggi mengeset Y ke 1; masukan R yang tinggi mereset Y ke 0. Jika R dan S kedua-duanya rendah, keluaran tetap tergrendel (latched) atau tertahan pada keadaan terakhirnya. Kondisi pertentangan yakni R dan S kedua-duanya tinggi pada saat yang sama juga masih terlarang.



Gambar 13. 1 Flip-flop RS dengan gerbang NOR



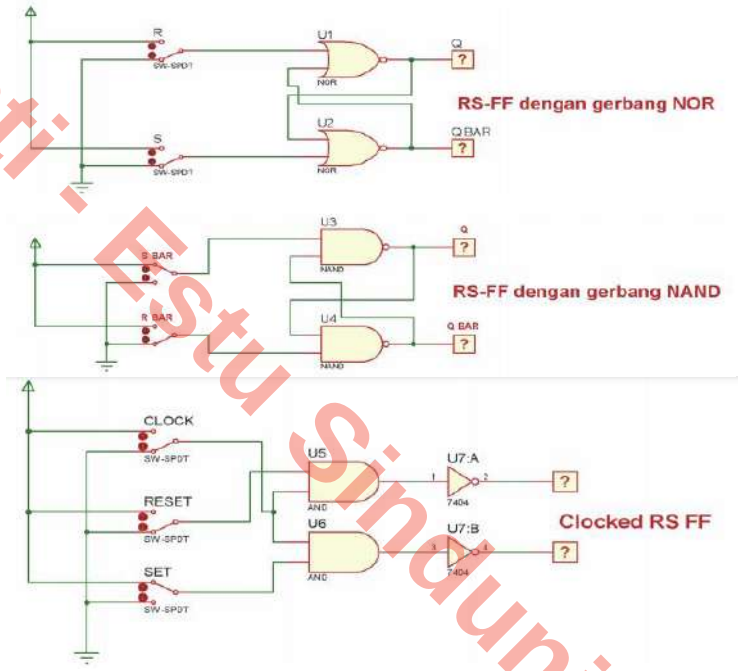
Gambar 13. 2 Flip-flop RS dengan gerbang NAND

Berbagai rancangan tingkat lanjutan dapat diwujudkan untuk menyempurnakan kecepatan perpindahan, impedansi keluaran, dan sebagainya.

Konsep Flip-flop RS yang harus diingat adalah sbb:

1. R dan S keduanya rendah berarti keluaran Y tetap berada pada keadaan terakhirnya secara tak terbatas akibat adanya aksi penggrendelan internal.
2. Masukan S yang tinggi mengeset keluaran Y ke 1, kecuali jika keluaran ini memang telah berada pada keadaan tinggi. Dalam hal ini keluaran tidak berubah, walaupun masukan S kembali ke keadaan rendah.
3. Masukan R yang tinggi mereset keluaran Y ke 0, kecuali jika keluaran ini memang telah rendah. Keluaran y selanjutnya tetap pada keadaan rendah, walaupun masukan R kembali ke keadaan rendah.
4. Memberikan R dan S keduanya tinggi pada saat yang sama adalah terlarang karena merupakan pertentangan (Kondisi ini mengakibatkan masalah pacu, yang akan dibahas kemudian).

Pengembangan lebih lanjut dari RS – FF adalah Clocked RS FF. Perbedaan cara kerja dari Clocked RS FF adalah bahwa flip –flop akan mengalami perubahan seperti pada RS FF menunggu sinyal clock aktif (logika tinggi).



Gambar 13. 3 Rangkaian simulasi RS-FF

Tabel 13. 2 Keyword dan komponen RS-FF

Nama Komponen	Keyword
7404	7404
Gerbang Logika AND	AND
Gerbang Logika NAND	NAND
Gerbang Logika NOR	NOR
Masukan	SW-SPDT
Keluaran	LOGICPROBE

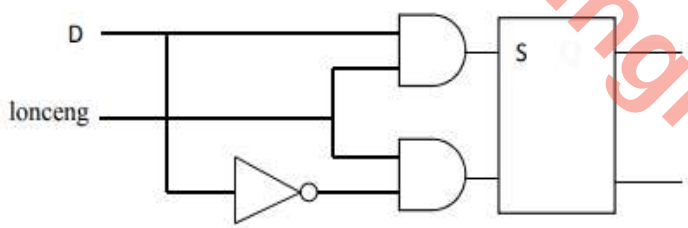
B. FLIP – FLOP

Flip-flop RS mempunyai dua masukan data, S dan R. Untuk menyimpan suatu bit tinggi, Anda membutuhkan S tinggi, untuk menyimpan bit rendah, Anda membutuhkan R tinggi. Membangkitkan dua buah sinyal untuk mendrive flip-flop merupakan suatu kerugian dalam berbagai penerapan. Demikian pula, kondisi terlarang yakni R dan S keduanya

tinggi dapat terjadi secara tidak sengaja. Hal ini telah membawa kita kepada flip-flop D (D dari *Data*), suatu rangkaian yang hanya membutuhkan sebuah masukan data.

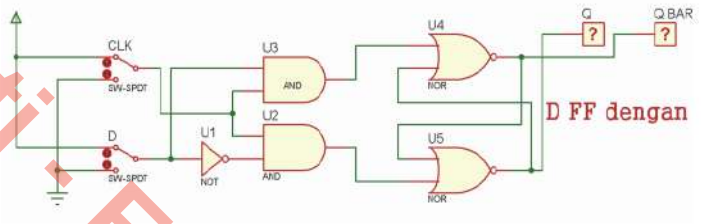
Gambar 13.4. memperlihatkan suatu cara sederhana untuk membangun sebuah flip-flop D. Jenis flip-flop ini mencegah nilai D mencapai keluaran Q sampai berlangsungnya pulsa lonceng. Cara kerja rangkaian yang bersangkutan cukup jelas, sebagai berikut. Bila lonceng adalah rendah, kedua gerbang AND tertutup; oleh karenanya D dapat berubah nilai tanpa mempengaruhi nilai Q. Sebaliknya, bila lonceng adalah tinggi, kedua gerbang AND terbuka. Dalam hal ini, Q terdorong untuk menyamai nilai D. Bila lonceng turun kembali, Q tak berubah dan menyimpan nilai D yang terakhir.

Terdapat berbagai cara untuk merancang flip-flop D. Pada dasarnya, flip-flop D merupakan multivibrator bistabil yang masukan D nya ditransfer ke keluaran setelah diterimanya sebuah pulsa lonceng.



Gambar 13. 4 Rangkaian D-FF

Untuk penggunaan praktis, kita dapat menggunakan IC 7474 yang berisi 2 buah *Positive-Edge-Triggered D Flip-Flop*. *Positive-Edge-Triggered* artinya nilai pada masukan kaki D akan diterima oleh Flip-Flop saat terjadi perubahan sinyal lonceng (clock) dari 0 ke 1 atau sering juga disebut *rising* edge. Perubahan masukan pada kaki D tidak akan berpengaruh pada keluaran Q bila tidak terjadi transisi pada lonceng dari 0 ke 1, walaupun misalnya lonceng bernilai 1.



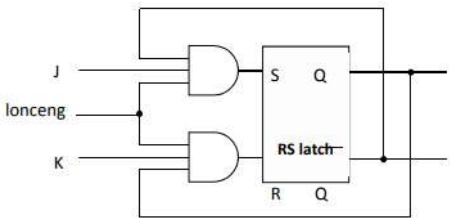
Gambar 13. 5 Rangkaian simulasi D-FF

Tabel 13. 3 Keyword Komponen

Nama Komponen	Keyword
7474	7474
Gerbang Logika AND	AND
Gerbang Logika NOR	NOR
Gerbang Logika NOT	NOT
Masukan	SW-SPDT
Keluaran	LOGICPROBE

C. JK- Flip – Flop

Gambar 13.6 memperlihatkan salah satu cara untuk membangun sebuah flip-flop J-K, J dan K disebut masukan pengendali karena menentukan apa yang dilakukan oleh flip-flop pada saat suatu pinggiran pulsa positif tiba.



Gambar 13. 6 Rangkaian flip-flop JK

Cara kerja rangkaian di atas dapat dijelaskan sebagai berikut.

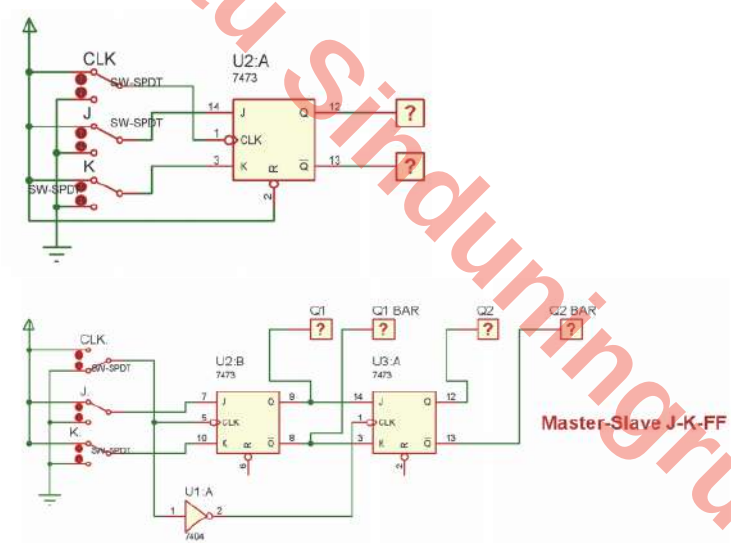
Pada saat J dan K keduanya 0, R dan S pasti bernilai 0 - 0, sehingga Q tetap pada nilai terakhirnya

- Pada saat J rendah dan K tinggi, gerbang atas tertutup (S bernilai 0), maka tidak terdapat kemungkinan untuk mengeset flip-flop. Bila Q tinggi (Q = 1) dan lonceng = 1, gerbang bawah (lonceng **AND** K **AND** Q) akan melewatkan pemicu reset (R = 1) yang akan menyebabkan Q menjadi rendah Jadi J = 0 dan K = 1 berarti lonceng = 1 akan mereset flip -flopnya (Q = 0), bila Q sebelumnya tinggi.
- Pada saat J tinggi dan K rendah, maka tidak terdapat kemungkinan untuk mereset flip-flop ((karena R pasti bernilai 0). Bila Q rendah (Q = 0 dan Q = 1) dan lonceng = 1, gerbang atas (lonceng **AND** J **AND** Q) akan melewatkan pemicu set (S = 1) yang akan menyebabkan Q menjadi tinggi, Jadi J = 1 dan K = 0 berarti lonceng = 1 akan mengeset flip-flopnya (Q = 1), bila Q sebelumnya rendah.
- Pada saat J dan K keduanya tinggi, dapat mengeset atau mereset flip -flopnya, tergantung kondisi Q sebelumnya. Bila Q tinggi (Q = 1) dan lonceng = 1, gerbang bawah akan melewatkan pemicu reset (R = 1) yang akan menyebabkan Q menjadi rendah.

Bila Q rendah (Q = 0) dan lonceng = 1, maka $\underline{Q}$= 1, gerbang atas akan melewatkan pemicu set (S = 1) yang akan menyebabkan Q menjadi tinggi. Jadi J = 1 dan K = 1 berarti bahwa pinggiran pulsa lonceng positif berikutnya akan membuat nilai Q yang baru adalah kebalikan dari nilai Q sebelumnya ($Q_{t+1} = \underline{Q_t}$)

Tabel 13. 4 FLIP-FLOP JK

CLK	J	K	Q
0	0	0	Keadaan terakhir
↑	0	1	0
↑	1	0	1
↑	1	1	$\overline{\text{Keadaan terakhir}}$



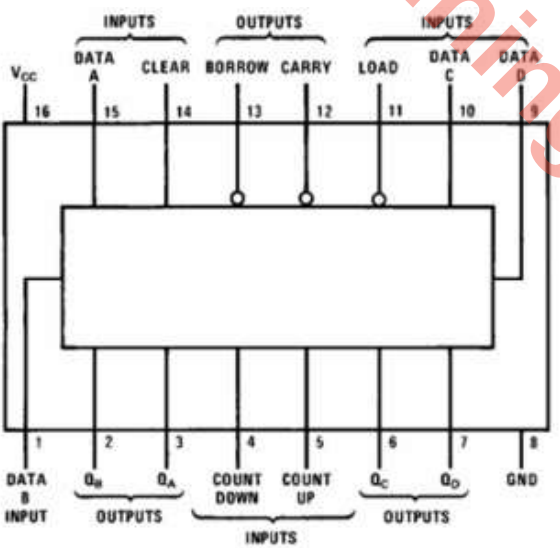
Gambar 13. 7 Rangkaian Simulasi JK-FF

Tabel 13. 5 Keyword Komponen

Nama Komponen	Keyword
7404	7404
7473	7473
Masukan	SW-SPDT
Keluaran	LOGICPROBE

D. Counter

Pencacah naik (atau kadang disebut pencacah maju) adalah pencacah yang urutan pencacahannya dari kecil ke besar, sedangkan sebaliknya pencacah turun (atau kadang disebut pencacah mundur) mencacah dari nilai tinggi ke rendah. Pada *labsheet* sebelumnya telah dijelaskan bagaimana menyusun pencacah naik dan pencacah turun dengan menggunakan serangkaian Flip-Flop. Untuk keperluan praktis, terdapat dua jenis IC pencacah naik dan turun yang sering digunakan yaitu IC 74192 dan 74193. IC counter 74192 adalah decade up/down counter yang mencacah dari nilai 0000 s/d 1001 biner atau 0 s/d 9 desimal. Sedangkan IC 74193 adalah IC *up/down counter* yang mencacah dari 0000 s/d 1111 biner atau 0 s/d 15 desimal. Pada Gambar 1 diperlihatkan diagram koneksi kaki IC 74192 yang disebut *Synchronous 4-Bit Up/Down Decade Counter* dalam lembar datanya.

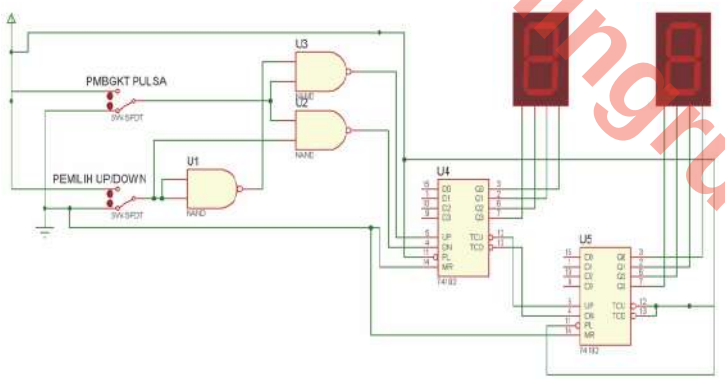


Gambar 13. 8 Diagram koneksi kaki-kaki IC 74193

Untuk memudahkan pemilihan operasi apakah pencacah naik atau pencacah turun maka dibuat suatu rangkaian kendali yang memanfaatkan gerbang-gerbang

logika. Dengan memanfaatkan sifat gerbang NAND, yaitu apabila salah satu input berlogika 0 maka output akan selalu 1, sehingga kondisi ini dapat mengunci output pada satu kondisi meskipun kondisi input kaki yang lain berubah-ubah. Dengan demikian rangkaian kendali *up/down counter* bisa direalisasikan. Rangkaian ini diperlihatkan pada Gambar Rangkaian 1.

Walaupun IC 74192 merupakan pencacah dekade, namun bila kita menginginkan untuk membentuk pencacah MOD-**n**, dengan **n** < 10, kita dapat mewujudkannya, seperti diperlihatkan pada gambar rangkaian 3. Adanya kaki *Carry Out* dan *Borrow Out* memungkinkan lebih dari satu IC 74192 dirangkai *cascade* untuk membentuk pencacah 0-99, 0-999, dan seterusnya, seperti diperlihatkan di gambar rangkaian 4. Selain itu IC 74192 juga mempunyai kaki masukan A, B, C, D, dan *Load* yang memungkinkan kita mempunyai nilai awal pencacah tertentu, tidak harus 0.



Gambar 13. 9 Rangakaian simulasi counter

Tabel 13. 6 Keyword dan komponen

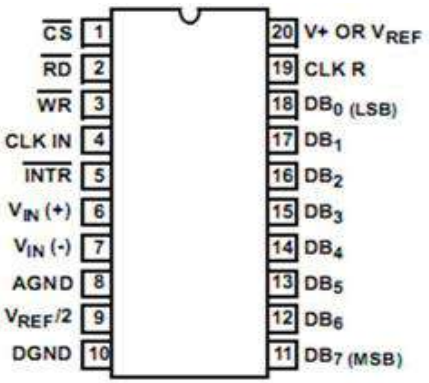
Nama Komponen	Keyword
7 Segment	7SEG-BCD
74192	74192
Masukan	SW-SPDT
Gerbang Logika Dasar NAND	NAND

E. Analog Digital Counter (ADC)

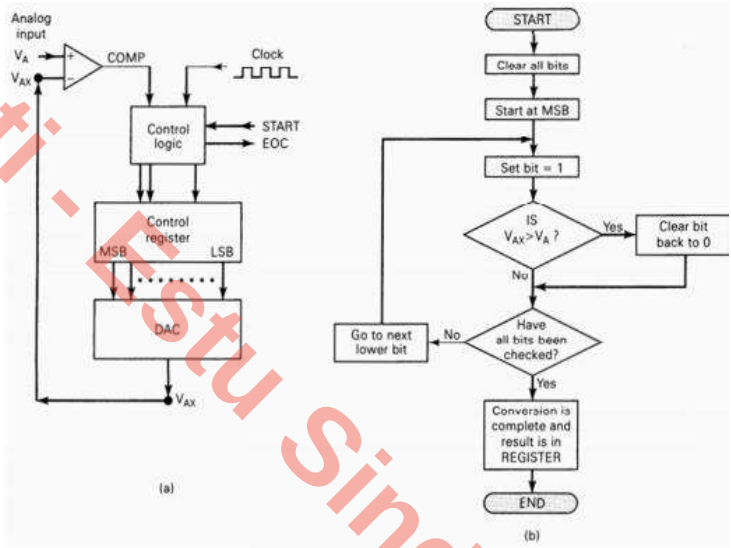
ADC adalah sebuah perangkat elektronik yang dirancang untuk mengubah sinyal-sinyal atau informasi yang bersifat analog menjadi sinyal-sinyal digital. Ada beberapa cara untuk mengubah sinyal analog menjadi sinyal digital, yaitu :

1. Succesive approximation,
2. Integration (single, dual, dan quad slope),
3. Counter comparator dan servo,
4. Paralel conversion,
5. Windows comparator, dan lain lain.

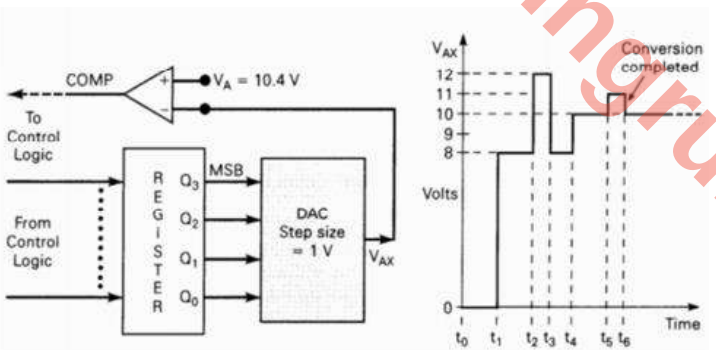
Rangkaian ADC ada yang sudah dikemas dalam satu chip IC, salah satu contohnya adalah ADC0804 yang sudah tersedia di trainer digital INEX. ADC0804 adalah ADC jenis CMOS 8 bit *succesive approximation*



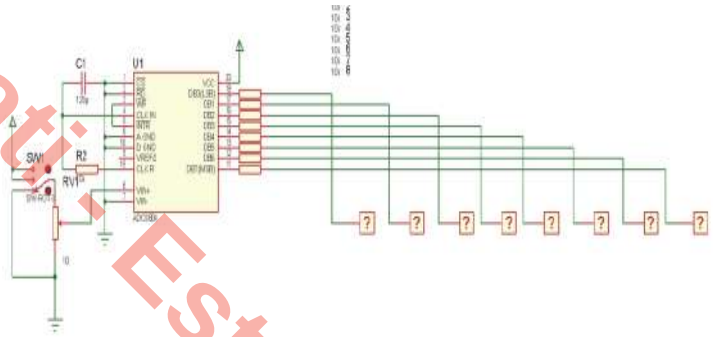
Gambar 13. 10 Diagram koneksi kaki-kaki IC ADC0804



Gambar 13. 11 DAC *Succesive-approxination* **(a) Diagram blok yang disederhanakan (b) Diagram alir cara kerja**



Gambar 13. 12 **Contoh operasi DAC *Succesive-approximation* dengan *step size* 1 V, V_A=10,4 V, dan keluaran digital $1011_2 = 11_{10}$**



Gambar 13. 13 Rangkaian Simulasi ADC

Tabel 13. 7 Keyword dan komponen

Nama Komponen	Keyword
IC ADC 0808	ADC0804
Resistor	RES
Masukan	SW-ROT-3
Variabel Resistor	POT-LIN
Keluaran	LOGICPROBE
Capasitor	CAP

Digital Analog Counter (DAC)

Karena kebanyakan metode konversi A/D menggunakan konversi D/A dalam proses konversinya, kita akan meninjau konversi D/A terlebih dahulu. DAC adalah suatu rangkaian elektronik yang berfungsi mengubah sinyal/data digital menjadi sinyal analog. Banyak sistem menerima data digital sebagai sinyal masukan dan kemudian mengubahnya menjadi tegangan atau arus analog. Data digital dapat disajikan dalam berbagai macam sandi/kode, yang paling lazim adalah dalam bentuk kode biner murni atau kode desimal dalam bentuk biner (*Binary Coded Desimal* / BCD).

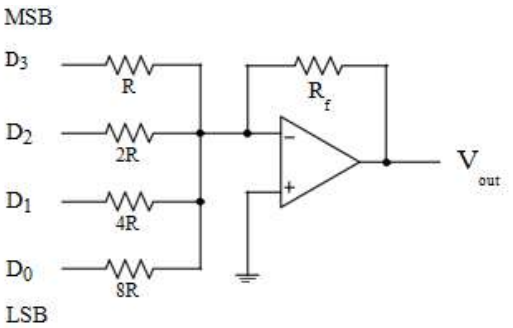
Keluaran V_{out} dari suatu DAC n bit diberikan oleh rumus:

$$V_{out} = (a_{n-1} \times 2^{-1} + a_{n-2} \times 2^{-2} + \ldots + a_0 \times 2^{-n}) \times V_{ref}$$

Koefisien-koefisien a di atas menggunakan kata biner, a = 1 atau 0, jika bit bit ke –n adalah 1 atau 0. Tegangan V_{ref} adalah tegangan acuan stabil yang digunakan dalam rangkaian. Bit paling berarti (*Most Significant Bit* / MSB) adalah bit yang bersesuaian dengan a_{n-1}, dan bobotnya adalah V_{ref} / 2, sedangkan bit paling tak berarti (LSB) bersesuaian dengan a_0, dan bobotnya sama dengan V_{ref} / (2n). Rangkaian DAC mempunyai banyak jenis dan tipe, salah satunya adalah DAC tipe tangga. Susunan tangga dalam rangkaian ini merupakan piranti pembagi arus, dan karena itu perbandingan hambatannya merupakan hal yang paling penting dari harga mutlaknya. Konfigurasi DAC tipe tangga adalah penguat jumlah, dengan R masukan yang naik 2^n kalinya.

$$V_{out} = -I \times R_f$$
$$= -(D_0/8R + D_1/4R + D_2/2R + D_3/R) \times V_{ref}$$

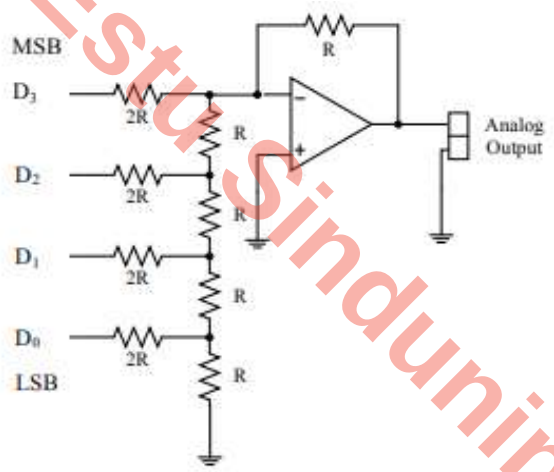
Logika digital diwujudkan dengan nilai tegangan D_0, D_1, D_2, D_3 = 0 Volt untuk logika "0" (*Low*) dan 5 Volt untuk logika "1" (*High*).



Gambar 13. 14 DAC tipe tangga

DAC yang lain adalah tipe R-2R seperti gambar berikut. Rangkaian DAC tipe ini lebih sederhana dan mudah dibangun karena nilai-nilai resistor yang digunakan dalam rangkaian hanya R dan 2R.

$V_{out} = -(D_3/2 + D_2/4 + D_1/8 + D_0/16)$



Gambar 13. 15 DAC tipe R-2R dengan penguat Op-Amp

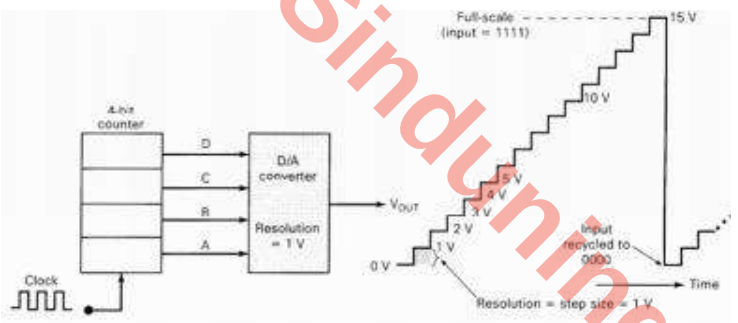
Resolusi dari sebuah DAC didefinisikan sebagai perubahan keluaran analog yang paling kecil yang bisa terjadi sebagai hasil perubahan pada input digital. Resolusi juga disebut ***step size***, karena mewakili besarnya perubahan di V_{out} seiring perubahan di masukan digital satu langkah demi langkah. Pada gambar di bawah ini, resolusi atau *step size* besarnya adalah 1 V. pada contoh tersebut, saat pencacah memberikan masukan 1111, maka keluaran DAC adalah 15 V, nilai ini disebut **keluaran skala-penuh** (*full-scale output*). Dengan demikian keluaran analog dari sebuah DAC dapat dirumuskan sebagai:

keluaran analog = *step size* x masukan digital

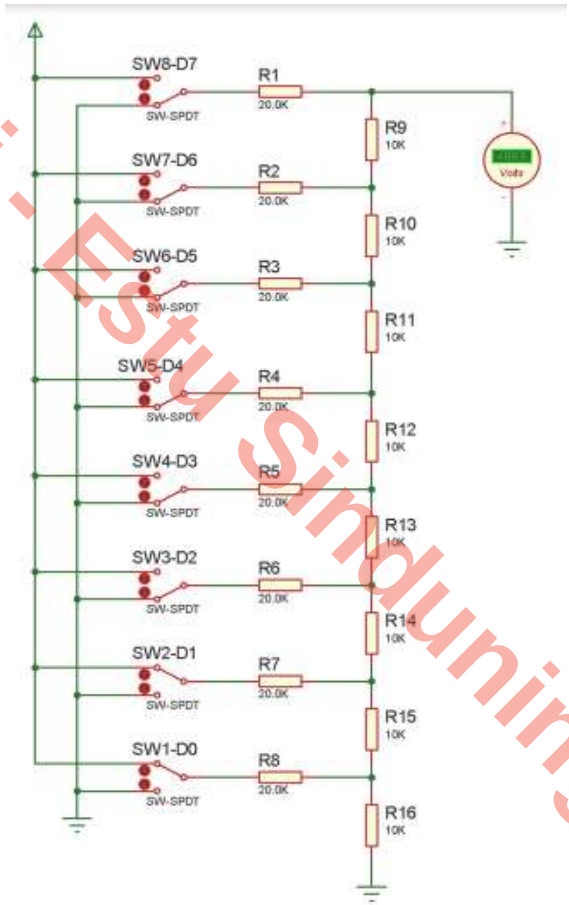
Cara lain untuk menghitung resolusi atau step size dari sebuah DAC adalah:

$$Resolusi = \frac{A_{fs}}{(2^n - 1)}$$

dengan A_{fs} adalah keluaran analog skala penuh dan n adalah cacah bit nilai digital. Untuk ADC, pada dasarnya resolusi (*step size*) dapat dihitung dengan cara yang sama, hanya sinyal analog adalah masukan dan sinyal digital adalah keluaran.



Gambar 13. 16 Keluaran dari DAC dengan masukan dari pencacah.



Gambar 13. 17 rangkaian Simulasi DAC

Tabel 13. 8 Keyword dan komponen

Nama Komponen	Keyword
Resistor	RES
Masukan	SW-SPDT

F. Digital Analog Counter (DAC)

Karena kebanyakan metode konversi A/D menggunakan konversi D/A dalam proses konversinya, kita akan meninjau konversi D/A terlebih dahulu. DAC adalah suatu rangkaian elektronik yang berfungsi mengubah

sinyal/data digital menjadi sinyal analog. Banyak sistem menerima data digital sebagai sinyal masukan dan kemudian mengubahnya menjadi tegangan atau arus analog. Data digital dapat disajikan dalam berbagai macam sandi/kode, yang paling lazim adalah dalam bentuk kode biner murni atau kode desimal dalam bentuk biner (*Binary Coded Desimal* / BCD).

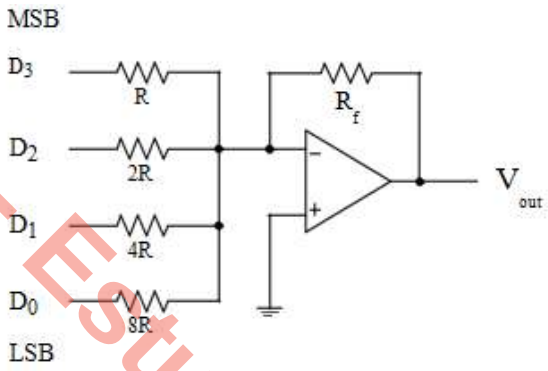
Keluaran V_{out} dari suatu DAC n bit diberikan oleh rumus:

$$V_{out} = (a_{n-1} \times 2^{-1} + a_{n-2} \times 2^{-2} + \ldots + a_0 \times 2^{-n}) \times V_{ref}$$

Koefisien-koefisien a di atas menggunakan kata biner, a = 1 atau 0, jika bit bit ke –n adalah 1 atau 0. Tegangan V_{ref} adalah tegangan acuan stabil yang digunakan dalam rangkaian. Bit paling berarti (*Most Significant Bit* / MSB) adalah bit yang bersesuaian dengan a_{n-1}, dan bobotnya adalah V_{ref} / 2, sedangkan bit paling tak berarti (LSB) bersesuaian dengan a_0, dan bobotnya sama dengan V_{ref} / (2n). Rangkaian DAC mempunyai banyak jenis dan tipe, salah satunya adalah DAC tipe tangga. Susunan tangga dalam rangkaian ini merupakan piranti pembagi arus, dan karena itu perbandingan hambatannya merupakan hal yang paling penting dari harga mutlaknya. Konfigurasi DAC tipe tangga adalah penguat jumlah, dengan R masukan yang naik 2^n kalinya.

$$V_{out} = -I \times R_f$$
$$= -(D_0/8R + D_1/4R + D_2/2R + D_3/R) \times V_{ref}$$

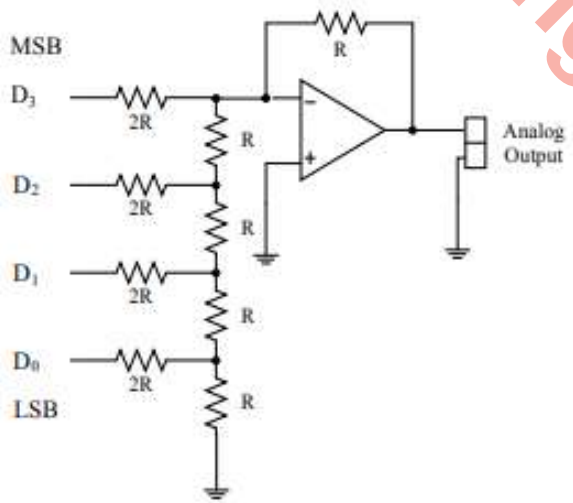
Logika digital diwujudkan dengan nilai tegangan D_0, D_1, D_2, D_3 = 0 Volt untuk logika "0" (*Low*) dan 5 Volt untuk logika "1" (*High*).



Gambar 13. 18 DAC tipe tangga

DAC yang lain adalah tipe R-2R seperti gambar berikut. Rangkaian DAC tipe ini lebih sederhana dan mudah dibangun karena nilai-nilai resistor yang digunakan dalam rangkaian hanya R dan 2R.

$V_{out} = -(D_3/2 + D_2/4 + D_1/8 + D_0/16)$



Gambar 13. 19 DAC tipe R-2R dengan penguat Op-Amp

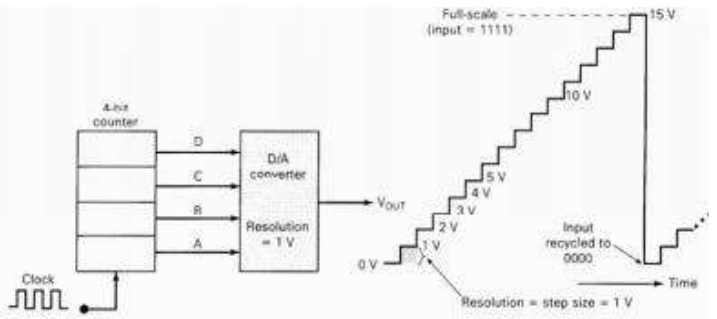
Resolusi dari sebuah DAC didefinisikan sebagai perubahan keluaran analog yang paling kecil yang bisa terjadi sebagai hasil perubahan pada input digital. Resolusi juga disebut ***step size***, karena mewakili besarnya perubahan di V_{out} seiring perubahan di masukan digital satu langkah demi langkah. Pada gambar di bawah ini, resolusi atau *step size* besarnya adalah 1 V. pada contoh tersebut, saat pencacah memberikan masukan 1111, maka keluaran DAC adalah 15 V, nilai ini disebut **keluaran skala-penuh** (*full-scale output*). Dengan demikian keluaran analog dari sebuah DAC dapat dirumuskan sebagai:

keluaran analog = *step size* x masukan digital

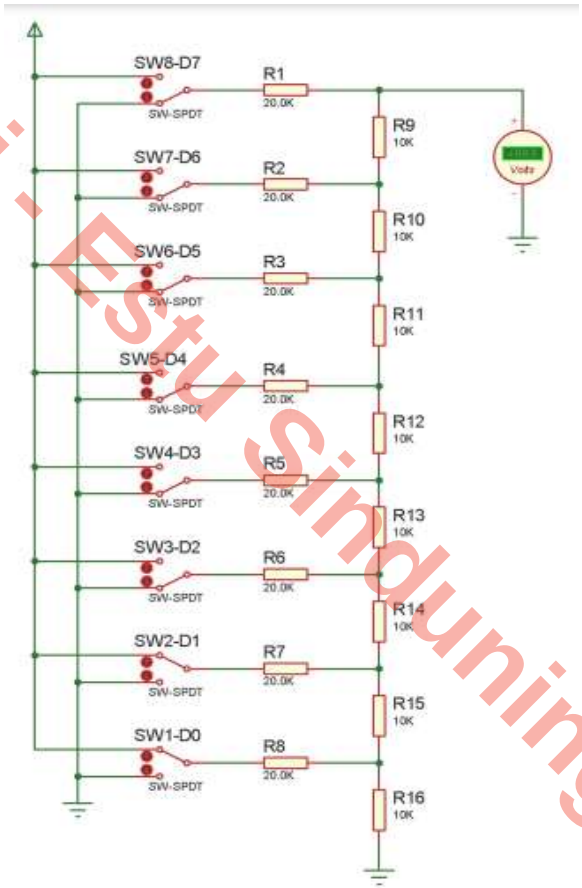
Cara lain untuk menghitung resolusi atau step size dari sebuah DAC adalah:

$$Resolusi = \frac{A_{fs}}{(2^n - 1)}$$

dengan A_{fs} adalah keluaran analog skala penuh dan n adalah cacah bit nilai digital. Untuk ADC, pada dasarnya resolusi (*step size*) dapat dihitung dengan cara yang sama, hanya sinyal analog adalah masukan dan sinyal digital adalah keluaran.



Gambar 13. 20 Keluaran dari DAC dengan masukan dari pencacah.



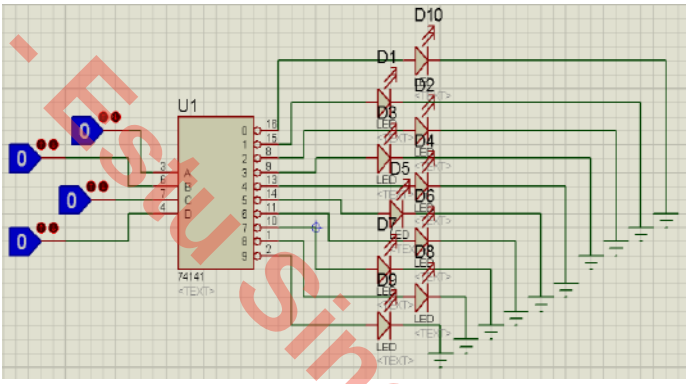
Gambar 13. 21 rangkaian Simulasi DAC

Tabel 13. 9 Keyword dan komponen

Nama Komponen	Keyword
Resistor	RES
Masukan	SW-SPDT

SOAL – SOAL LATIHAN

1. Ujilah Tabel Percobaan dari gambar dibawah ini



2. Isilah tabel Hasil Percobaan Kode BCD Ke Bilangan Desimal

Kode BCD				Bilangan Desimal
D	C	B	A	

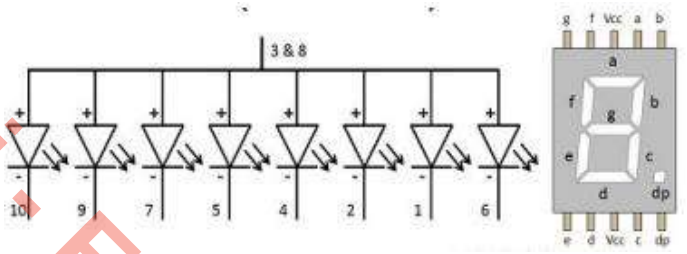
BAB XIV
PRAKTEK MENGGUNAKAN PROTEUS

A. SEVEN SEGMENTS

Salah satu jenis Seven segment Display yang sering digunakan adalah 7 Segmen yang menggunakan LED (Light Emitting Diode) sebagai penerangnya. LED 7 Segmen ini umumnya memiliki 7 Segmen atau elemen garis dan 1 segmen titik yang menandakan "koma" Desimal. Jadi Jumlah keseluruhan segmen atau elemen LED sebenarnya adalah 8. Cara kerjanya pun boleh dikatakan mudah, ketika segmen atau elemen tertentu diberikan arus listrik, maka Display akan menampilkan angka atau digit yang diinginkan sesuai dengan kombinasi yang diberikan. Terdapat 2 Jenis LED 7 Segmen yaitu "LED 7 Segmen common Cathode" dan "LED 7 Segmen common Anode".

1. LED 7 Segmen Tipe Common Cathode (Katoda)

   IC 7448 digunakan untuk driver dispaly 7 segment common cathode. Pada LED 7 Segmen jenis Common Cathode (Katoda), Kaki Katoda pada semua segmen LED adalah terhubung menjadi 1 Pin, sedangkan Kaki Anoda akan menjadi Input untuk masing-masing Segmen LED. Kaki Katoda yang terhubung menjadi 1 Pin ini merupakan Terminal Negatif (-) atau Ground sedangkan Signal Kendali (Control Signal) akan diberikan kepada masing-masing Kaki Anoda Segmen LED

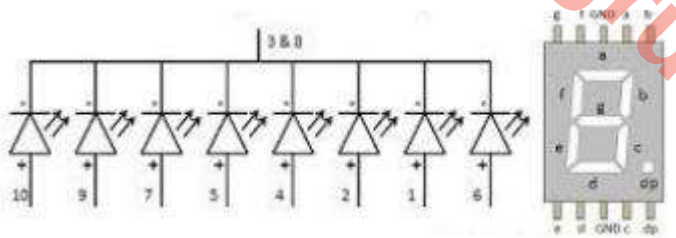


Gambar 14. 1 Jenis 7 Segmen Tipe Common Anoda

2. LED 7 Segmen Tipe Common Cathode (Katoda)

IC 7447 digunakan untuk driver 7 segment common anoda. Pada LED 7 Segmen jenis Common Anode (Anoda), Kaki Anoda pada semua segmen LED adalah terhubung menjadi 1

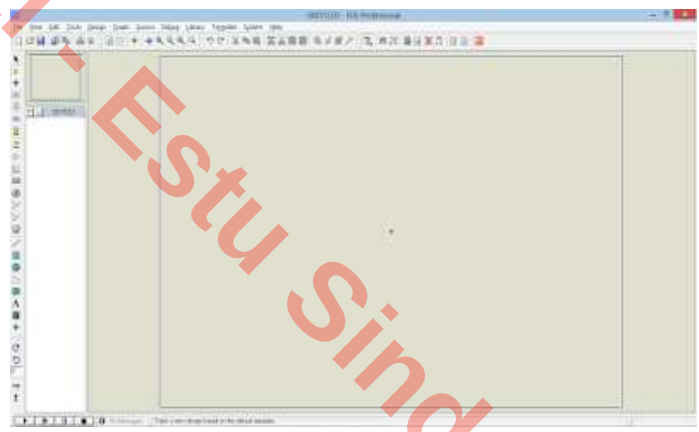
Pin, sedangkan kaki Katoda akan menjadi Input untuk masing-masing Segmen LED. Kaki Anoda yang terhubung menjadi 1 Pin ini akan diberikan Tegangan Positif (+) dan Signal Kendali (control signal) akan diberikan kepada masing-masing Kaki Katoda Segmen LED.



Gambar 14. 2 Jenis 7 Segmen Tipe Common Anoda

Adapun langkah- langkah untuk memulai praktek rangkaian Flip Flop sebagai berikut :

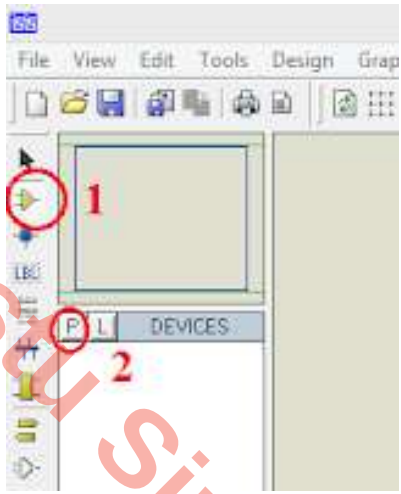
1. Buka Applikasi proteus professional



Gambar 14. 3 tampilan proteus 8.0

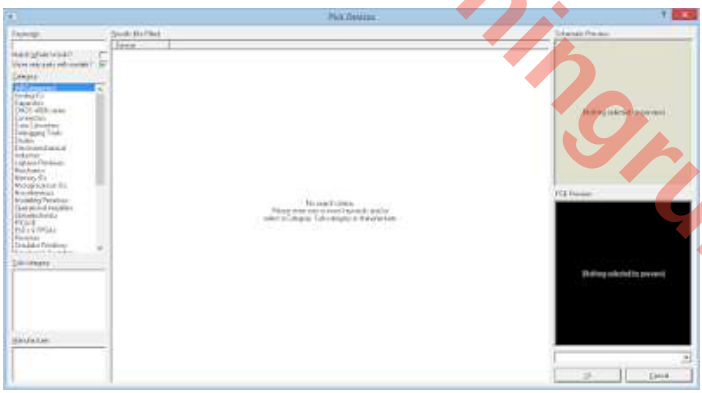
2. Selanjutnya, untuk membuat rangkaian Flip-Flop diperlukan komponen:
   - IC 7448
   - 7 segment Katoda
   - 7 segment anoda
   - Logicstate

3. Untuk dapat mencari komponen-komponen diatas dengan cara mengarahkan kursor kepada Toolbar dan memilih bagian Component Mode, kemudian pilihlah symbol "P".



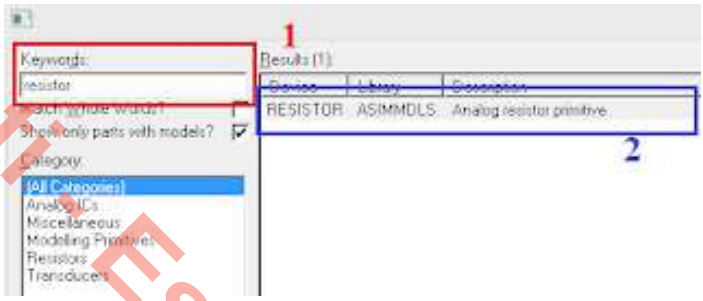
Gambar 14. 4 untuk pencarian komponen

4. Maka akan muncul sebuah kontak dialog seperti ini



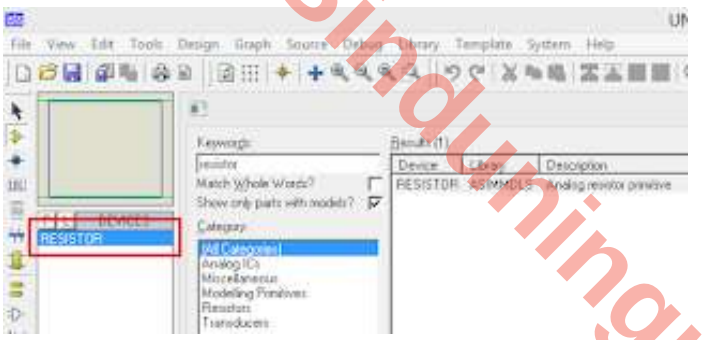
Gambar 14. 5 dialog komponen

5. Selanjutnya, tuliskanlah komponen yang dibutuhkan pada bagian Keyword dan double-klik pada bagian result sesuai dengan komponen yang dibutuhkan.



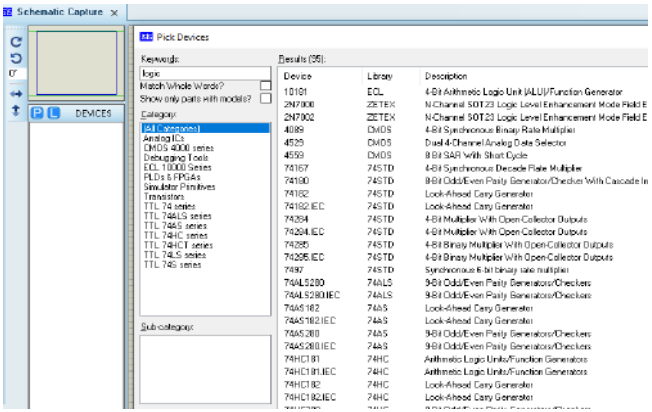
Gambar 14. 6 pencarian komponen

6. Maka dapat melihat bahwa komponen yang kita pilih masuk kedalam list devices.



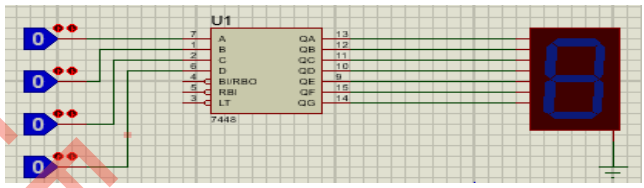
Gambar 14. 7 tipe – tipe komponen

7. Selanjutnya, lakukanlah hal yang sama untuk komponen lainnya

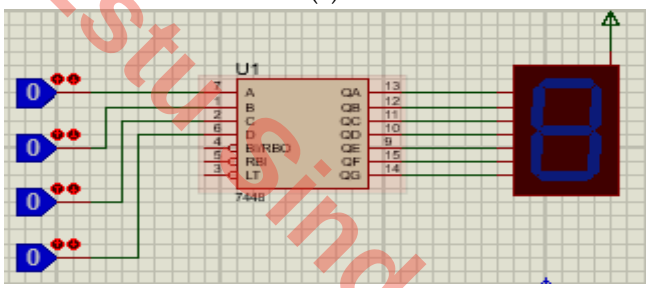


Gambar 14. 8 library komponen

8. Buat rangkaian skematik sesuai dengan gambar dibawah ini



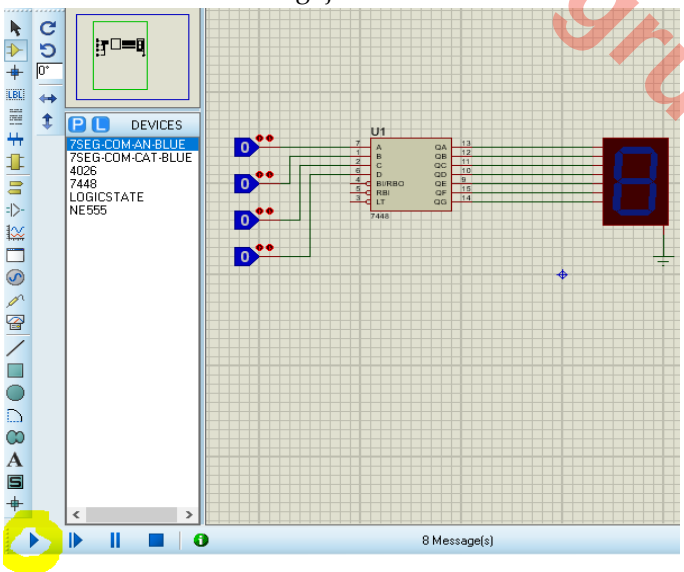
(a)



(b)

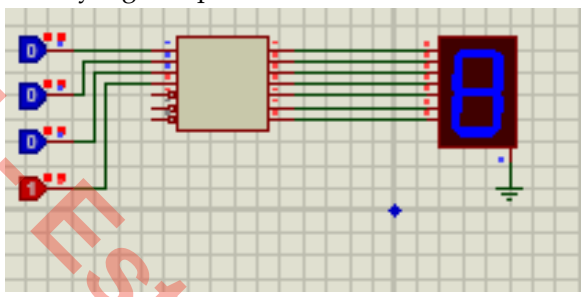
Gambar 14. 9 rangkaian skematik 7segment katoda dan anoda

9. Maka klik run untuk menguji coba

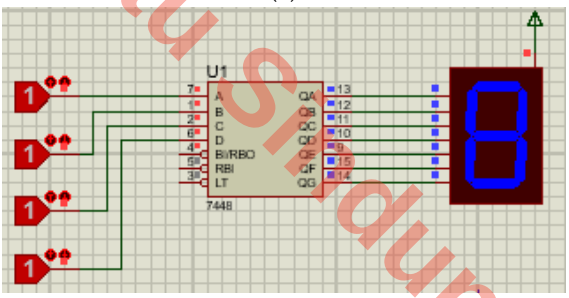


Gambar 14. 10 run simulasi

10. Maka hasil yang didapat adalah



(a)



(b)

Gambar 14. 11 hasil run dengan 7segment katoda (a) dan anoda (b)

Tabel 14. 1 hasil run 7 segment katoda

A	B	C	D	OUTPUT
0	0	0	0	
0	0	0	1	
0	0	1	0	
0	0	1	1	
0	1	0	0	
0	1	0	1	
0	1	1	0	
0	1	1	1	
1	0	0	0	
1	0	0	1	
1	0	1	0	
1	0	1	1	

1	1	0	0	
1	1	0	1	
1	1	1	0	
1	1	1	1	

B. Counter

Counter adalah sebuah rangkaian sekuensial yang mengeluarkan urutan statestate tertentu, yang merupakan aplikasi dari pulsa-pulsa inputnya. Pulsa input dapat berupa pulsa clock atau pulsa yang dibangkitkan oleh sumber eksternal dan muncul pada interval waktu tertentu. Counter banyak digunakan pada peralatan yang berhubungan dengan teknologi digital, biasanya untuk menghitung jumlah kemunculan sebuah o kejadian/event atau untuk menghitung pembangkit waktu. Counter yang mengeluarkan urutan biner dinamakan Biner Counter. Sebuah n-bit binary counter terdiri dari n buah flip-flop, dapat menghitung dari 0 sampai 2n - 1 . Counter secara umum diklasifikasikan atas counter asyncron dan counter syncronous.

1. Counter Asyncronous

   Counter Asyncronous disebut juga Ripple Through Counter atau Counter Serial (Serial Counter), karena output masing-masing flip-flop yang digunakan akan bergulingan (berubah kondisi dan "0" ke "1") dan sebaliknya secara berurutan atau langkah demi langkah, hal ini disebabkan karena hanya flipflop yang paling ujung saja yang dikendalikan oleh sinyal clock, sedangkan sinyal clock untuk flip-flop lainnya diambilkan dan masing-masing flipflop sebelumnya.

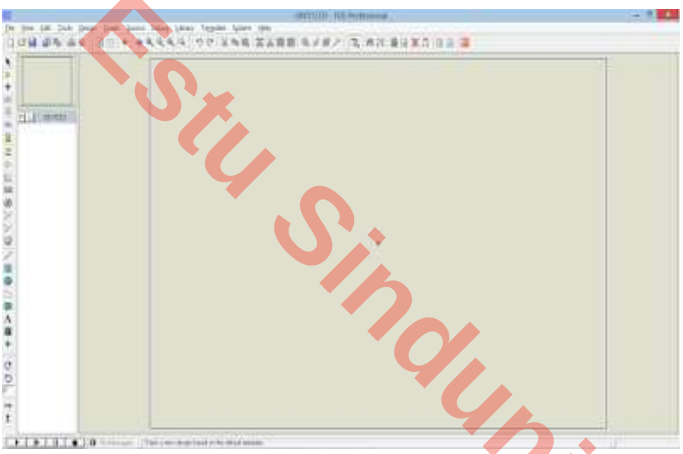
2. Counter Syncronous

   Counter syncronous disebut sebagai Counter parallel, output flip-flop yang digunakan bergulingan secara serempak. Hal mi disebabkan karena masingmasing

flip- flop tersebut dikendalikan secara serempak oleh sinyal clock

Adapun langkah- langkah untuk memulai praktek rangkaian counter sebagai berikut :

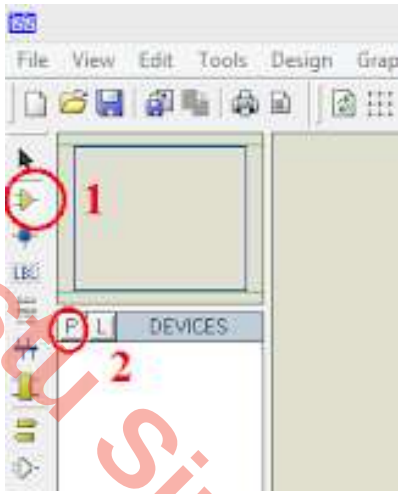
1. Buka Applikasi proteus professional



Gambar 14. 12 tampilan proteus 8.0

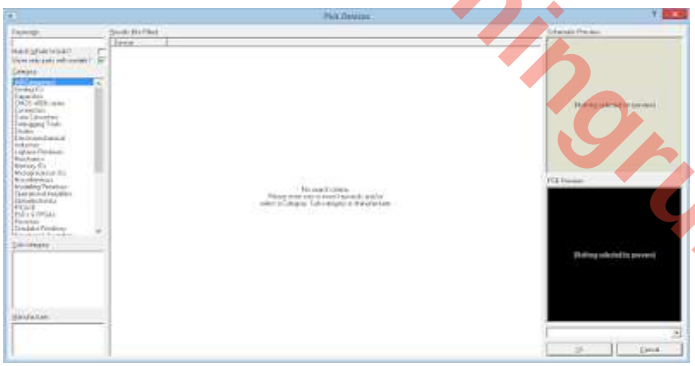
2. Selanjutnya, untuk membuat rangkaian Flip-Flop diperlukan komponen:
   - Led
   - JK-FF
   - AND
   - Switch
   - Function Generator
   - Osiloscop

3. Untuk dapat mencari komponen-komponen diatas dengan cara mengarahkan kursor kepada Toolbar dan memilih bagian Component Mode, kemudian pilihlah symbol "P".



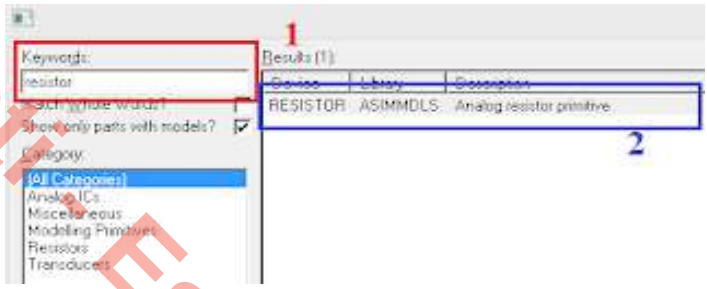
Gambar 14. 13 untuk pencarian komponen

4. Maka akan muncul sebuah kontak dialog seperti ini



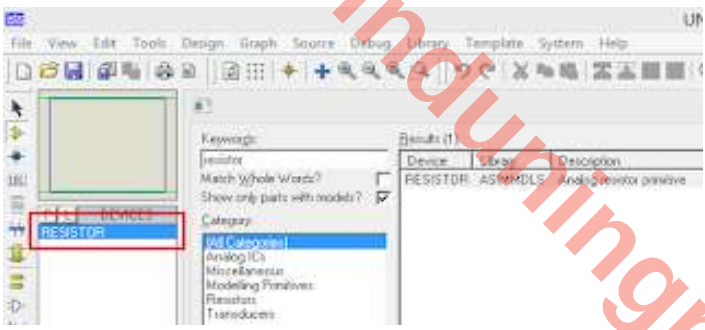
Gambar 14. 14 dialog komponen

5. Selanjutnya, tuliskanlah komponen yang dibutuhkan pada bagian Keyword dan double-klik pada bagian result sesuai dengan komponen yang dibutuhkan.



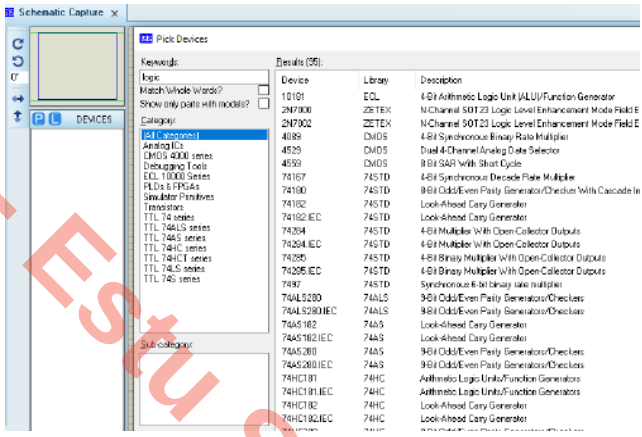
Gambar 14. 15 pencarian komponen

6. Maka dapat melihat bahwa komponen yang kita pilih masuk kedalam list devices.



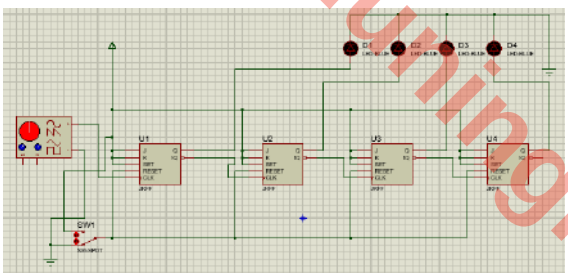
Gambar 14. 16 tipe – tipe komponen

7. Selanjutnya, lakukanlah hal yang sama untuk komponen lainnya

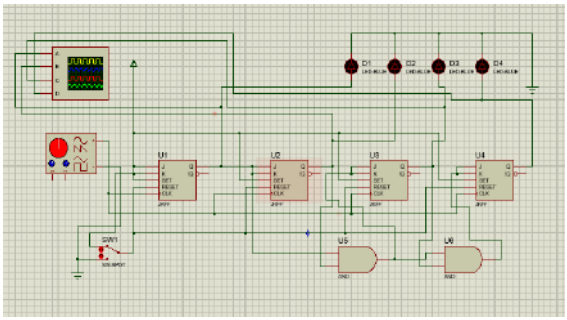


Gambar 14. 17 library komponen

8. Buat rangkaian skematik sesuai dengan gambar dibawah ini



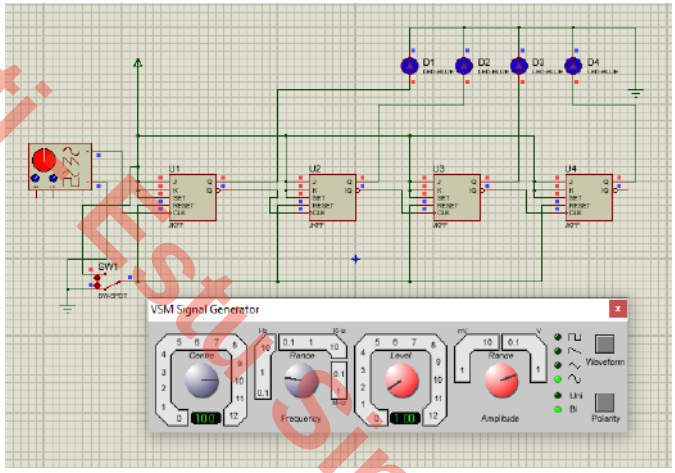
(a)



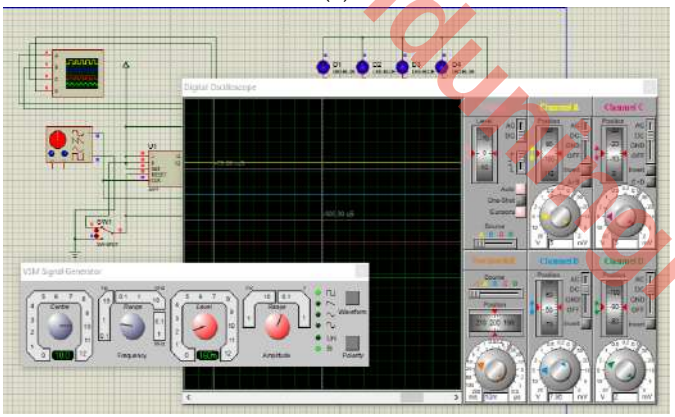
(b)

Gambar 14. 18 rangkaian skematik Counter Asyncronous dan rangkaian counter Syncronous

9. Hasil simulasi dari rangkaian tersebut



(a)

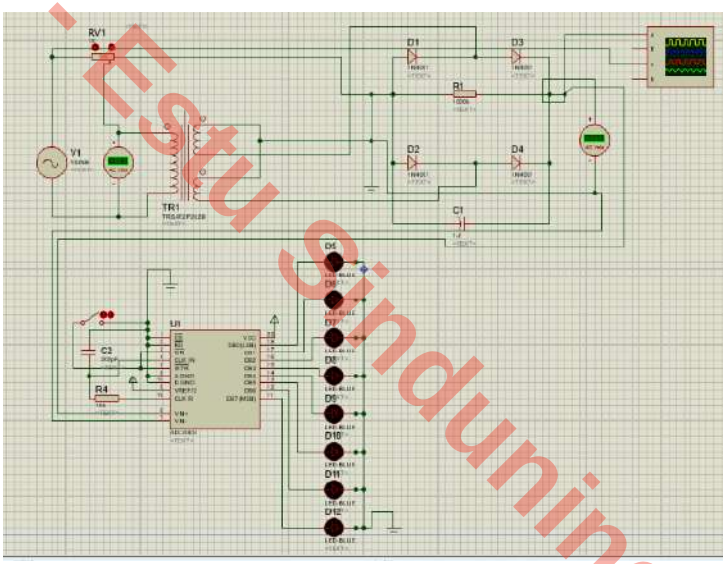


(b)

Gambar 14. 19 hasil simulasi antara rangkaian Counter Asyncronous dan Syncronous

SOAL SOAL LATIHAN

Buatlah rangkaian dibawah ini menggunakan proteus professional dan berikan hasil simulasi beserta penjelasannya



DAFTAR PUSTAKA

Estu Sinduningrum.2020. Teori dan praktek rangkaian Digital dan gelombang. Deepublish

Muhamad Ali, M.T., Andik Asmara, S.Pd. 2013. Teknik Digital dengan software simulasi Proteus

Ali, Muhammad. 2013.Modul Praktik Teknik Digital Dengan Sofeware Simulasi Proteus.Universitas Negeri Yogyakarta: Yogyakarta. Diunduh 24 april 2017

Assa'idah dan Yulinar Adnan.2009. Investigasi Terhadap Kemampuan 2 Tipe ADC. Universitas Sriwijaya: Palembang. Diunduh 24 april 2017

Defatta, J David.dkk. 1995. Digital Signal Processing. John Willey and Sons:Singapore

Hariyanto, Didik. 2013.Analog to Digital Converter. Universitas Negeri Yogyakarta: Yogyakarta. Diunduh 24 april 2017

Malvino, Paul Albert.1988. Elektronika Komputer Digital. Erlangga: Bandung

Permana, Jaka.2011. Makalah Gerbang Logika, (STIMIK) Tunas Bangsa: Banjarnegara. Diunduh 24 april 2017

Thokeim, L Roger. 1995. Elektronika Digital. Erlangga: Jakarta

Sarjana, dkk.2015. Bahan Ajar Praktek Perancangan Rangkaian Digital. Politeknik Negeri Sriwijaya: Palembang

Sutanto.1997. Rangkaian Elektronika Analog dan Terpadu. Universitas Indonesia: Jakarta

Suzansefi, Rapiko Duri.2015. Bahan Ajar Elektronika Digital. Politeknik Negeri Sriwijaya: Palembang

About collaboration Autor's

Irawati lahir di Bogor. Menyelesaikan pendidikan D-3 Teknik Elektronika di Politeknik Swadharma, S-1 Teknik Elektronika di Universitas Mercubuana, dan S-2 Magister Teknik Elektro Konsentrasi Telekomunikasi di Universitas Mercubuana. Saat ini, menjadi dosen tetap dan homebase Jurusan Teknik Elektronika di Institut Teknologi dan Bisnis Swadharma. Ira membuat buku sesuai dengan matakuliah yang pernah diajarkan dibidang Teknik infomatika serta Teknik elektronika sebagai penunjang perkuliahan, serta mempermudah mahasiswa dalam pembelajaran. Buku pertamanya diterbitkan pada tahun 2020 hingga sekarang.

Estu Sinduningrum lahir di Jakarta. Menyelesaikan pendidikan D-3 Teknik Telekomunikasi di Politeknik Negeri Jakarta (PNJ), S-1 Teknik Elektronika di Universitas Indonesia, dan S-2 Magister Teknik Elektro Konsentrasi Manajemen Telekomunikasi di Institut Teknologi Telkom (Bandung). Saat ini, menjadi dosen tetap dan homebase Jurusan Teknik di Universitas Muhammadiyah Prof. Dr. HAMKA. Estu memfokuskan dalam membuat buku sesuai dengan matakuliah yang pernah diajarkan dibidang Teknik infomatika serta Teknik elektro sebagai penunjang perkuliahan, serta mempermudah mahasiswa dalam pembelajaran. Buku pertamanya diterbitkan pada tahun 2014 hingga sekarang.

PENGANTAR TEKNIK DIGITAL MENGGUNAKAN PROTEUS PROFESIONAL

Pengantar Teknik Digital menggunakan Proteus Profesional merupakan salah satu mata kuliah wajib yang harus diketahui, dipahami dan dipraktikan bagi mahasiswa khususnya Teknik Elektro dan Teknik Informatika. Proteus merupakan software aplikasi yang multifungsi, selain untuk aplikasi simulasi bisa juga untuk layout PCB secara otomatis dari rangkaian skematik.

Buku ini membahas mengenai materi teknik digital secara teori dan juga bisa dipraktik dengan menggunakan software proteus. Secara garis besar yang dibahas di dalam buku ini, yaitu: **Konsep Dasar Teknik Digital, Sistem Bilangan, Gerbang Logika Dasar, Aljabar Boolean serta Teori De Morgan, Penyederhanaan Rangkaian Digital, Rangkaian Flip-Flop, Seven Segment, Rangkaian Counter, dan Rangkaian Register**.

ISBN 978-623-315-999-9

PENGANTAR TEKNIK DIGITAL MENGGUNAKAN PROTEUS PROFESIONAL | Irawati | Estu Sinduningrum