REPUBLIK INDONESIA
KEMENTERIAN HUKUM DAN HAK ASASI MANUSIA

SURAT PENCATATAN
CIPTAAN

Dalam rangka pelindungan ciptaan di bidang ilmu pengetahuan, seni dan sastra berdasarkan Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta, dengan ini menerangkan:

Nomor dan tanggal permohonan	:	EC00202107105, 19 Januari 2021
Pencipta		
Nama	:	**Dr. Muchlas, M.T**
Alamat	:	Gedongan Baru, Pelemwulung RT/RW 007, Banguntapan, Banguntapan, Bantul, DI Yogyakarta , Bantul, DI YOGYAKARTA, 55198
Kewarganegaraan	:	Indonesia
Pemegang Hak Cipta		
Nama	:	**UNIVERSITAS AHMAD DAHLAN**
Alamat	:	Kampus 2 Unit B Jl. Pramuka 5F, Pandeyan, Umbulharjo, Yogyakarta, DI Yogyakarta , Yogyakarta, DI YOGYAKARTA, 55161
Kewarganegaraan	:	Indonesia
Jenis Ciptaan	:	**Buku**
Judul Ciptaan	:	**Simulator Breadboard Perangkat Pembelajaran Teknik Digital**
Tanggal dan tempat diumumkan untuk pertama kali di wilayah Indonesia atau di luar wilayah Indonesia	:	1 November 2020, di Yogyakarta
Jangka waktu pelindungan	:	Berlaku selama 50 (lima puluh) tahun sejak Ciptaan tersebut pertama kali dilakukan Pengumuman.
Nomor pencatatan	:	000237002

adalah benar berdasarkan keterangan yang diberikan oleh Pemohon.
Surat Pencatatan Hak Cipta atau produk Hak terkait ini sesuai dengan Pasal 72 Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta.

a.n. MENTERI HUKUM DAN HAK ASASI MANUSIA
DIREKTUR JENDERAL KEKAYAAN INTELEKTUAL

Dr. Freddy Harris, S.H., LL.M., ACCS.
NIP. 196611181994031001

Simulator Breadboard

Perangkat Pembelajaran Teknik Digital

Dr. Muchlas, M.T.

Universitas Ahmad Dahlan
Indonesia

UAD PRESS

Simulator *Breadboard*

Perangkat Pembelajaran Teknik Digital

Oleh:

Dr. Muchlas, M. T.
(Universitas Ahmad Dahlan, Indonesia)

UAD PRESS

Sanksi Pelanggaran Pasal 113
Undang-Undang Nomor 28 Tahun 2014
Tentang Hak Cipta

1. Setiap orang yang dengan tanpa hak melakukan pelanggaran hak ekonomi sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf i untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 1 (satu) tahun dan/atau pidana denda paling banyak Rp100.000.000 (seratus juta rupiah).
2. Setiap orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf c, huruf d, huruf f, dan/atau huruf h untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 3 (tiga) tahun dan/atau pidana denda paling banyak Rp500.000.000,00 (lima ratus juta rupiah).
3. Setiap orang yang dengan tanpa hak dan/atau tanpa izin Pencipta atau pemegang Hak Cipta melakukan pelanggaran hak ekonomi Pencipta sebagaimana dimaksud dalam Pasal 9 ayat (1) huruf a, huruf b, huruf e, dan/atau huruf g untuk penggunaan secara komersial dipidana dengan pidana penjara paling lama 4 (empat) tahun dan/atau pidana denda paling banyak Rp1.000.000.000,00 (satu miliar rupiah).
4. Setiap orang yang memenuhi unsur sebagaimana dimaksud pada ayat (3) yang dilakukan dalam bentuk pembajakan, dipidana dengan pidana penjara paling lama 10 (sepuluh) tahun dan/atau pidana denda paling banyak Rp. 4.000.000.000,00 (empat miliar rupiah).

Simulator *Breadboard*

Perangkat Pembelajaran Teknik Digital

Oleh:

Dr. Muchlas, M. T.
(Universitas Ahmad Dahlan, Indonesia)

UAD PRESS

Simulator *Breadboard*: Perangkat Pembelajaran Teknik Digital

ISBN: 978-602-0737-80-5
16 x 24 cm, viii + 112 hlm
Cetakan Pertama, Desember 2020

Penulis:
Dr. Muchlas, M. T. (Universitas Ahmad Dahlan, Indonesia)

Editor: Budi Ashari
Layout: Ratih Purwandari

Diterbitkan oleh:
UAD PRESS
Anggota IKAPI dan APPTI

Alamat Penerbit:
Kampus II Universitas Ahmad Dahlan
Jl. Pramuka No.42, Pandeyan, Kec. Umbulharjo,
Kota Yogyakarta, Daerah Istimewa Yogyakarta 55161
E-mail: uadpress@uad.ac.id
HP/WA: 088239499820

Prakata

Seiring dengan perkembangan teknologi informasi dan komunikasi, saat ini, *trend* penggunaan laboratorium virtual semakin meningkat. Walaupun terdapat kelemahan-kelemahan dibandingkan laboratorium yang bersifat *hands-on,* namun penggunaan laboratorium virtual dapat meningkatkan efisiensi pendanaan dan fleksibilitas dalam pelaksanaannya. Salah satu perangkat laboratorum virtual yang mampu menyediakan alat dan bahan virtual yang diperlukan dalam mendukung praktikum teknik digital adalah simulator *breadboard.* Selain itu, simulator ini juga dapat melakukan simulasi watak dari rangkaian-rangkaian digital yang telah disusun menggunakan *breadboard* dan komponen-komponen virtual.

Buku ini mendeskripsikan secara lengkap cara pengoperasian simulator *breadboard* yang didahului dengan penjelasan penggunaan papan rangkaian tersebut dalam bentuk *real*. Uraian dalam buku ini dapat digunakan oleh dosen dan mahasiswa di lingkungan program studi teknik elektro atau program studi-program studi serumpunnya, sebagai acuan dalam menyiapkan praktikum teknik digital secara virtual. Selain itu, buku ini juga dapat digunakan oleh instruktur, guru, dan siswa sekolah menengah atas maupun sekolah menengah kejuruan (SMK) sebagai salah satu rujukan dalam membantu mempelajari dasar-dasar teknik digital.

Ucapan terima kasih dan penghargaan yang setinggi-tingginya diberikan kepada Dr. Chris Bailey dan Dr. Michael Freeman dari University of York, United Kingdom atas kontribusi tulisannya pada Bagian III yakni Kelengkapan Simulator *Breadboard* dan Bagian IV, yakni Penyusunan dan Simulasi Rangkaian Logika.

Dengan berbagai keterbatasannya, buku ini diharapkan dapat digunakan secara efektif sebagai sarana untuk melatih keterampilan peserta didik dalam menggunakan simulator *breadboard.*

Yogyakarta, November 2020

Penyusun

Muchlas

Daftar Isi

Bagian I

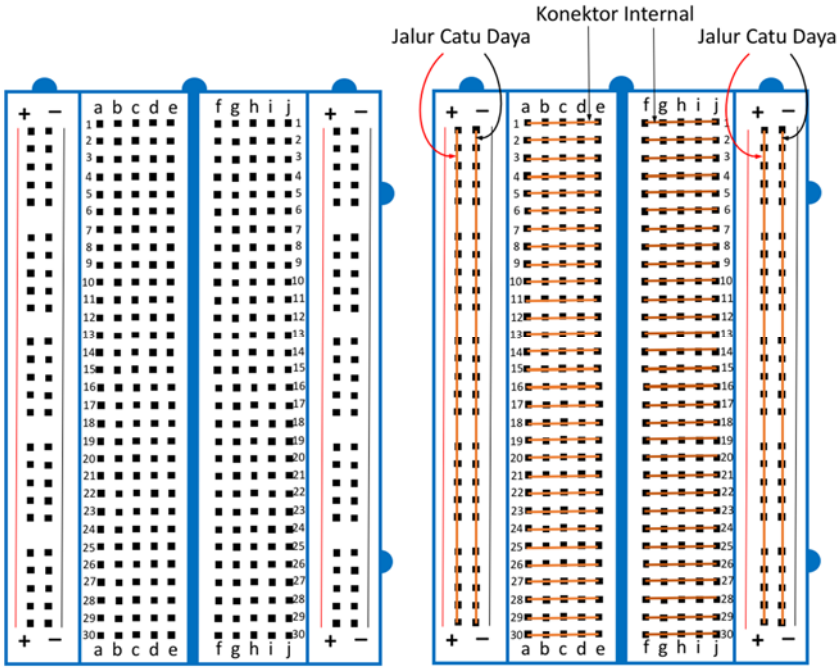
Papan Rangkaian Elektronik *Breadboard*

A. Pengertian

Breadboard adalah papan yang digunakan untuk menempatkan dan menyusun piranti/komponen-komponen elektronika menjadi rangkaian elektronika tanpa penyolderan. Hubungan antar piranti/komponen yang satu dengan piranti/komponen elektronika yang lain pada *breadboard* dilakukan melalui kawat/kabel.

B. Struktur *Breadboard*

Papan rangkaian ini dibuat dari plastik dan di dalamnya terdapat konektor-konektor yang dapat menjepit kaki-kaki piranti/komponen maupun ujung-ujung kabel. Struktur *breadboard* dengan koneksi internalnya ditunjukkan pada Gambar 1 berikut ini.



Gambar 1. Struktur *breadboard*

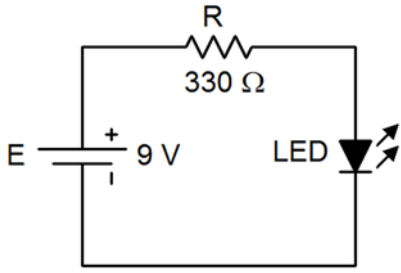
Lubang-lubang pada *breadboard* berfungsi menjepit kaki-kaki komponen/piranti dan kabel/kawat yang akan dirangkai. Gambar 1 sebelah kanan menunjukkan struktur konektor dalam *breadboard.* Konektor ditunjukkan dengan garis yang menghubungkan penjepit satu dengan lainnya. Pada sisi kiri dan kanan masing-masing terdapat dua jalur konektor yang berfungsi sebagai terminal positif (+) dan terminal negatif (-) catu daya. Sedangkan konektor-konektor pada jalur yang bertanda a-b-c-d-e dan f-g-h-i-j terputus dipisahkan oleh spasi yang digunakan untuk menempatkan piranti IC (*integrated circuit*) atau komponen dengan kemasan *dual-in-line package* (DIP).

Oleh karena penyusunan rangkaian menggunakan *breadboard* dilakukan tanpa penyolderan dan hubungan antar komponen/piranti yang satu dengan lainnya dilakukan melalui kabel, maka perlu disediakan kabel yang memenuhi syarat. Pada umumnya kabel yang digunakan untuk

mendukung penggunaan *breadboard* adalah kabel tunggal yang mudah dimasukkan ke dalam lubang-lubang pada papan itu.

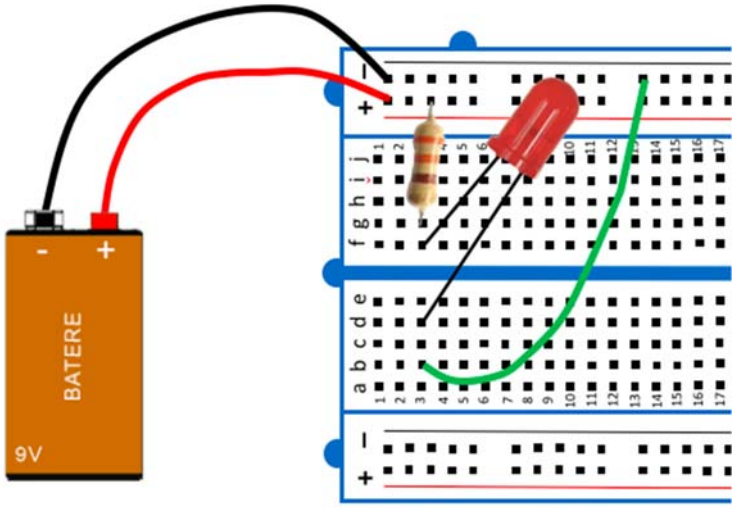
C. Penyusunan Rangkaian Dengan *Breadboard*

Bagaimanakah cara menggunakan *breadboard* untuk menyusun sebuah rangkaian LED (*light emitting diode)* sederhana dengan skema seperti pada Gambar 2 berikut ini?



Gambar 2. Rangkaian LED sederhana

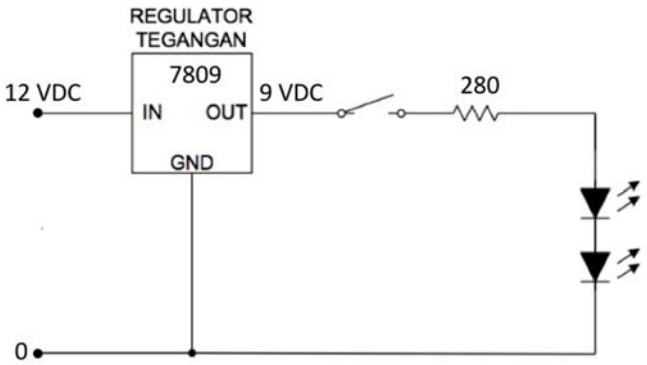
Untuk menyusun rangkaian tersebut menggunakan *breadboard,* perlu disiapkan terlebih dahulu alat dan bahan yakni sebuah LED, sebuah resistor 330 ohm, sebuah batere 9 volt, beberapa potong kabel tunggal dan pemotong kabel. Selanjutnya, piranti dan komponen dipasang seperti Gambar 3 berikut ini.



Gambar 3.

Susunan piranti dan komponen rangkaian LED pada *breadboard*

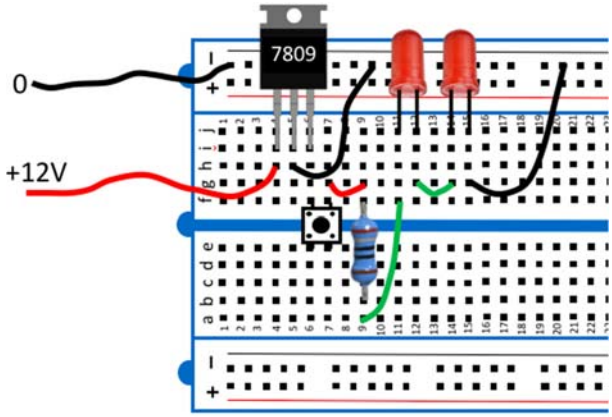
Contoh lain, coba susun rangkaian regulator tegangan dengan beban LED seperti pada gambar berikut ini menggunakan *breadboard*!



Gambar 4.

Rangkaian regulator tegangan dengan beban LED terhubung seri

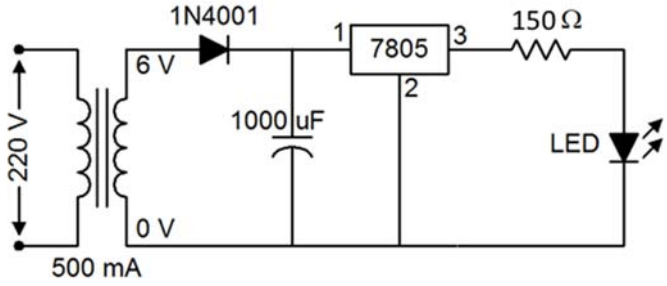
Untuk menyusun rangkaian pada Gambar 4 menggunakan *breadboard,* perlu disiapkan terlebih dahulu bahan dan alat yang diperlukan yakni *breadboard,* sumber tegangan DC 12V, IC regulator 7809, saklar tunggal, resistor 280 ohm, LED dua buah, beberapa potong kabel tunggal dan peralatan pemotong kabel. Selanjutnya dilakukan penyusunan komponen-komponen dan piranti-piranti yang tersedia sehingga menghasilkan rangkaian seperti pada Gambar 5.



Gambar 5.

Susunan rangkaian menggunakan *breadboard* untuk Gambar 4

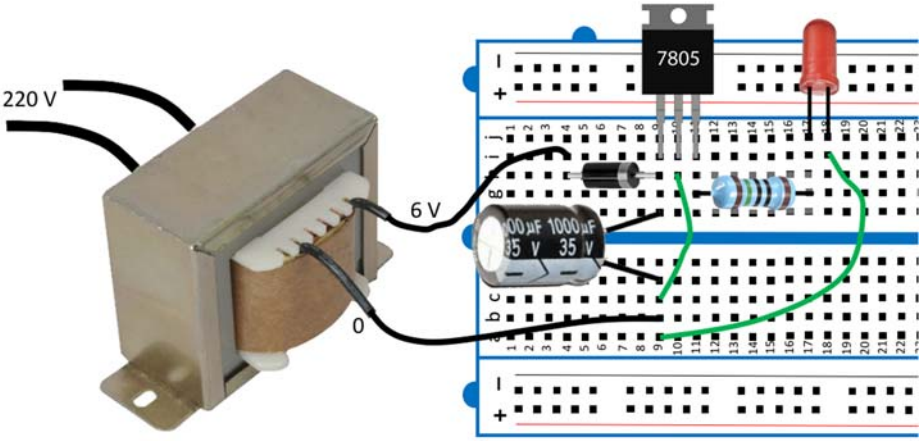
Coba, untuk rangkaian LED yang dicatu menggunakan penyearah setengah gelombang pada gambar berikut ini, seperti apakah rangkaiannya jika disusun di atas *breadboard*?



Gambar 6.

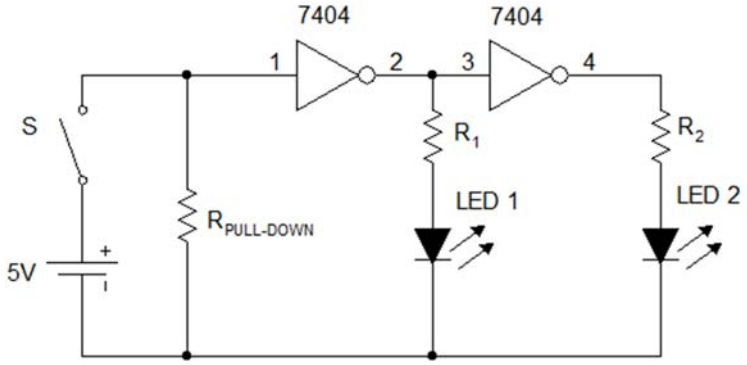
Susunan rangkaian penyearah setengah gelombang untuk mencatu LED

Jika disusun menggunakan *breadboard,* maka susunan rangkaiannya dapat dibuat seperti pada gambar berikut ini.



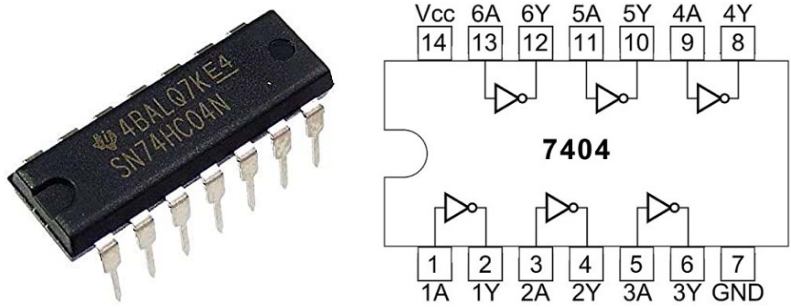
Gambar 7.
Susunan rangkaian menggunakan *breadboard* untuk Gambar 6

Untuk melatih keterampilan anda dalam menggunakan *breadboard*, coba susun rangkaian berikut ini di atas papan *breadboard*!



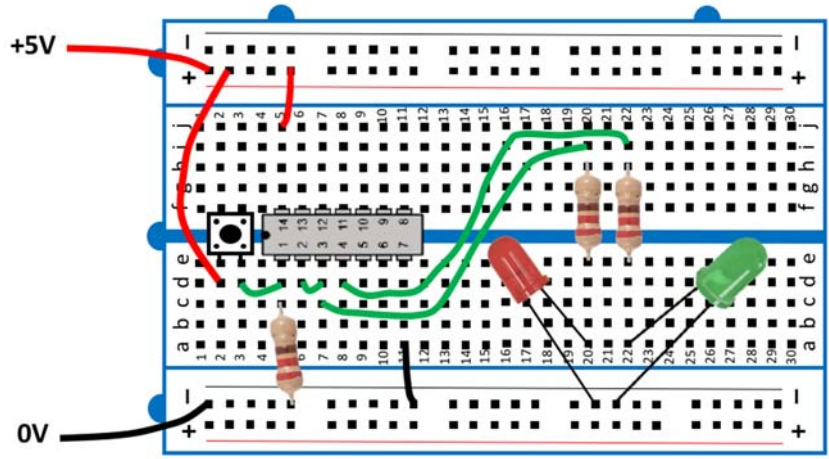
Gambar 8.
Susunan rangkaian komplemen ganda menggunakan IC 7404

Agar dapat menyusun rangkaian tersebut dengan benar, anda harus mengetahui terlebih dahulu susunan pin (kaki) dari IC 7404 (Gerbang NOT). Berdasarkan *datasheet*, susunan pin IC 7404 ditunjukkan Gambar 9.



Gambar 9.
Bentuk nyata IC 7404 (kiri), dan susunan *pinout* (kanan)

Dengan memperhatikan susunan pin IC 7404, maka rangkaian pada Gambar 8 dapat disusun menggunakan *breadboard* seperti ditunjukkan pada Gambar 10.



Gambar 10.
Susunan rangkaian pada Gambar 8 menggunakan *breadboard*

Selain tersedia dalam bentuk nyata, *breadboard* tersedia juga dalam bentuk virtual. Perangkat lunak yang menyediakan *breadboard* virtual ini dinamakan simulator *breadboard*.

Bagian II

Persyaratan Operasi Simulator *Breadboard*

Simulator *breadboard* dapat digunakan untuk mendukung kegiatan praktikum Teknik Digital melalui laboratorium virtual. Simulator *breadboard* merupakan perangkat lunak aplikasi yang menyediakan fasilitas analisis rangkaian elektronika digital berbasis papan rangkaian *breadboard.* Simulator ini menyedikan alat dan bahan virtual berupa *breadboard* dengan catu dayanya, berbagai IC logika jenis TTL (*transistor-transistor logic*), komponen saklar, peraga LED (*light emitting diode*) dan 7-segmen *display*, kabel serta *logic probe.* Simulator *breadboard* dikembangkan oleh tim dari Jurusan Ilmu Komputer Universitas *York* Inggris di bawah arahan *project supervisor* Dr. Chris Bailey. Simulator ini merupakan program aplikasi *desktop* dibuat dengan program Java dan dapat berjalan pada komputer lokal di atas sistem operasi Windows.

A. Perangkat Keras dan Perangkat Lunak

Simulator *breadboard* dapat berjalan dengan baik jika tersedia perangkat keras dan perangkat lunak dengan spesifikasi sebagai berikut:

1. Perangkat keras yang tersedia dapat berupa komputer desktop atau laptop dengan spesifikasi standar untuk pengoperasian aplikasi *desktop.* Perangkat keras ini harus bisa mendukung operasi program-program aplikasi *desktop* seperti program-program *Microsoft Office*.

2. Pada komputer lokal harus terpasang sistem operasi keluarga Windows seperti Windows XP, Windows Vista atau Wondows 7
3. Pada komputer lokal juga harus terpasang program *Java Runtime Environment* (JRE).

B. Instalasi *Java Runtime Environment* (JRE)

Simulator *breadboard* yang digunakan merupakan *applet* Java sehingga memerlukan JRE untuk menjalankannya. Cara instalasi program JRE dapat diikuti sebagai berikut:

1. *Download* terlebih dahulu *installer* dari *server* UAD menggunakan link:
   http://muchlas.ee.uad.ac.id/downloads/jre-7u21-windows-i586.exe

2. Lakukan instalasi JRE tersebut pada komputer anda dengan urut-urutan: jalankan *installer* JRE dengan klik *double* pada *file* **hasil download** dan klik *Install* setelah muncul jendela *Java Setup-Welcome* berikut ini!

Gambar 11. Jendela *Java Setup Welcome*

3. Sesaat akan ditampilkan jendela status instalasi dan anda harus menunggunya.

Gambar 12. Jendela status instalasi *Java Setup*

4. Setelah muncul jendela *Java Setup-Complete*, klik pada tombol *Close* untuk mengakhiri proses instalasi JRE.

Gambar 13. Jendela *Java Setup-Complete*

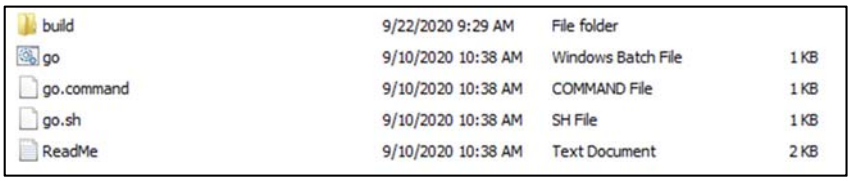
Sampai tahap ini instalasi persyaratan operasi simulator *breadboard* telah selesai dilakukan dan komputer anda sekarang telah siap menjalankan simulator *breadboard.*

C. Instalasi Simulator *Breadboard*

Untuk menginstalasi simulator *breadboard* di komputer lokal, lakukan langkah-langkah berikut ini:

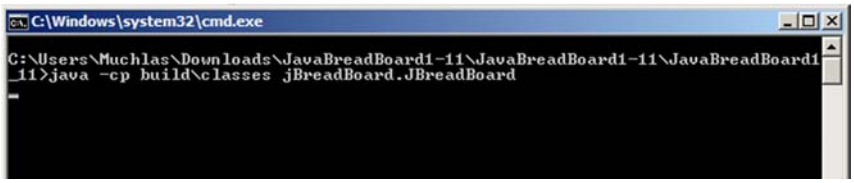
1. Lakukan *download* terlebih dahulu program simulator ini dari server UAD dengan link:
   http://muchlas.ee.uad.ac.id/downloads/JavaBreadBoard1-11.zip

2. Lakukan *extract* terhadap *file* JavaBreadBoard1-11.zip menggunakan program seperti WinZip, pilihlah lokasi *folder* hasil *extract* misalnya di C. Jika ekstraksi *file* tersebut berhasil maka akan menghasilkan folder C:\JavaBreadBoard1-11 dan di dalamnya terdapat *file* go dengan icon  . Secara lengkap isi foldernya seperti gambar berikut ini.



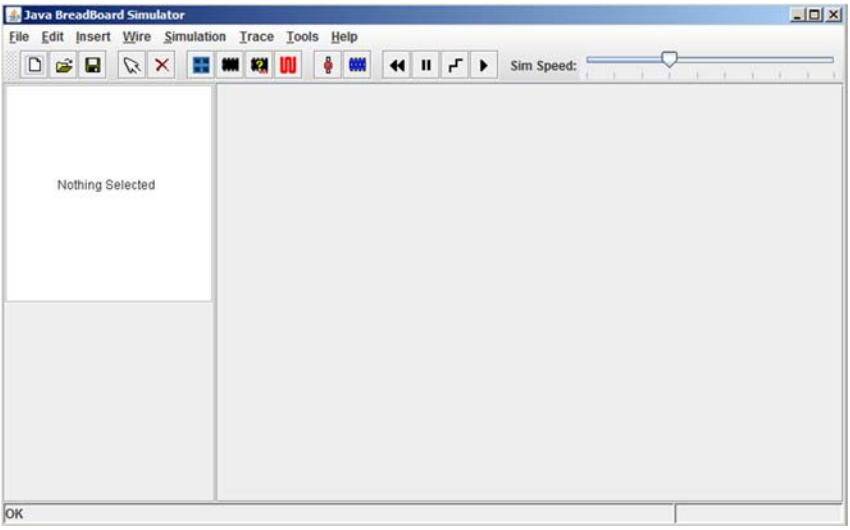
Gambar 14. Isi folder simulator *breadboard*

3. Untuk mencoba keberhasilan instalasi, jalankan simulator *breadboard* dengan klik *double* pada  . Program akan menampilkan 2 jendela, yakni pertama:



Gambar 15. Jendela eksekusi Java

Biarkan jendela ini tetap ada selama anda menjalankan program simulator *breadboard*. Jika tampilannya mengganggu, lakukan *minimizing* saja, dengan klik pada tanda "-", jangan ditutup, dan jendela kedua adalah:



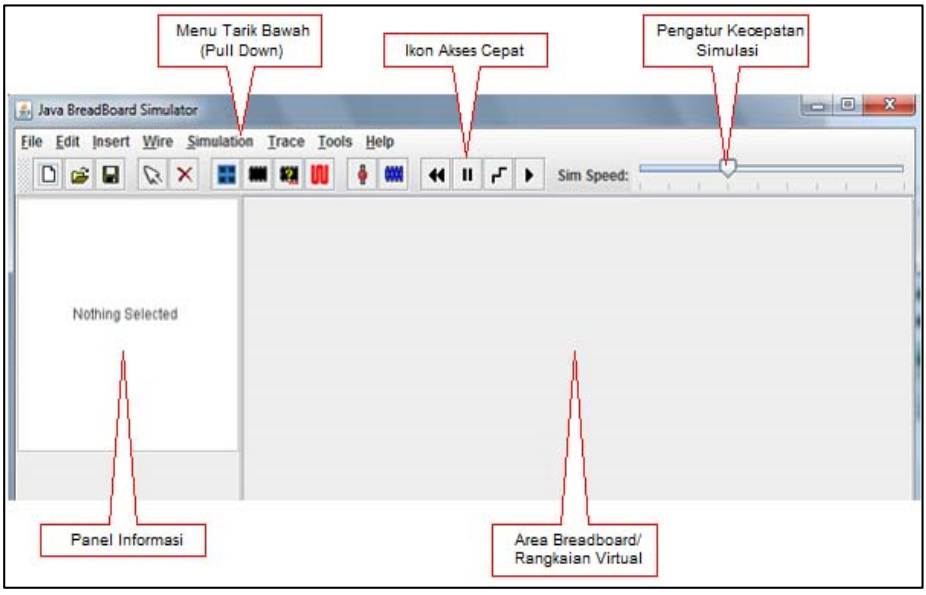
Gambar 16. Jendela antarmuka simulator *breadboard*

4. Untuk selanjutnya, simulator *breadboard* dijalankan dengan cara masuk terlebih dahulu ke folder C:\JavaBreadBoard1-11 diteruskan dengan klik pada go .

Bagian III

Kelengkapan Simulator *Breadboard*

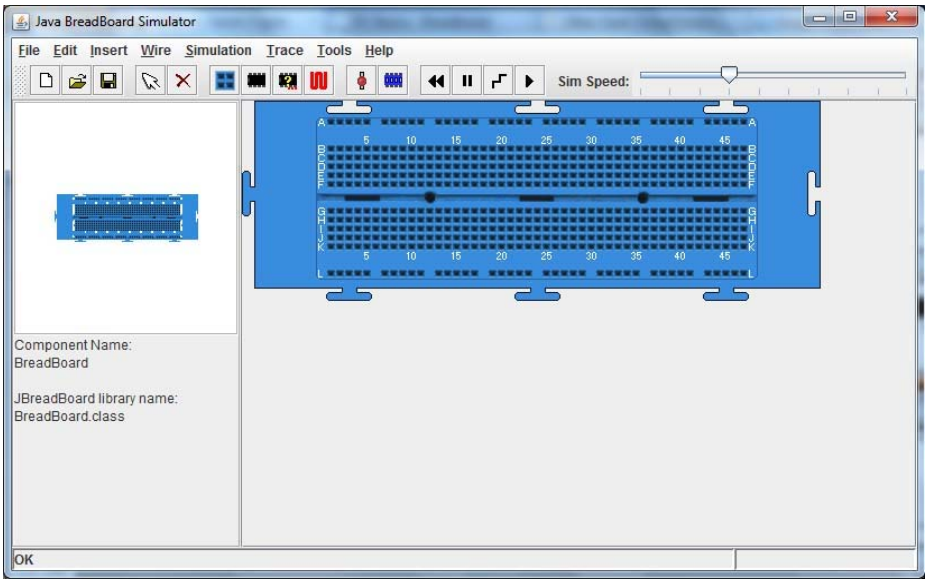
Jika simulator *breadboard* dijalankan dengan prosedur seperti pada Bagian II di atas, maka akan tampil antarmuka seperti berikut ini.



Gambar 17. Tampilan antarmuka simulator *breadboard*

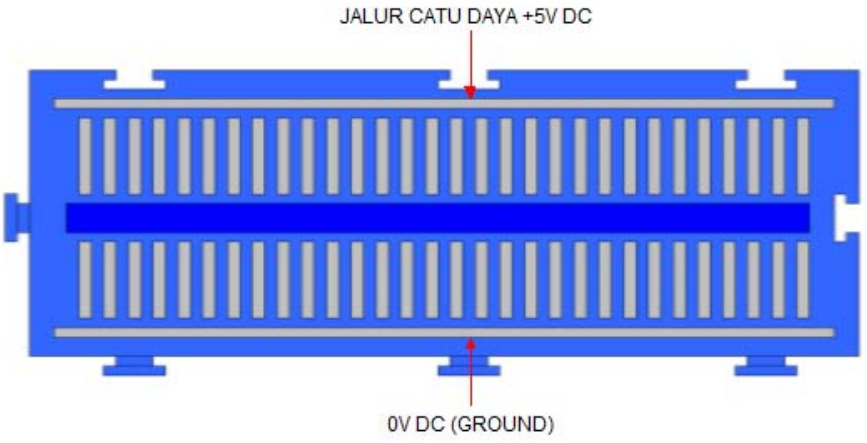
A. *Breadboard*

Untuk menampilkan *breadboard* klik pada ikon  pada deretan ikon akses cepat, atau klik **Insert>Breadboard**, maka akan ditayangkan:



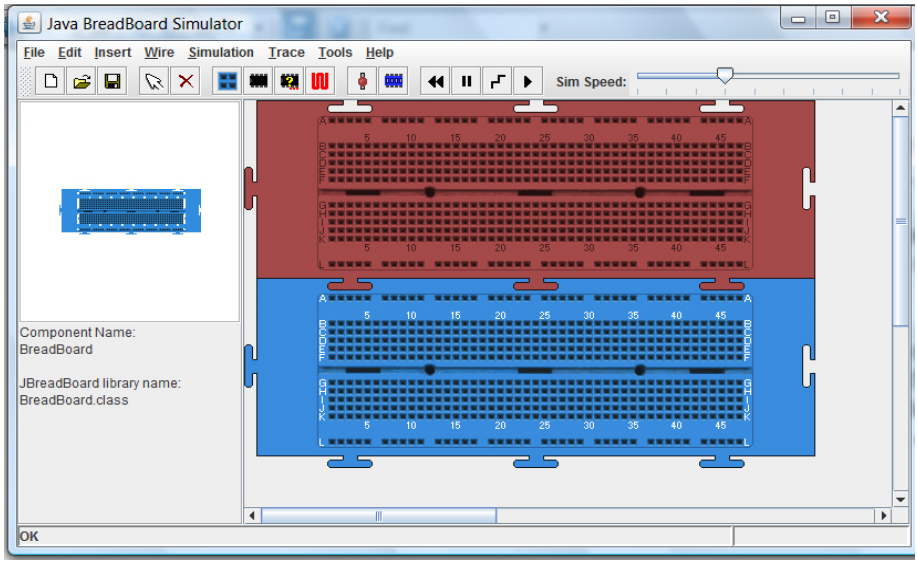
Gambar 18. *Breadboard* virtual

Struktur konektor-konektor internal pada model *breadboard* virtual simulator ini ditunjukkan pada gambar berikut.



Gambar 19. Struktur internal *breadboard* virtual

Dari struktur tersebut terlihat bahwa pada sisi atas terdapat jalur catu daya +5V dan pada sisi bawah terdapat jalur catu daya 0V (ground). Untuk menambah *breadboard* klik kembali **Insert>Breadboard** atau klik  sehingga muncul:

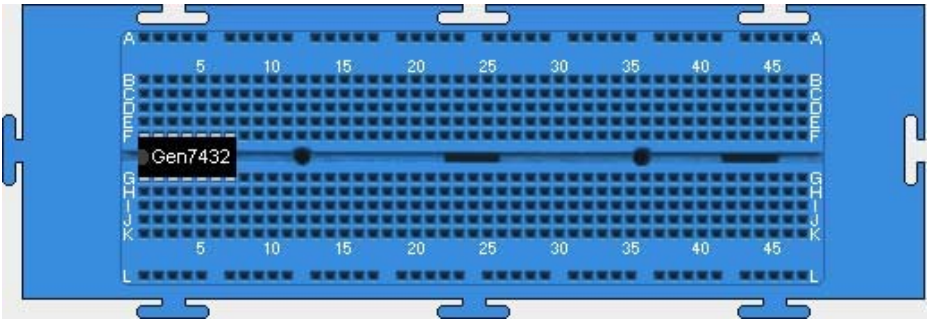


Gambar 20. *Breadboard* ganda

Untuk membersihkan layar dari *breadboard* klik ikon  atau **Edit>Delete.**

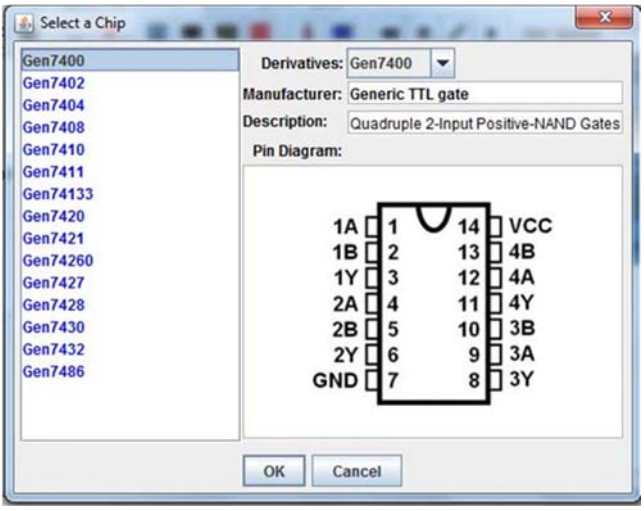
B. Piranti IC (*Chip*)

Simulator ini menyediakan berbagai piranti IC yang dapat digunakan sebagai bahan untuk eksperimen teknik digital. Untuk mengambil piranti IC, misalnya *chip* 7432 (gerbang OR), *load* terlebih dahulu *breadboard* dengan klik **Insert>Breadboard** atau klik  , selanjutnya diteruskan dengan melakukan klik **Insert>Chip>TTL>Logic>Gen7432>OK.** Jika prosedur itu dilakukan dengan benar, maka akan muncul gambar berikut.



Gambar 21. Penyiapan piranti IC pada *breadboard*

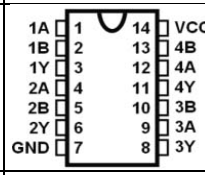
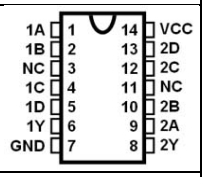
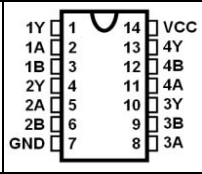
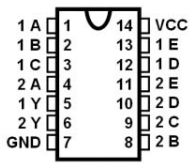
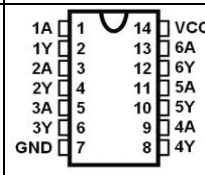
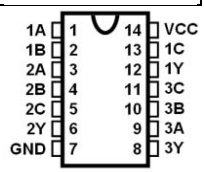
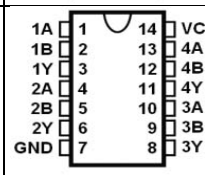
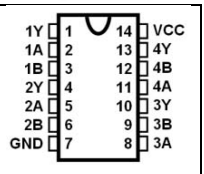
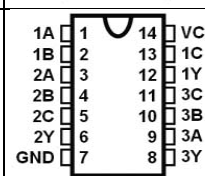
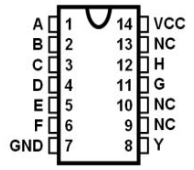
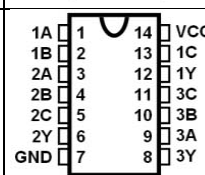
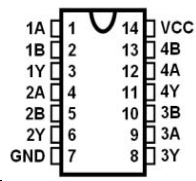
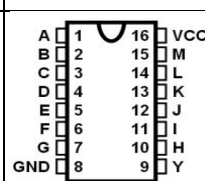
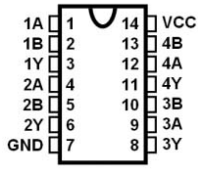
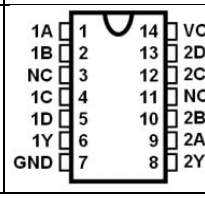
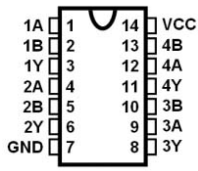
IC gerbang logika dasar yang disediakan oleh simulator ini diakses melalui antarmuka seperti gambar berikut ini.



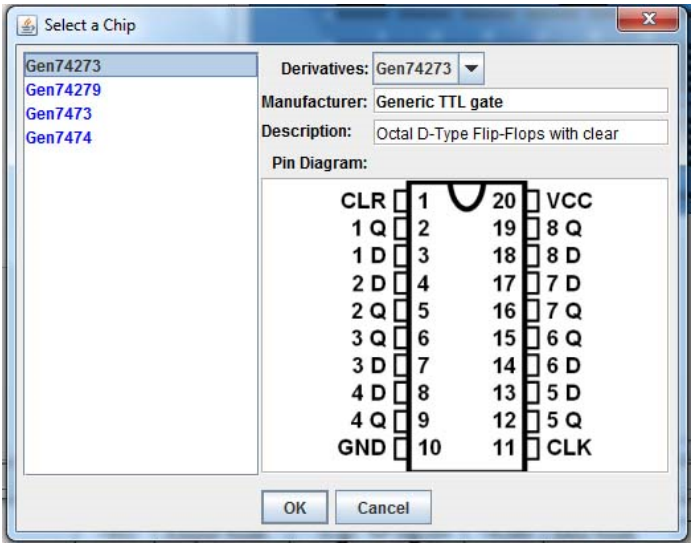
Gambar 22. Antarmuka untuk mengakses IC gerbang logika dasar

Secara lebih lengkap, IC gerbang logika dasar yang disediakan oleh simulator *breadboard* disajikan pada tabel berikut ini.

Tabel 1. Nomor seri IC gerbang logika dasar pada simulator *breadboard*

Nomor IC	Deskripsi	*Pinout*	Nomor IC	Deskripsi	*Pinout*
7400	Empat buah gerbang NAND dengan 2-input		7421	Dua buah gerbang AND dengan 4-input	
7402	Empat buah gerbang NOR dengan 2-input		74260	Dua buah gerbang NOR dengan 5-input	
7404	Enam buah gerbang NOT		7427	Tiga buah gerbang NOR dengan 3-input	
7408	Empat buah gerbang AND dengan 2-input		7428	Empat buah gerbang NOR dengan 2-input	
7410	Tiga buah gerbang NAND dengan 3-input		7430	Sebuah buah gerbang NAND dengan 8-input	
7411	Tiga buah gerbang AND dengan 3-input		7432	Empat buah gerbang OR dengan 2-input	
74133	Gerbang NAND dengan 13-input		7486	Empat buah gerbang XOR dengan 2-input	
7420	Dua buah gerbang NAND dengan 4-input		7486	Empat buah gerbang XOR dengan 2-input	

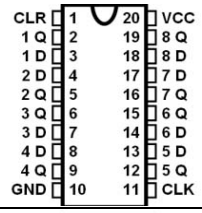
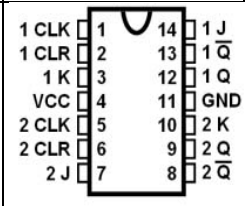
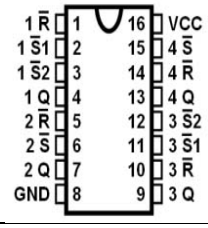
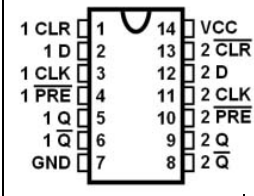
Simulator ini juga menyediakan IC *flip-flop* seperti 74273. Untuk mengambilnya, pastikan *breadboard* telah diambil, diteruskan dengan melakukan klik pada **Insert>Chip>TTL>flipFlop>Gen74273>OK.**



Gambar 23. Antarmuka untuk mengakses IC *flip-flop*

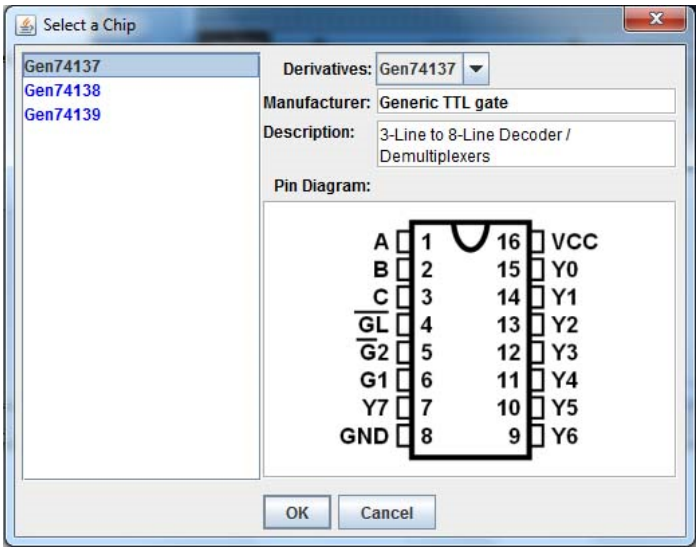
Secara lengkap IC *flip-flop* yang disediakan simulator *breadboard* disajikan tabel berikut ini.

Tabel 2. Nomor seri IC *flip-flop* pada simulator *breadboard*

Nomor IC	Deskripsi	*Pinout*	Nomor IC	Deskripsi	*Pinout*
74273	Delapan buah flip-flop D dengan *clear*		7473	Dua buah *flip-flop* JK dengan *clear*	
74279	Empat buah flip-flop SR-*latch*		7474	Dua buah *flip-flop* D dengan *preset* dan *clear*	

Notasi pada tabel di atas adalah D input *flip-flop* D, J dan K input *flip-flop* JK, S dan R input *flip-flop* SR, CLR input *clear,* PRE input *preset*, CLK input *clock,* dan Q merupakan output *flip-flop* semua jenis.

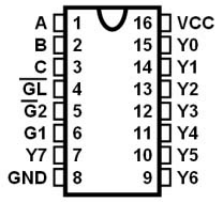
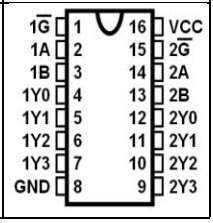
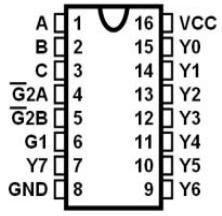
IC lain yang disediakan simulator ini adalah IC dekoder seperti 74137. Untuk mengambil IC ini, pastikan *breadboard* telah diambil, dilanjutkan dengan klik **Insert>Chip>TTL>decoder>Gen74137>OK.**



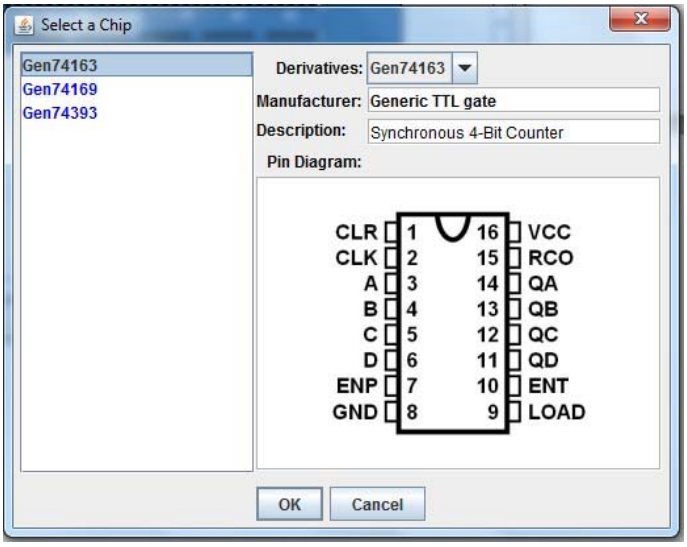
Gambar 24. Antarmuka untuk mengakses IC dekoder

Secara lebih lengkap, IC TTL dekoder yang disediakan simulator *breadboard* disajikan pada tabel berikut ini.

Tabel 3. Nomor seri IC dekoder pada simulator *breadboard*

Nomor IC	Deskripsi	*Pinout*	Nomor IC	Deskripsi	*Pinout*
74137	Dekoder/ Demultiplekser 3-ke-8		74139	Dekoder/ Demultiplekser 3-ke-8	
74138	Dekoder/ Demultiplekser 3-ke-8				

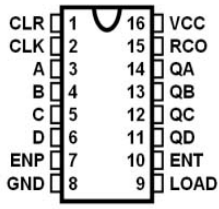
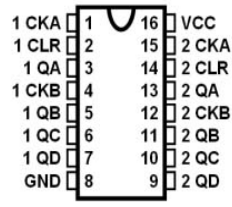
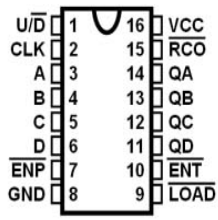
Selain IC gerbang logika dasar, *flip-flop* dan dekoder, simulator *breadboard* juga menyediakan IC pencacah (*counter*) dan register. Untuk mengambil IC pencacah misalnya 74163, pastikan *breadboard* telah diambil, klik **Insert>Chip>TTL>counter>Gen74163>OK.**



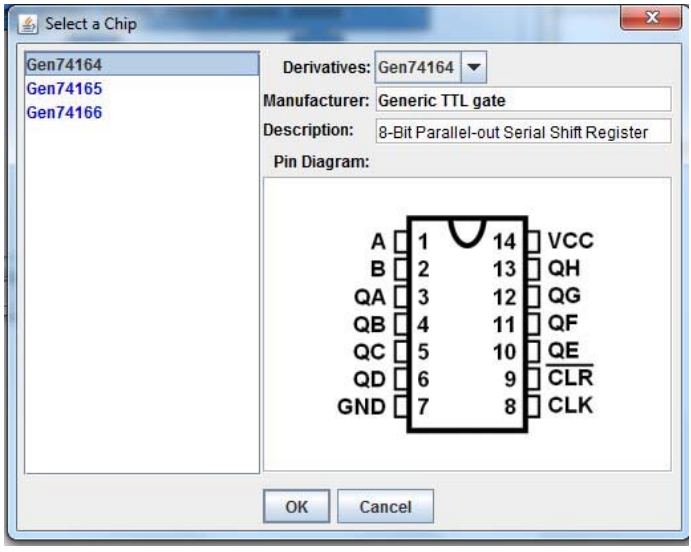
Gambar 25. Antarmuka untuk mengakses IC *counter*

IC *counter* yang disediakan oleh simulator *breadboard* secara lengkap disajikan melalui tabel berikut ini.

Tabel 4. Nomor seri IC *counter* pada simulator *breadboard*

Nomor IC	Deskripsi	*Pinout*	Nomor IC	Deskripsi	*Pinout*
74163	Pencacah 4-bit sinkron		74393	Dua buah pencacah biner 4-bit	
74169	Pencacah 4-bit sinkron				

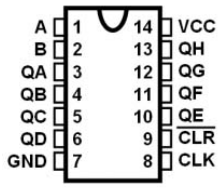
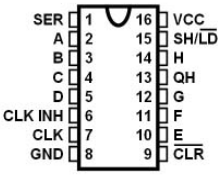
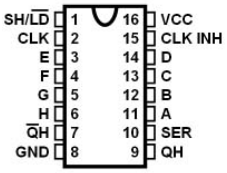
Sedangkan untuk mengambil IC register misalnya 74164, pastikan *breadboard* telah diambil, klik **Insert>Chip>TTL>counter>Gen74164> OK.**



Gambar 26. Antarmuka untuk mengakses IC register

Daftar IC register yang disediakan oleh simulator *breadboard* secara lengkap disajikan pada tabel berikut ini.

Tabel 5. Nomor seri IC register pada simulator *breadboard*

Nomor IC	Deskripsi	*Pinout*	Nomor IC	Deskripsi	*Pinout*
74164	8-Bit Parallel-out Serial Shift Register		74166	Parallel-load 8-Bit Shift Register	
74165	Parallel-load 8-Bit Shift Register				

C. Kabel Penghubung

Simulator *breadboard* menyediakan kabel penghubung dalam berbagai warna. Untuk mengambil kabel dan memasangnya pada *breadboard,* mode pengkabelan (*wiring mode*) harus diaktifkan dengan klik pada ikon  , atau **klik Wire>Add** Wires, dan untuk menonaktifkannya tekan tombol *keyboard* ***backspace.*** Secara umum, cara memasang kabel pada *breadboard* virtual ini adalah sebagai berikut:

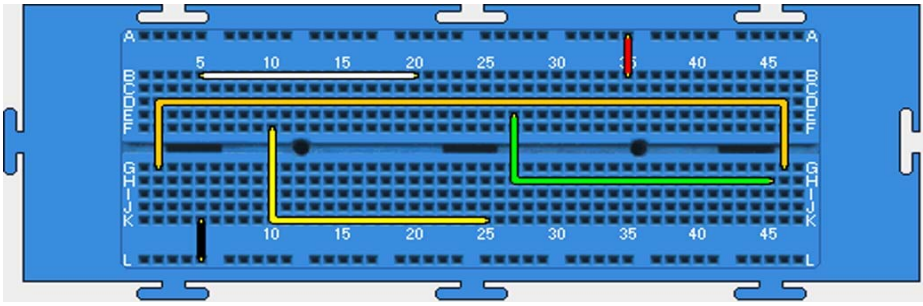
1. Bersihkan layar dengan **klik**  , dan ambil *breadboard* dengan klik **Insert>Breadboard** atau klik 
2. Pastikan dahulu *wiring mode* dalam keadaan aktif dengan klik  .
3. Pilih warna kabel misalnya putih dengan klik **Wire>White**
4. Klik pada titik awal kabel misalnya **B5**
5. Untuk membatalkan titik awal, tekan **Esc** pada *keyboard*
6. Arahkan kabel ke titik akhir misalnya **B20**
7. ***Double*** **klik** pada titik akhir **B20** untuk mengakhiri pemasangan kabel
8. Untuk menonaktifkan *wiring mode* tekan **backspace** pada *keyboard.*

Jika kabel mengalami jalur yang membelok, **klik satu kali** untuk membelokkannya. Kabel yang telah terpasang juga dapat dihapus dengan **klik pada kabel** diteruskan menekan tombol **delete** pada keyboard, atau **klik pada kabel>edit>delete** atau **klik pada kabel>**. Pada tabel berikut ini ditunjukkan cara menghubungkan titik tertentu ke titik lain pada *breadboard* dengan kabel berbagai warna.

Tabel 6. Cara memasang kabel pada simulator *breadboard*

Hubungan Antar Titik	Warna Kabel	Cara
B5-B20	Putih	Klik Wire>White>B5>double klik B20
F10-K25	Kuning	Klik Wire>Yellow>F10>K10>double klik K25
B35-Vcc(+5V)	Merah	Klik Wire>Red>B35>double klik A35
K5-Ground (0V)	Hitam	Klik Wire>Black>K5>double klik L5
E27-H45	Hijau	Klik Wire>Green>E27>H27>double klik H45
G2-G46	Orange	Klik Wire>Orange>G2>D2>D46>double klik G46

Jika prosedur pemasangan kabel tersebut dilakukan dengan benar maka akan diperoleh susunan kabel pada *breadboard* seperti gambar berikut ini.



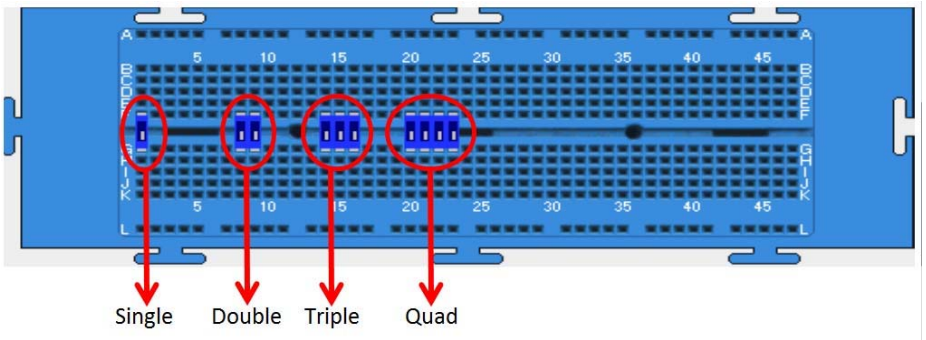
Gambar 27. Contoh pemasangan kabel pada simulator *breadboard*

D. Komponen Input

Simulator *breadboard* menyediakan berbagai jenis komponen input seperti saklar, *keypad*, dan osilator.

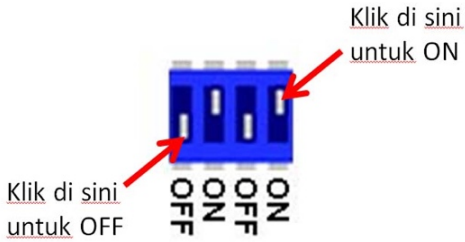
1. Input Saklar

Komponen input saklar yang disediakan oleh simulator ini menggunakan desain DIP *switch* terdiri atas empat jenis yakni saklar tunggal (*single*), saklar ganda (*double*), saklar *triple* dan saklar *quad*. Untuk mengambil saklar, ambil terlebih dahulu *breadboard* dengan dengan klik **Insert>Breadboard** atau klik  klik **Insert>Dip Switches>single** untuk saklar tunggal, atau klik **Insert>Dip Switches>double** untuk saklar ganda, atau klik **Insert>Dip Switches>treble** untuk saklar triple, atau klik **Insert>Dip Switches>quad** untuk saklar quad.



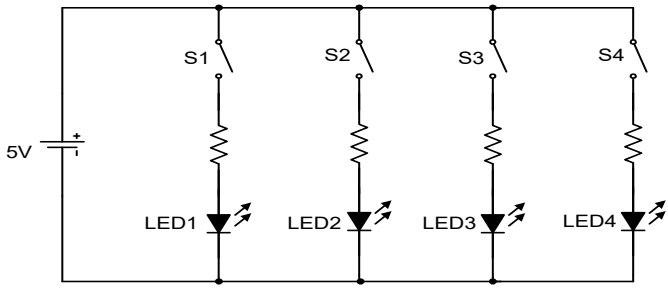
Gambar 28. Saklar pada simulator *breadboard*

Cara mengubah posisi saklar adalah dengan klik pada saklar dilanjutkan dengan klik pada posisi yang diinginkan. Jika dilakukan klik pada saklar, maka saklar berubah warnanya menjadi kuning dan pada keadaan ini sklar siap diubah dengan cara seperti ditunjukkan pada gambar berikut ini.



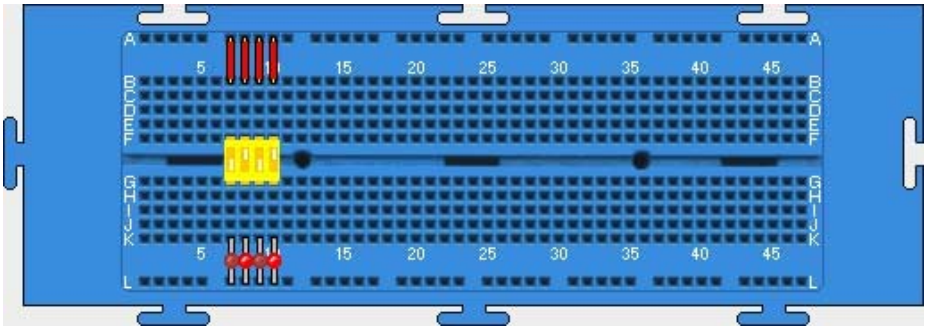
Gambar 29. Cara mengubah posisi saklar pada simulator *breadboard*

Untuk mempelajari cara mengubah posisi saklar, coba susun rangkaian berikut ini menggunakan *breadboard*!



Gambar 30. Rangkaian untuk mencoba saklar

Jika rangkaian di atas disusun dengan benar, maka akan menghasilkan susunan seperti pada gambar berikut ini.

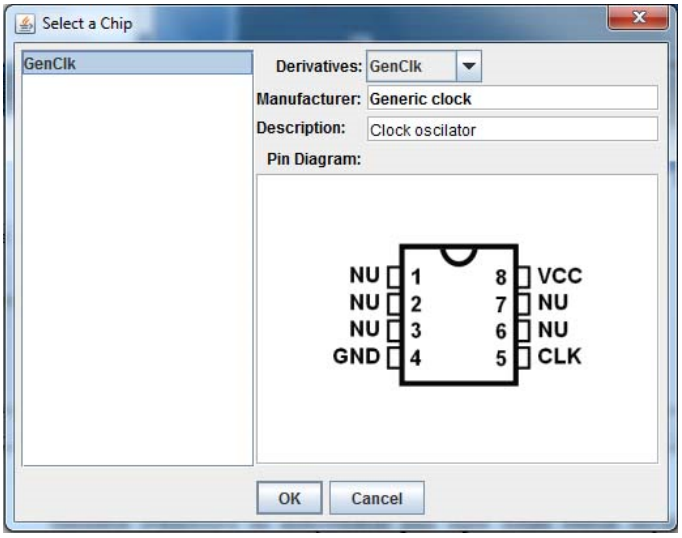


Gambar 31. Rangkaian untuk mencoba saklar pada simulator *breadboard*

Coba ubah-ubahlah posisi keempat saklar tersebut, apakah sudah memberikan pengaruh terhadap nyala LED?

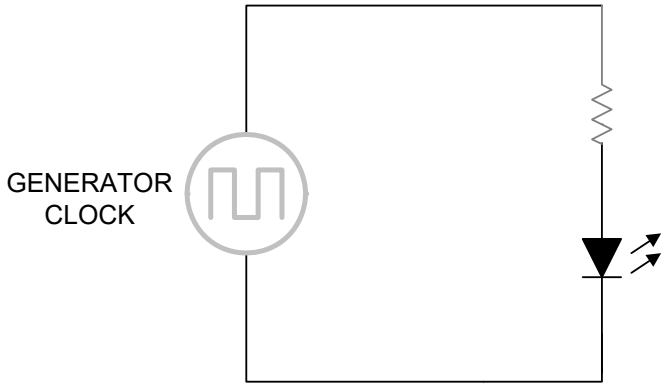
2. Input *Clock*

Simulator *breadboard* ini menyediakan pula input dalam bentuk sinyal *clock* yang frekuensinya dapat diatur. Sinyal *clock* dibangkitkan oleh sebuah osilator yang disediakan oleh sebuah IC *clock.* Untuk mengambil IC *clock*, pastikan *breadboard* telah ditampilkan pada jendela simulator lakukan klik **Insert>Chip>Oscillator>GenClk>OK.** Antar muka untuk mengakses *clock* ditunjukkan pada gambar berikut ini.



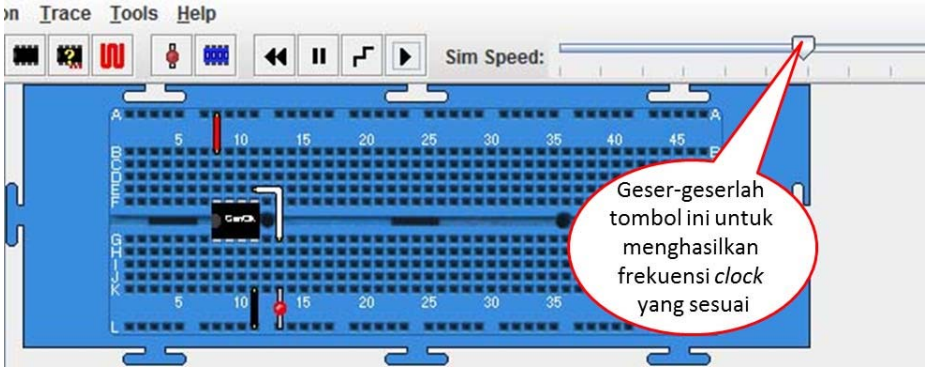
Gambar 32. Antarmuka untuk akses IC *clock*

Untuk melihat watak IC *clock*, coba susun rangkaian seperti berikut ini!



Gambar 33. Rangkaian LED dengan input sinyal *clock*

Jika rangkaian tersebut disusun dengan benar menggunakan *breadboard* virtual, maka susunannya akan tampak seperti gambar berikut ini.

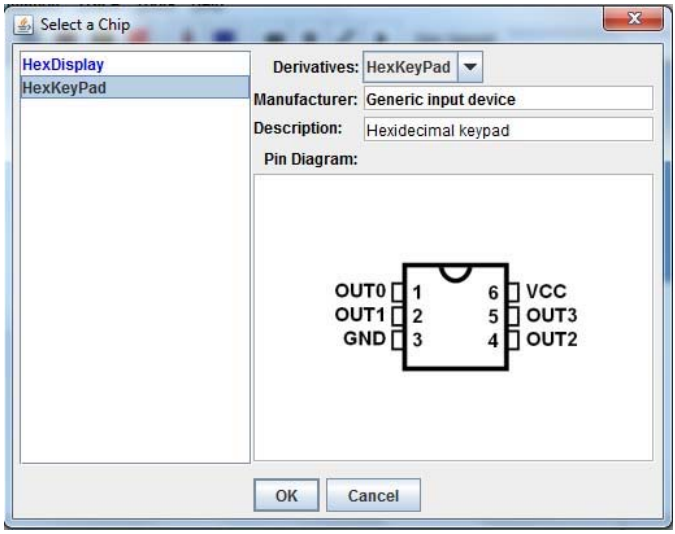


Gambar 34. Rangkaian pembangkit *clock*

Jika simulasi dijalankan dan pengatur kecepatan digeser-geser maka akan diperoleh nyala LED yang berkedip menandakan adanya sinyal *clock*.

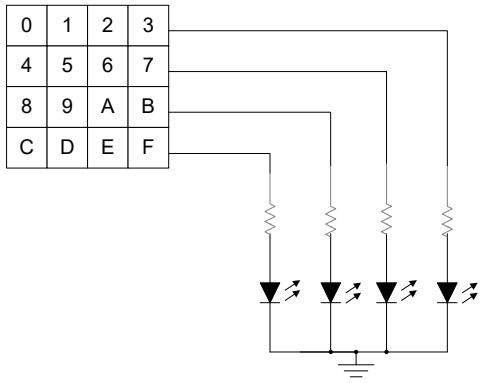
3. Input *Keypad* Heksadesimal

Selain saklar dan generator *clock*, simulator *breadboard* juga menyediakan komponen input dalam bentuk *keypad* heksadesimal. Komponen ini pada dasarnya merupakan sebuah enkoder yang mengubah kondisi saklar input heksadesimal menjadi kode biner. Komponen ini diakses dengan melakukan klik **Insert>Chip> Components>HexKeyPad>OK.** Antarmuka untuk akses *keypad* heksadesimal ditunjukkan pada gambar berikut ini.



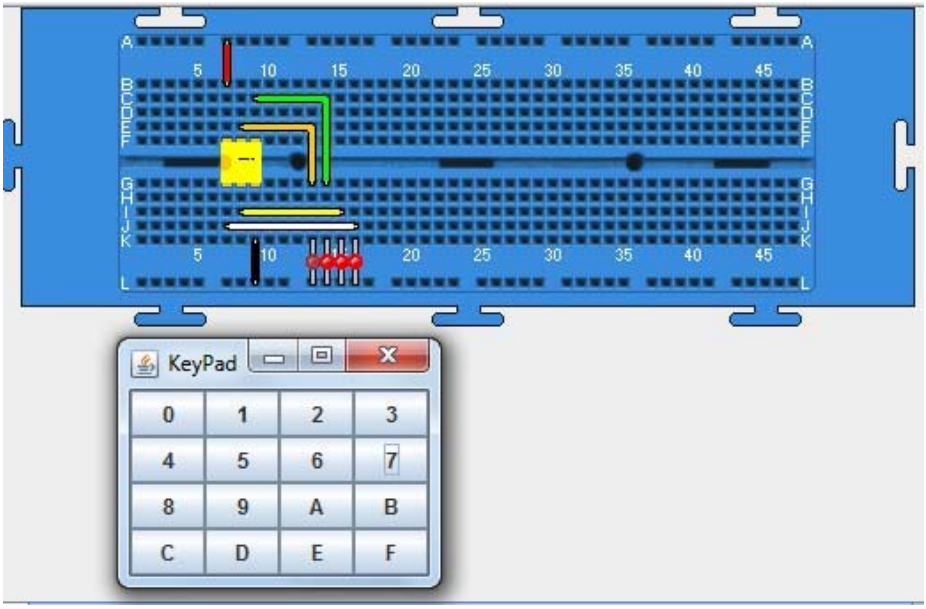
Gambar 35. Antarmuka untuk akses *keypad* heksadesimal

Untuk mencoba fungsi *keypad* heksadesimal coba susun rangkaian berikut ini di atas papan *breadboard* virtual.



Gambar 36. Rangkaian untuk mencoba *keypad* heksadesimal

Rangkaian tersebut dapat disusun menggunakan *breadboard* seperti pada gambar berikut ini.



Gambar 37. Rangkaian untuk mencoba *keypad* heksadesimal pada simulator *breadboard*

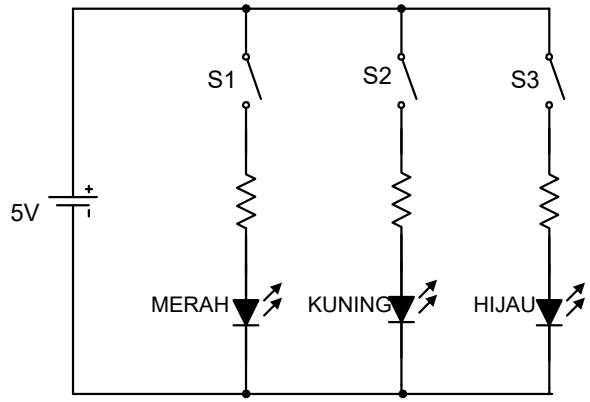
Untuk memunculkan *keypad* tekan klik *double* pada IC *clock.* Cobalah tekan/klik beberapa angka pada *keypad* dan amati nyala LED setiap penekanan tombol. Apakah nyala LED sudah menunjukkan kode biner yang sesuai dengan angka yang ditekan?

E. Komponen Output

Komponen output yang disediakan simulator *breadboard* terdiri atas indikator LED dan peraga heksadesmal.

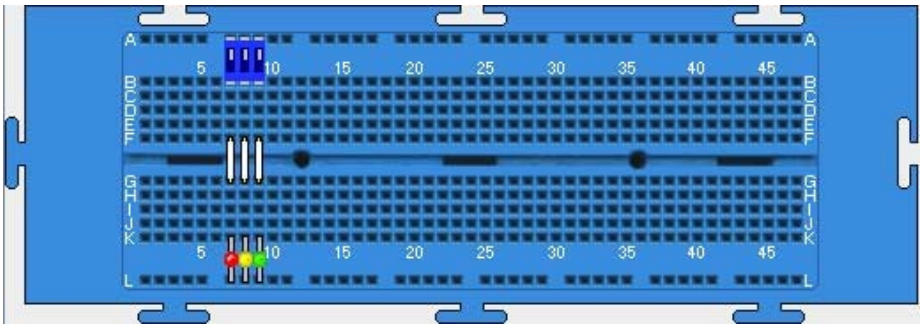
1. Indikator LED

Indikator LED merupakan komponen yang digunakan untuk menunjukkan keadaan logika suatu output gerbang atau rangkaian logika dan tersedia dalam tiga warna yakni merah, kuning dan hijau. LED diambil dengan cara klik **Insert>LED>Red** untuk warna merah, **Insert>LED>Yellow** untuk warna kuning, dan **Insert>LED>Green** untuk warna hijau. Coba susun rangkaian untuk menyalakan tiga buah LED berikut ini yang dilengkapi dengan tiga buah saklar.



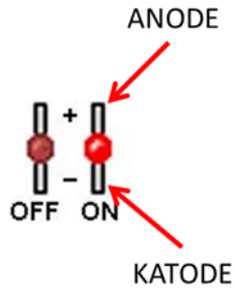
Gambar 38. Rangkaian untuk menyalakan LED

Rangkaian pada gambar di atas dapat disusun menggunakan *breadboard* seperti pada gambar berikut ini.



Gambar 39.
Rangkaian untuk menyalakan LED pada simulator *breadboard*

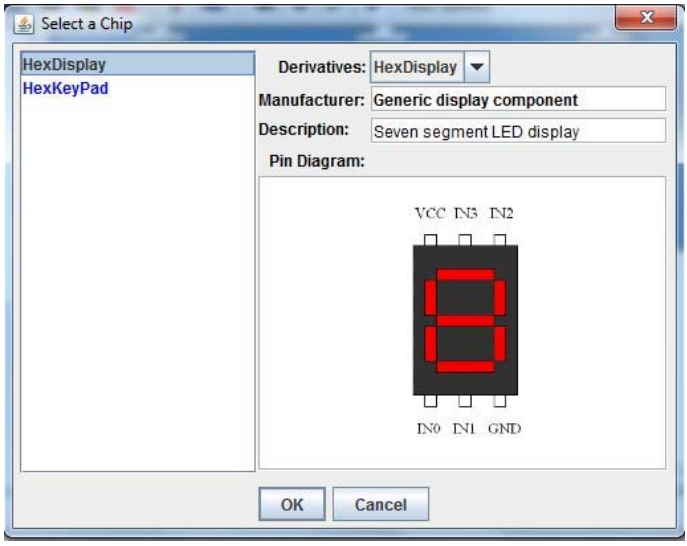
Untuk melihat kerja indikator LED, setelah simulasi dijalankan, coba atur ketiga saklar agar pada posisi ON. Jika saklar pada posisi ON maka ketiga LED akan menyala dengan warna merah, kuning dan hijau. Agar LED dapat menyala maka harus diberi prasikap maju dengan menghubungkan anode ke sumber tegangan (+) dan katode ke sumber tegangan (-). Susunan kaki anode dan katode LED pada simulator ini diperlihatkan pada gambar berikut ini.



Gambar 40. Susunan kaki anode dan katode LED

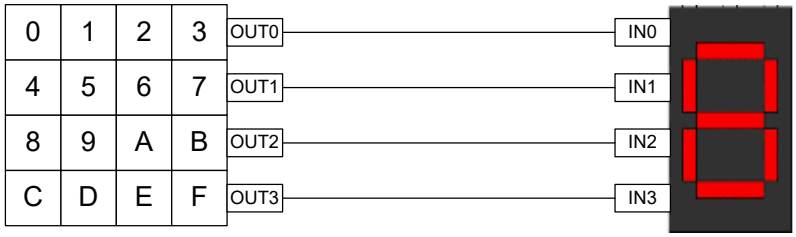
2. Peraga Heksadesimal

Peraga heksadesimal diambil dengan cara klik **Insert>Chip> Components>HexDisplay>OK.** Antarmuka untuk mengakses peraga heksadesimal diperlihatkan pada gambar berikut ini.



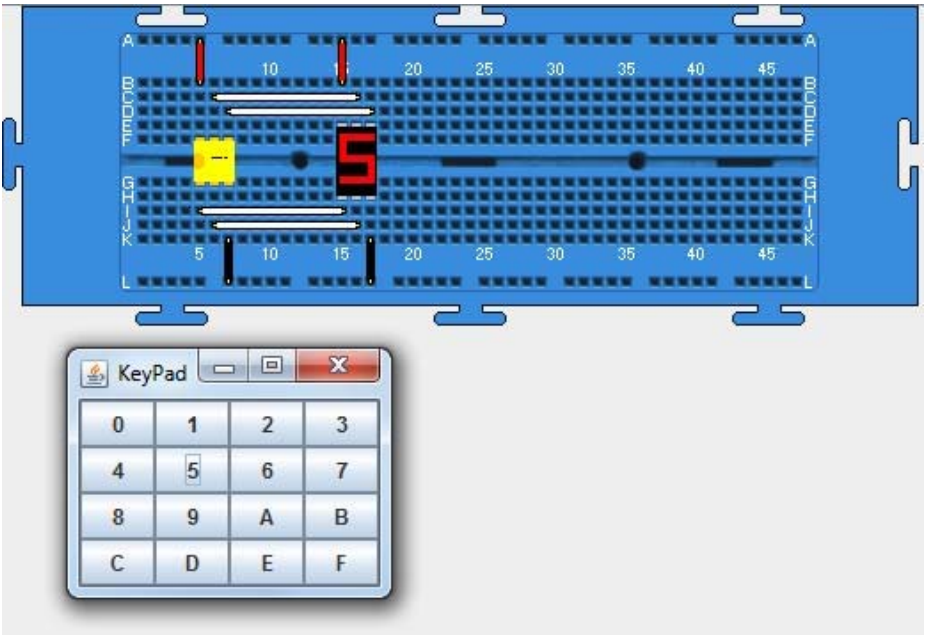
Gambar 41. Antarmuka untuk mengakses peraga heksadesimal

Untuk melihat cara kerja peraga heksadesimal, coba hubungkan peraga heksadesimal dengan *keypad* heksadesimal seperti pada gambar berikut ini.



Gambar 42. Rangkaian untuk mencoba peraga heksadesimal

Jika digunakan *breadboard* virtual, rangkaian di atas dapat disusun seperti ditunjukkan pada gambar berikut ini.



Gambar 43.
Susunan rangkaian untuk mencoba peraga heksadesimal
pada simulator *breadboard*

Untuk memperlihatkan cara kerja peraga heksadesimal, coba klik *double* pada *keypad* agar *keypad* muncul di layar. Coba tekan/klik beberapa angka pada *keypad*, apakah peraga heksadesimal telah menunjukkan angka yang sesuai?

Bagian IV

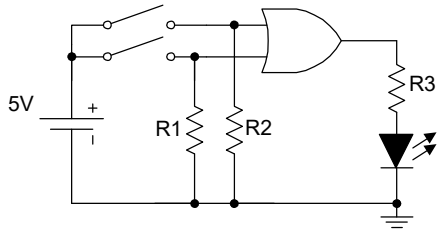
Penyusunan dan Simulasi Rangkaian Logika

Simulator ini, selain menyediakan *breadboard* virtual yang dapat digunakan untuk menyusun rangkaian logika dengan berbagai kelengkapannya, juga menyediakan fasilitas untuk simulasi terhadap rangkaian yang telah disusun.

A. Membuat Rangkaian Baru

Agar cara penyusunan rangkaian dapat dipelajari dengan mudah, perlu disediakan terlebih dahulu suatu rangkaian logika sebagai contoh.

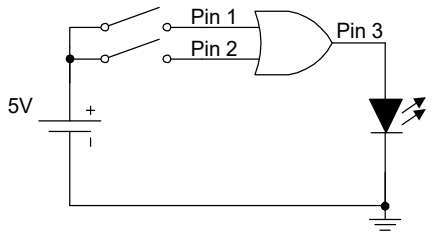
Contoh 1: Perhatikan rangkaian untuk menyelidiki watak gerbang OR berikut ini!



Gambar 44. Rangkaian asli untuk menentukan watak gerbang OR

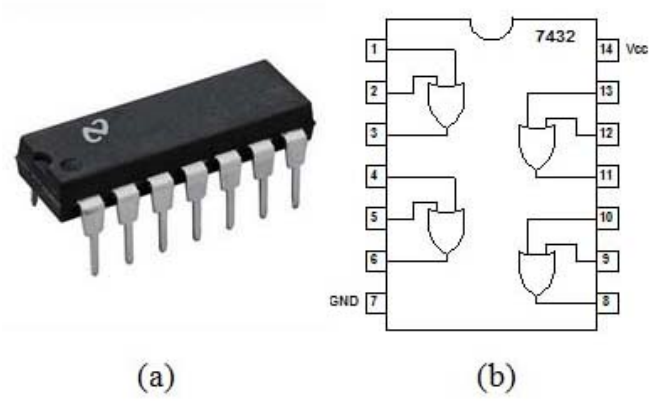
Sebelum menyusun rangkaian, perlu diketahui terlebih dahulu aturan-aturan pada simulator *breadboard* dan spesifikasi komponen-komponen yang digunakan. Simulator *breadboard* memiliki aturan bahwa setiap input

gerbang yang mengambang (saklar dalam keadaan OFF atau *floating input*), memiliki level logika rendah (0). Hal ini berbeda dengan keadaan sesungguhnya bahwa *floating input* dari IC TTL bernilai tinggi. Dengan demikian resistor *pull down*, yakni resistor yang menyebabkan suatu titik bernilai logika rendah yakni R1 dan R2, tidak diperlukan lagi. Selain itu, untuk menyalakan LED, simulator ini tidak memerlukan resistor pembatas arus R3, sehingga rangkaiannya menjadi seperti berikut ini.



Gambar 45. Rangkaian menurut aturan simulator *breadboard*

Rangkaian di atas mengandung sebuah gerbang OR. Telah diketahui bahwa gerbang OR disediakan oleh IC TTL dengan nomor seri 7432 dengan susunan pin seperti pada gambar berikut ini.



Gambar 46. IC 7432: (a) bentuk nyata, (b) susunan *pinout*

Berdasarkan gambar di atas terlihat bahwa di dalam IC 7432 terdapat empat buah gerbang OR. Untuk menyusun rangkaian pada Gambar 45, diperlukan hanya sebuah gerbang OR sehingga dapat ditentukan misalnya gerbang dengan input pin 1 dan 2 serta output pin 3. Dengan demikian dapat ditulis daftar komponen yang diperlukan untuk menyusun rangkaian, seperti disajikan pada tabel berikut ini.

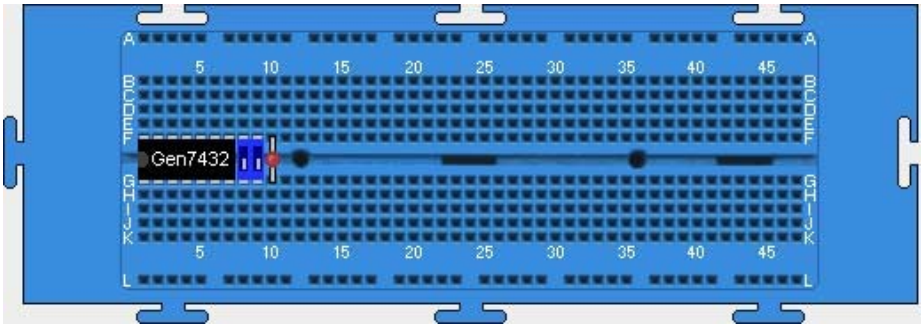
Tabel 7. Daftar komponen untuk rangkaian pada Gambar 45

Nama Komponen	Spesifikasi	Jumlah
Gerbang OR	IC TTL 7432	1 buah
Saklar	DIP *Switch* ganda (*double*)	1 buah
LED	Warna merah	1 buah

Langkah-langkah untuk menyusun rangkaian tersebut dengan *breadboard* virtual dalam simulator ini dapat diikuti sebagai berikut:

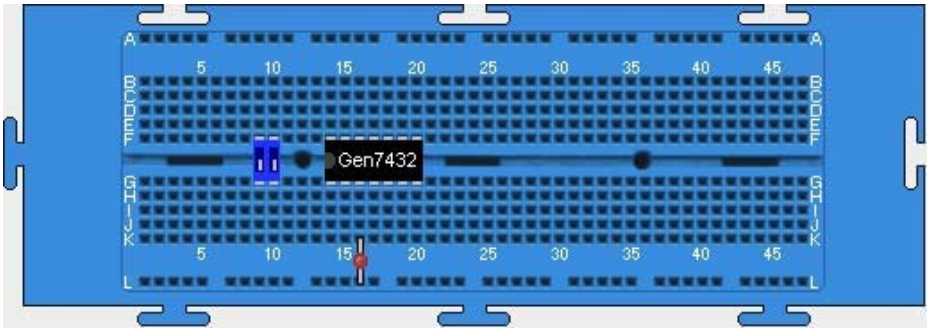
1. Jalankan simulator, diteruskan *load breadboard* dengan klik **Insert>Breadboard** atau klik .

2. Siapkan semua komponen yang diperlukan untuk menyusun rangkaian dengan klik **Insert>Chip>TTL>Logic>Gen7432>OK** untuk mengambil IC 7432, klik **Insert>DIP Switches>Double** untuk mengambil saklar DIP ganda, dan klik **Insert>LED>Red** untuk mengambil LED warna merah sehingga menghasilkan tampilan seperti gambar berikut ini.



Gambar 47. Penyiapan komponen pada *breadboard* virtual

3. Atur tata letak komponen misalnya seperti pada gambar berikut ini.

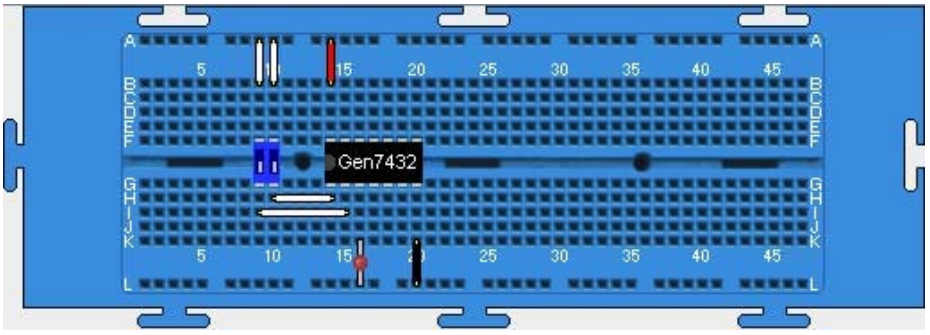


Gambar 48. Tata letak komponen pada *breadboard* virtual

4. Pasanglah catu daya pada IC dengan menghubungkan pin-14 IC ke kutub +Vcc menggunakan kabel merah dengan klik **Wires>Red>**  **>titik B14>arahkan kabel ke A14 dan klik *double*.**

5. Hubungkan pin-7 IC ke *Ground* (0 V) catu daya menggunakan kabel hitam dengan klik **Wires>Black>**  **>titik K20>arahkan kabel ke L20 dan klik *double*.**

6. Dengan cara yang sama dengan langkah di atas, pasang kabel putih untuk menghubungkan titik B9-A9, B10-A10, H10-H14, dan I9-I15 sehingga menghasilkan susunan rangkaian seperti gambar berikut ini.



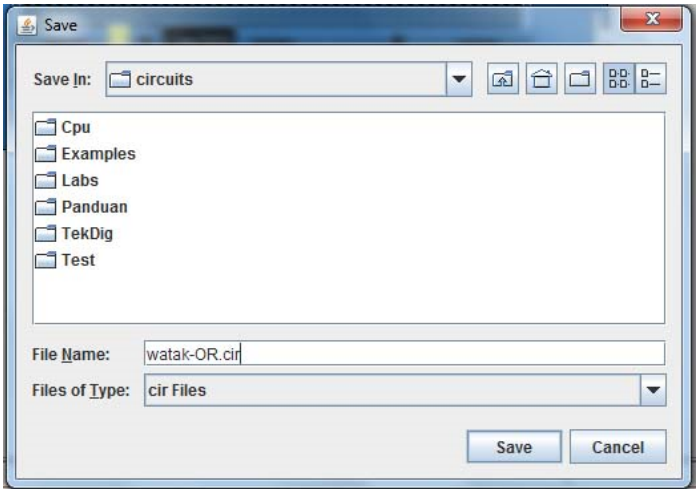
Gambar 49. Susunan rangkaian untuk Gambar 45

7. Untuk mencoba keberhasilan dalam penyusunan rangkaian, jalankan simulasi dengan klik **Simulation>Run** atau klik  , diteruskan dengan mengubah-ubah posisi saklar. Rangkaian telah berhasil disusun dengan baik jika menghasilkan tabel kebenaran sebagai berikut.

Tabel 8.
Tabel untuk memastikan keberhasilan penyusunan rangkaian

KONDISI SAKLAR		KONDISI LED
SAKLAR 1	SAKLAR 2	
OFF	OFF	PADAM
OFF	ON	MENYALA
ON	OFF	MENYALA
ON	ON	MENYALA

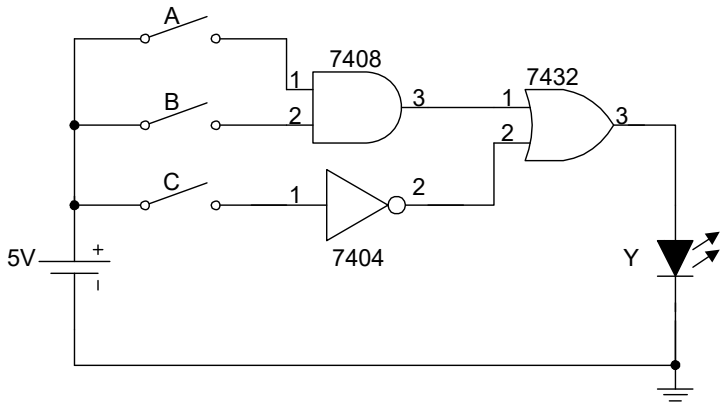
8. Simpanlah rangkaian tersebut menjadi *file* ke dalam media penyimpan seperti *harddisk* dengan klik **File>Save>** diteruskan dengan memilih lokasi penyimpanan dan penulisan nama *file* misalnya **watak-OR.cir** (jangan lupa mencantumkan ekstensi **.cir**), serta diakhiri dengan klik **Save.**
   Antarmuka untuk penyimpanan *file* ditunjukkan pada gambar berikut ini.



Gambar 50. Antarmuka penyimpanan *file* rangkaian

9. Untuk menghentikan simulasi klik pada tanda ⏸ dan untuk reset tekan **backspace** pada *keyboard*.

Contoh 2: Coba susun rangkaian menggunakan *breadboard* virtual dari rangkaian pada gambar berikut ini!



Gambar 51.
Rangkaian logika untuk contoh ke-2 penggunaan simulator

Rangkaian tersebut mengandung tiga jenis gerbang logika yakni NOT (disediakan oleh IC 7404), AND (disediakan oleh IC 7408) dan OR (disediakan oleh IC 7432). Spesifikasi dan susunan *pinout* ketiga IC tersebut dapat dilihat pada tabel 1 di atas. Dari gambar rangkaian tersebut dapat dituliskan daftar komponen yang harus disediakan seperti disajikan pada tabel berikut ini.

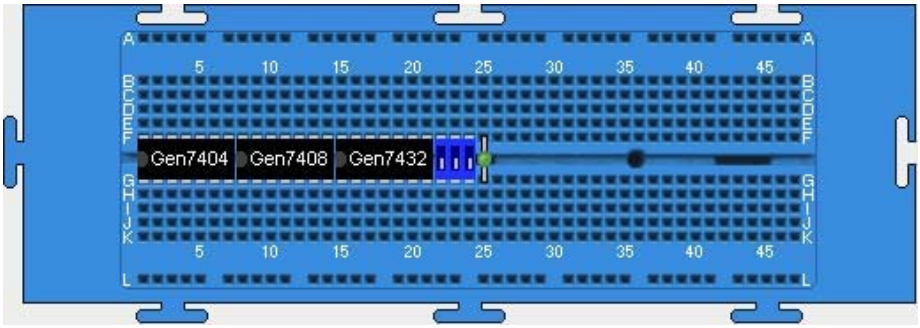
Tabel 9. Daftar komponen untuk rangkaian pada Gambar 51

Nama Komponen	Spesifikasi	Jumlah
Gerbang OR	IC TTL 7432	1 buah
Gerbang AND	IC TTL 7408	1 buah
Gerbang NOT	IC TTL 7404	1 buah
Saklar	DIP *Switch triple*	1 buah
LED	Warna hijau	1 buah

Berdasarkan rangkaian pada Gambar 51 dan daftar komponen yang diperlukan pada tabel 9, penyusunan rangkaian dapat dilakukan melalui cara:

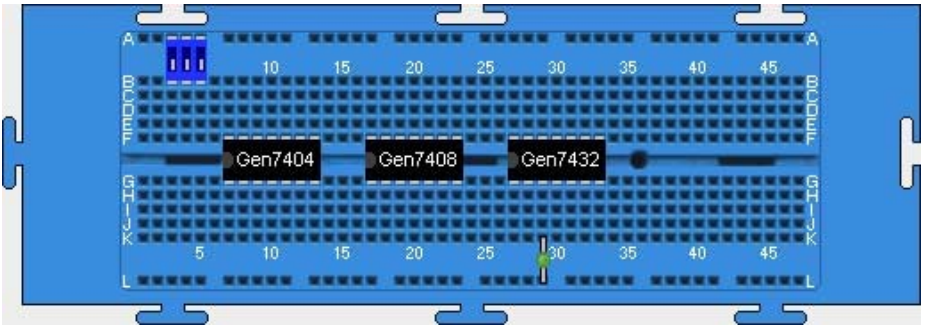
1. Jalankan simulator dan *load* papan *breadboard* dengan klik **Insert>Breadboard.**

2. Siapkan semua komponen yang diperlukan untuk menyusun rangkaian dengan klik **Insert>Chip>TTL>Logic>Gen7404>OK** untuk mengambil IC NOT 7404, klik **Insert>Chip>TTL>Logic> Gen7408>OK** untuk mengambil IC AND 7408, klik **Insert>Chip> TTL>Logic>Gen7432>OK** untuk mengambil IC OR 7432, klik **Insert>DIP Switches>Treble** untuk mengambil saklar DIP *triple*, dan klik **Insert>LED>Green** untuk mengambil LED warna hijau sehingga menghasilkan tampilan seperti gambar berikut ini.



Gambar 52.
Penyiapan komponen untuk rangkaian pada Gambar 51

3. Atur tata letak komponen misalnya seperti pada gambar berikut ini.

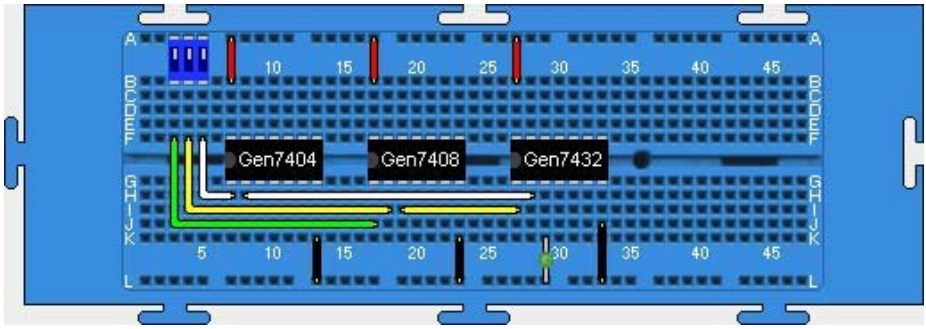


Gambar 53.
Tata letak komponen untuk rangkaian Gambar 51

4. Pasang catu daya +Vcc untuk ketiga IC menggunakan kabel merah. Untuk IC 7404, klik **Wires>Red>** **>titik B7>arahkan kabel ke A7 dan klik *double***, untuk IC 7408 klik **Wires>Red>** **>titik B17>arahkan kabel ke A17 dan klik *double***, dan untuk IC 7404 klik **Wires>Red>** **>titik B27>arahkan kabel ke A27 dan klik *double***.

5. Pasang catu daya ground (0 V) untuk ketiga IC menggunakan kabel hitam. Untuk IC 7404, klik **Wires>Black>** **>titik K13>arahkan kabel ke L13 dan klik *double***, untuk IC 7408 klik **Wires>Red>** **>titik K23>arahkan kabel ke L23 dan klik *double***, dan untuk IC 7404 klik **Wires>Red>** **>titik K33>arahkan kabel ke L33 dan klik *double***.

6. Dengan cara yang sama seperti di atas, pasang kabel untuk menghubungkan titik-titik sesuai rangkaian pada Gambar 51 sehingga menghasilkan gambar ini.



Gambar 54.
Susunan rangkaian dengan *breadboard* virtual untuk Gambar 51

7. Setelah rangkaian tersusun, jalankan simulasi dengan klik **Simulation>Run** atau klik ▶, diteruskan dengan mengubah-ubah posisi saklar. Rangkaian telah berhasil disusun dengan baik jika menghasilkan tabel kebenaran sebagai berikut.

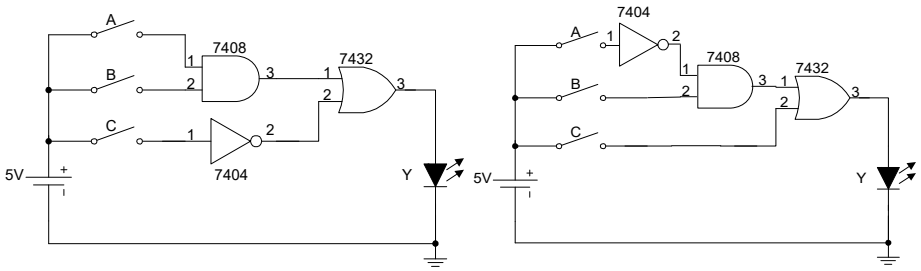
Tabel 10. Tabel kebenaran rangkaian Gambar 51

SAKLAR KIRI	SAKLAR TENGAH	SAKLAR KANAN	KONDISI LED
OFF	OFF	OFF	MENYALA
OFF	OFF	ON	PADAM
OFF	ON	OFF	MENYALA
OFF	ON	ON	PADAM
ON	OFF	OFF	MENYALA
ON	OFF	ON	PADAM
ON	ON	OFF	MENYALA
ON	ON	ON	MENYALA

8. Simpan rangkaian tersebut ke dalam harddisk menjadi *file* dengan klik **File>Save>** diteruskan dengan memilih lokasi penyimpanan dan penulisan nama *file* misalnya **kombinasi.cir** (jangan lupa mencantumkan ekstensi **.cir**), serta diakhiri dengan klik **Save.**

B. Memperbaiki Rangkaian

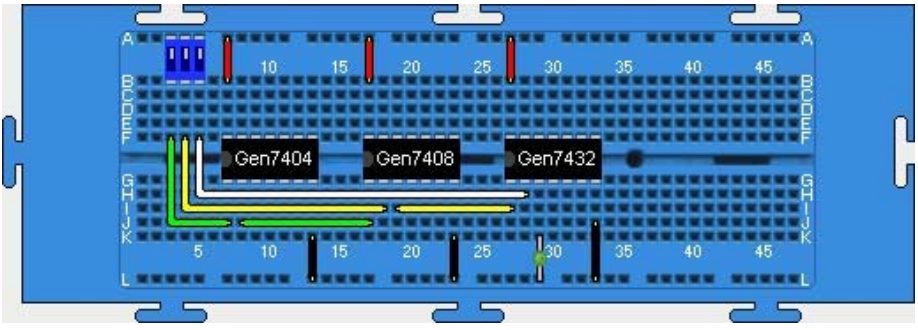
Rangkaian-rangkaian yang telah dibuat dan disimpan dalam suatu *file* dapat diambil dan diperbaiki. Rangkaian diambil dengan klik **File>Open> Nama File>Open. Nama File** merupakan nama-nama file dengan ekstensi **cir.** Coba buka file yang berisi rangkaian pada Gambar 51 (dalam panduan ini nama file adalah **kombinasi.cir**), lakukan perubahan berdasarkan rangkaian berikut ini.



Gambar 55. Rangkaian asli (kiri) dan hasil modifikasi (kanan)

Untuk mengubah rangkaian yang telah dibuat, lakukan prosedur sebagai berikut.

1. Jalankan simulator dan *load file* **kombinasi.cir.**
2. Hapus semua kabel putih dan hijau dengan **klik pada kabel putih>** ✕**, klik pada kabel hijau>** ✕
3. Hubungkan kembali kabel-kabel putih dan hijau sesuai dengan Gambar 55 sebelah kanan sehingga diperoleh gambar berikut ini.



Gambar 56.
Susunan rangkaian pada Gambar 55 kanan dengan *breadboard*

4. Jalankan simulasi dengan klik **Simulation>Run** atau klik  , diteruskan dengan mengubah-ubah posisi saklar. Jika rangkaian disusun dengan benar, maka akan menghasilkan tabel kebenaran sebagai berikut.

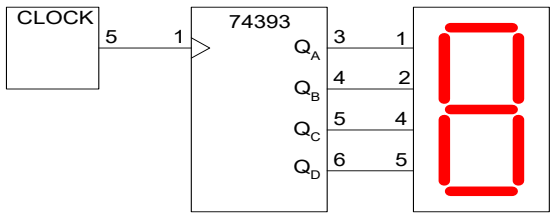
Tabel 11. Tabel kebenaran rangkaian Gambar 55 kanan

SAKLAR KIRI	SAKLAR TENGAH	SAKLAR KANAN	KONDISI LED
OFF	OFF	OFF	PADAM
OFF	OFF	ON	MENYALA
OFF	ON	OFF	MENYALA
OFF	ON	ON	MENYALA
ON	OFF	OFF	PADAM
ON	OFF	ON	MENYALA
ON	ON	OFF	PADAM
ON	ON	ON	MENYALA

5. Setelah selesai simulasi, lakukan reset dengan menekan tombol ***Backspace*** dan simpan rangkaian ke dalam *harddisk* dengan nama **kombinasi-2.cir**.

C. Penggunaan Multi *Breadboard*

Simulator ini memungkinkan penggunaan beberapa *breadboard* untuk menempatkan komponen-komponen dari suatu rangkaian. Perhatikan rangkaian pencacah (*counter*) berikut ini!

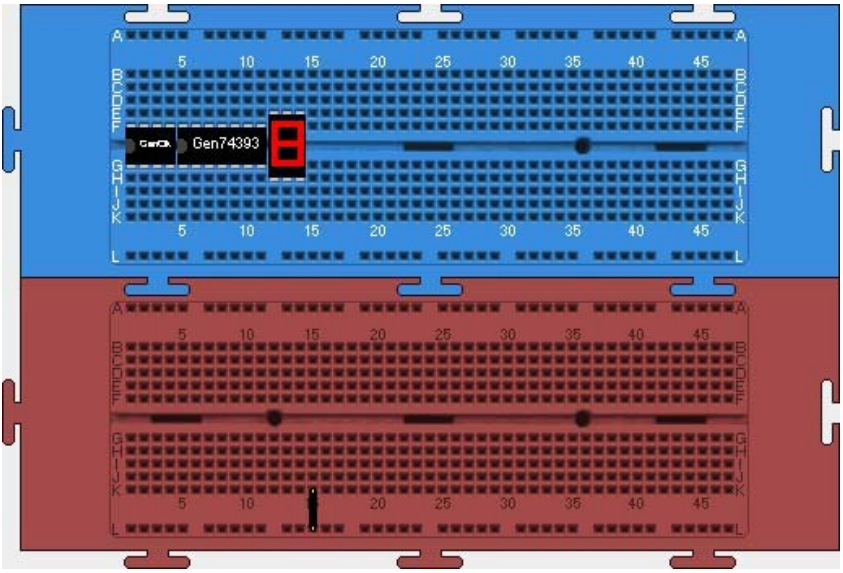


Gambar 57. Rangkaian *counter*

Jika rangkaian tersebut akan disusun menggunakan dua buah papan *breadboard,* maka langkah-langkahnya adalah sebagai berikut.

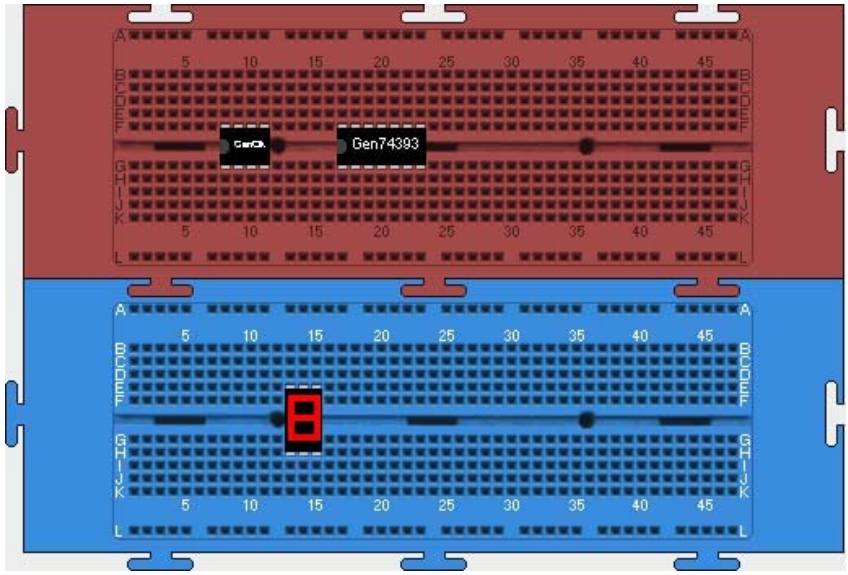
1. Jalankan simulator dan ambil papan *breadboard* dengan klik **Insert>Breadboard** atau klik  dua kali.

2. Ambil generator *Clock* dengan klik **Insert>Chip>Oscillator>OK**, chip 74393 dengan klik **Insert>Chip>TTL>Counter>Gen74393>OK**, dan display 7-segmen dengan klik **Insert>Chip>Components> HexDisplay>OK.**



Gambar 58. *Breadboard* ganda dan bahan rangkaian *counter*

3. Susun tata letak komponen seperti berikut ini.



Gambar 59. Tata letak komponen pada *breadboard* ganda

4. Lakukan pemasangan kabel sesuai dengan Gambar 57 di atas. Langkah awal, pasang terlebih dahulu kabel-kabel yang menghubungkan ketiga komponen ke terminal catu daya +Vcc menggunakan kabel merah dan ke terminal 0V (*ground*) menggunakan kabel hitam. Ikuti langkah-langkahnya seperti pada tabel berikut ini. Aktifkan terlebih dahulu *wiring mode* dengan klik  .

Tabel 12.

Langkah-langkah pemasangan kabel catu daya untuk Gambar 57

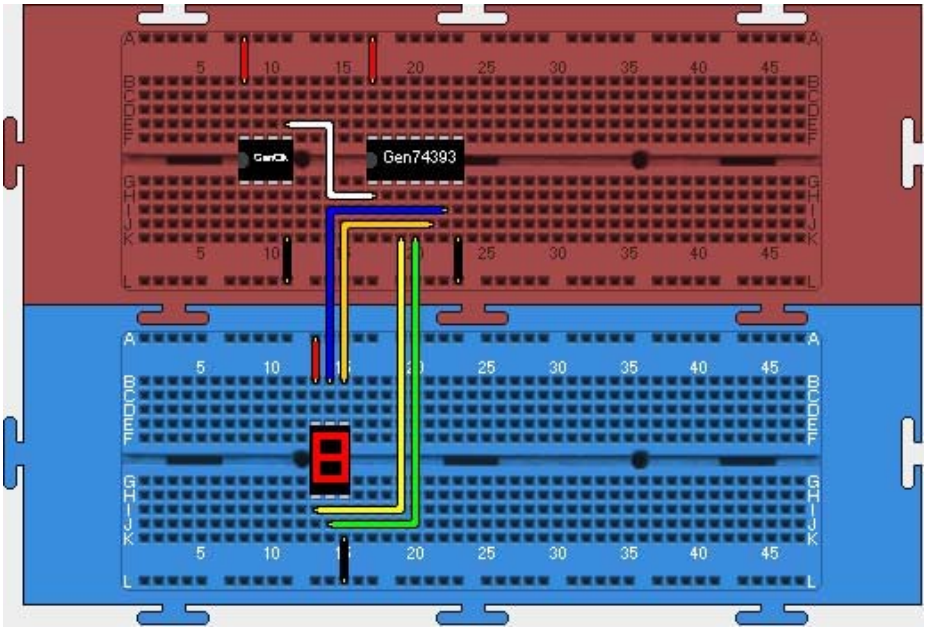
Hubungan Antar Titik	Warna Kabel	Cara
B8-A8 *breadboard 1*	Merah	Klik Wire>Red> >Titik B8>klik double pada A8
B17-A17 *breadboard 1*	Merah	Klik Wire>Red> >Titik B17>klik double pada A17
B13-A13 *breadboard 2*	Merah	Klik Wire>Red> >Titik B13>klik double pada A13
K11-L11 *breadboard 1*	Hitam	Klik Wire>Black> >Titik K11>klik double pada L11
K23-L23 *breadboard 1*	Hitam	Klik Wire>Black> >Titik K23>klik double pada L23
K15-L15 *breadboard 2*	Hitam	Klik Wire>Black> >Titik K15>klik double pada L15

5. Selanjutnya, lakukan pemasangan kabel-kabel antar komponen dan ikuti langkah-langkahnya seperti pada tabel berikut ini.

Tabel 13.
Langkah-langkah pemasangan kabel antar komponen Gambar 57

Hubungan Antar Titik	Warna Kabel	Cara
E11-H17 *breadboard 1*	Putih	Pada breadboard 1: Klik Wire>White>Titik E11 >Titik E14>Titik H14>klik double pada H17
K19 (*breadboard 1*)-I13 (*breadboard 2*)	Kuning	Klik Wire>Yellow>Titik K19 (Breadboard 1)>Titik I19 (breadboard 2)>double pada I13 (breadboard 2)
K20 (*breadboard 1*)-J14 (*breadboard 2*)	Hijau	Klik Wire>Green>Titik K20 (Breadboard 1)>Titik J20 (breadboard 2)> double pada J14 (breadboard 2)
J21 (*breadboard 1*)-B15 (*breadboard 2*)	Orange	Klik Wire>Orange>Titik J21 (Breadboard 1)>Titik J15 (breadboard 1)> double pada B15 (breadboard 2)
I22 (*breadboard 1*)-B14 (*breadboard 2*)	Biru	Klik Wire>Blue>Titik I22 (Breadboard 1)>Titik I14 (breadboard 1)> double pada B14 (breadboard 2)

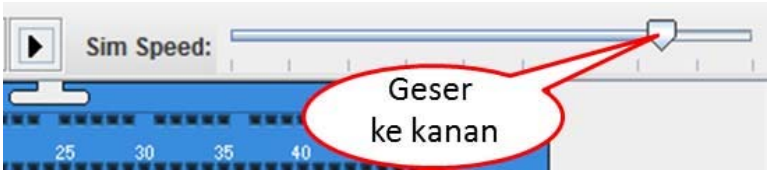
Jika pemasangan kabel dilakukan dengan benar maka akan dihasilkan gambar sebagai berikut.



Gambar 60.

Susunan rangkaian Gambar 57 menggunakan *breadboard* ganda

6. Untuk mencoba keberhasilan rangkaian *counter* yang telah disusun, jalankan simulasi dengan klik **Simulation>Run** atau klik ▶. Coba atur kecepatan simulasi dengan menggeser tombolnya ke kanan agar rangkaian *counter* menjadi cepat jalannya.



Gambar 61. Cara mempercepat proses simulasi

7. Rangkaian *counter* pada Gambar 57 telah berhasil disusun dengan benar jika hasil simulasinya menunjukkan *display* 7-segmennya dapat menampilkan angka heksadesimal dari 0 sampai dengan F, kembali ke 0 dan seterusnya.

8. Simpan rangkaian tersebut ke dalam file dengan nama **pencacah.cir.** Untuk mengakhiri simulasi tekan tombol **Backspace** pada *keyboard*.

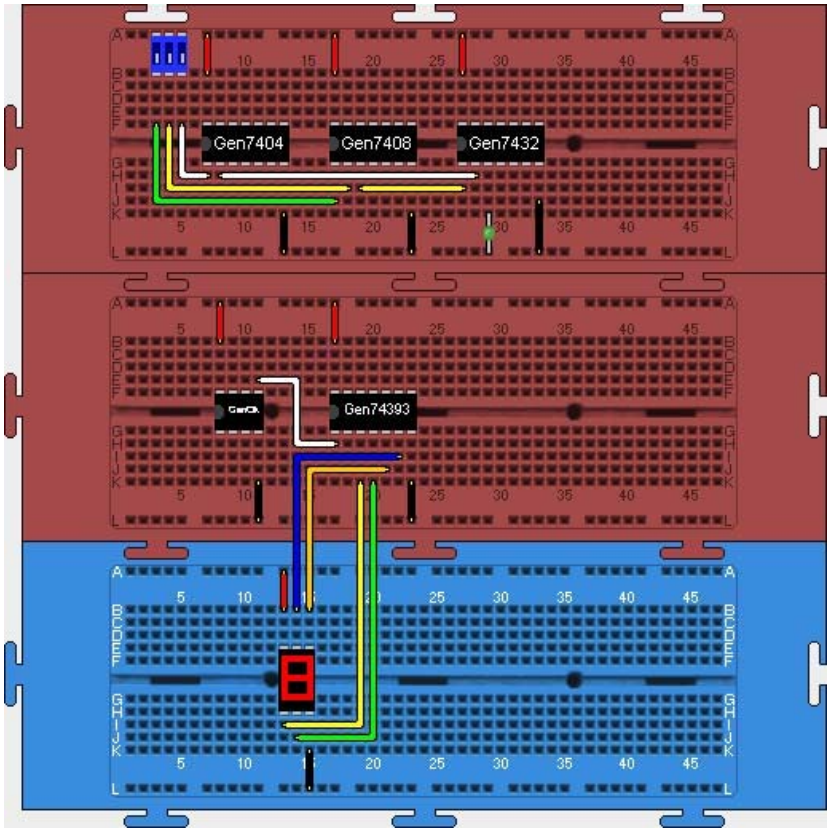
D. Menyisipkan Rangkaian

Simulator ini juga memungkinkan penggunanya dapat membuka beberapa *file* rangkaian dalam sebuah layar. Misal akan dibuka dua buah *file* **kombinasi.cir** dan **pencacah.cir** dalam satu layar, maka langkah-langkah adalah sebagai berikut.

1. Buka file **kombinasi.cir** dengan klik **File>Open>Yes>Pilih file kombinasi.cir>Open.**

2. Sisipkan file pencacah dengan klik **File>Insert Circuit>Pilih file pencacah.cir>Open.**

3. Jika langkah-langkah tersebut dilakukan dengan benar, maka akan diperoleh tampilan pada layar seperti pada gambar berikut ini.



Gambar 62. Pembukaan beberapa *file* rangkaian dalam satu layar

4. Walaupun beberapa rangkaian ditampilkan dalam satu layar, namun setiap rangkaian berdiri sendiri-sendiri dan tidak saling mempengaruhi. Dengan cara seperti ini memungkinkan pengguna dapat bekerja dengan berbagai topik eksperimen secara simultan sehingga dari sisi waktu menjadi lebih efisien.

5. Pada Gambar 62, terlihat bahwa dua buah rangkaian telah disediakan dalam satu layar yakni rangkaian dalam *file* **kombinasi.cir** (*breadboard 1*) dan rangkaian dalam *file* **pencacah.cir** (*breadboard 2* dan *breadboard* 3).

6. Coba jalankan simulasi untuk mencoba kedua rangkaian tersebut!

7. Rangkaian-rangkaian yang telah ditampilkan dalam satu layar ini juga dapat disimpan menjadi rangkaian baru ke dalam sebuah *file*. Simpan rangkaian yang telah digabung ini dengan nama **gabungan.cir.**

Bagian V

Aplikasi Simulator *Breadboard* untuk Praktikum Teknik Digital

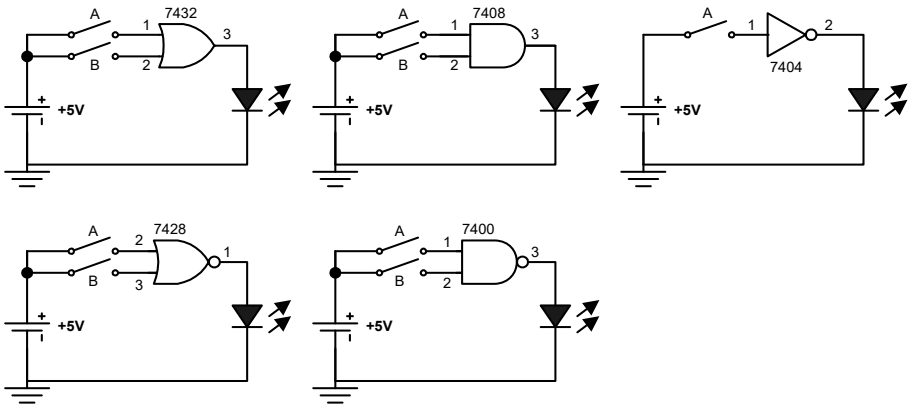
Uraian pada bab-bab sebelumnya menunjukkan bahwa kemampuan simulator *breadboard* sangat memadai dalam menyediakan piranti-piranti digital secara virtual. Selain itu, cara pengoperasian simulator ini juga sangat mudah dan memberikan kesan seperti menggunakan peralatan real, sehingga layak digunakan sebagai perangkat praktikum teknik digital. Bab ini mendeskripsikan contoh-contoh penerapan simulator *breadboard* untuk praktikum teknik digital.

A. Watak Gerbang Logika Dasar

Gerbang logika dasar OR, AND, NOT dan gerbang universal NOR dan NAND merupakan elemen-elemen yang mendasari perancangan suatu rangkaian digital. Oleh sebab itu, pemahaman secara komprehensif terhadap watak setiap gerbang-gerbang logika tersebut sangat diperlukan sebelum kita mempelajari lebih lanjut analisis dan perancangan rangkaian digital. Praktikum dengan topik ini ditujukan untuk: (1) menentukan watak gerbang-gerbang logika dasar secara eksperimen; dan (2) membuktikan universalitas gerbang NOR dan NAND.

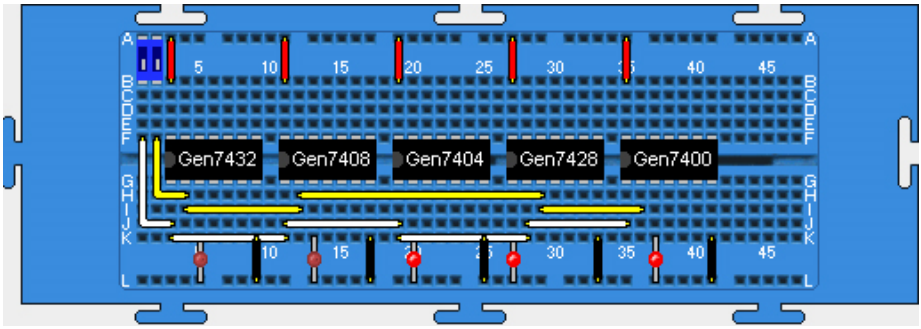
1. Menyelidiki Watak Gerbang OR, AND, NOT, NAND, dan NOR

Rangkaian-rangkaian logika untuk menyelidiki watak gerbang logika dasar ditunjukkan pada gambar berikut ini.



Gambar 63.
Rangkaian untuk menentukan watak gerbang logika dasar

Berdasarkan Gambar 63 terlhat bahwa piranti yang dibutuhkan untuk menyusun rangkaian adalah 5 jenis IC yakni seri 7432 (OR), 7408 (AND), 7404 (NOT), 7428 (NOR) dan 7400 (NAND), 5 buah LED, dan sebuah saklar ganda. Tempatkan semua piranti tersebut di atas papan *breadboard*, selanjutnya gunakan kabel-kabel penghubung untuk mewujudkan rangkaian yang diinginkan. Untuk mengetahui susunan pin pada setiap IC yang akan digunakan, anda dapat merujuk ke Tabel 1 di muka. Salah satu dari wujud rangkaian untuk Gambar 63 ditunjukkan pada Gambar 64.



Gambar 64.
Rangkaian untuk menyelidiki watak gerbang logika dasar menggunakan simulator *breadboard*

Berikan kombinasi input yang mungkin melalui kedua saklar yang ada dan amati output tiap rangkaian. Jika percobaan anda benar, maka akan diperoleh tabel kebenaran seperti pada tabel berikut ini. Catatan: input 0 mewakili tegangan rendah dan 1 tegangan tinggi. Untuk output, 0 mewakili LED padam dan 1 LED menyala.

Tabel 14. Watak gerbang logika dasar

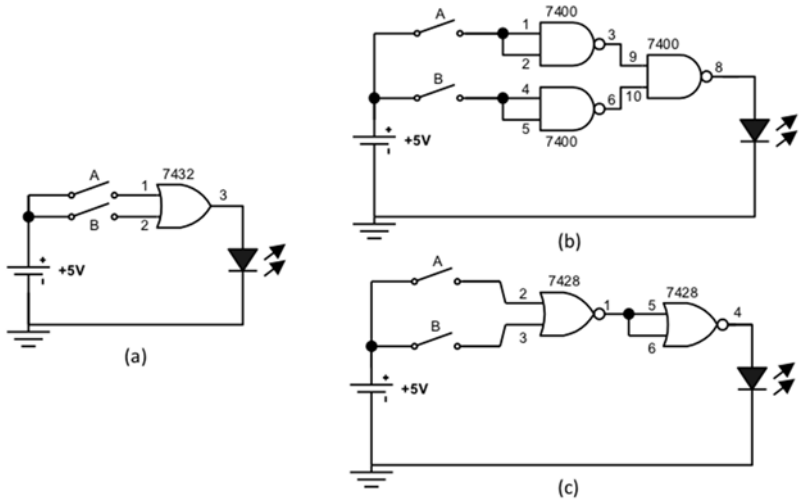
INPUT		OUTPUT				
A	B	OR	AND	NOT A	NOR	NAND
0	0	0	0	1	1	1
0	1	1	0	1	0	1
1	0	1	0	0	0	1
1	1	1	1	0	0	0

2. Membuktikan Universalitas Gerbang NAND dan NOR

Sifat universal gerbang NOR dan NAND dibuktikan dengan menunjukkan bahwa gerbang OR, AND dan gerbang NOT dapat disusun dengan menggunakan rangkaian-rangkaian yang mengandung hanya gerbang NAND saja atau NOR saja.

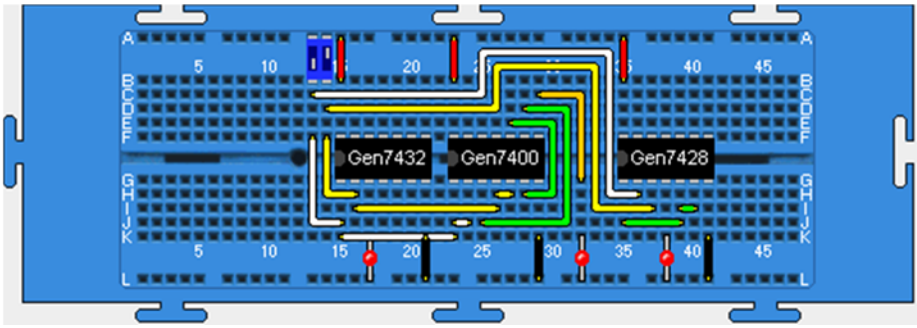
a. Menyusun Gerbang OR dari Gerbang NAND atau Gerbang NOR

Rangkaian untuk menunjukkan bahwa gerbang OR dapat dibuat dari gerbang NAND atau NOR ditunjukkan pada gambar berikut ini.



Gambar 65.
Rangkaian: (a) OR asli; (b) OR menggunakan NAND;
dan (c) OR menggunakan NOR

Piranti-piranti yang dibutuhkan untuk menyusun rangkaian pada Gambar 65 adalah: 3 buah IC yakni 7432 (OR), 7400 (NAND), 7428 (NOR), 3 buah LED, dan sebuah saklar ganda. Salah satu bentuk susunan rangkaian menggunakan simulator *breadboard* untuk Gambar 65 ditunjukkan pada gambar berikut ini.

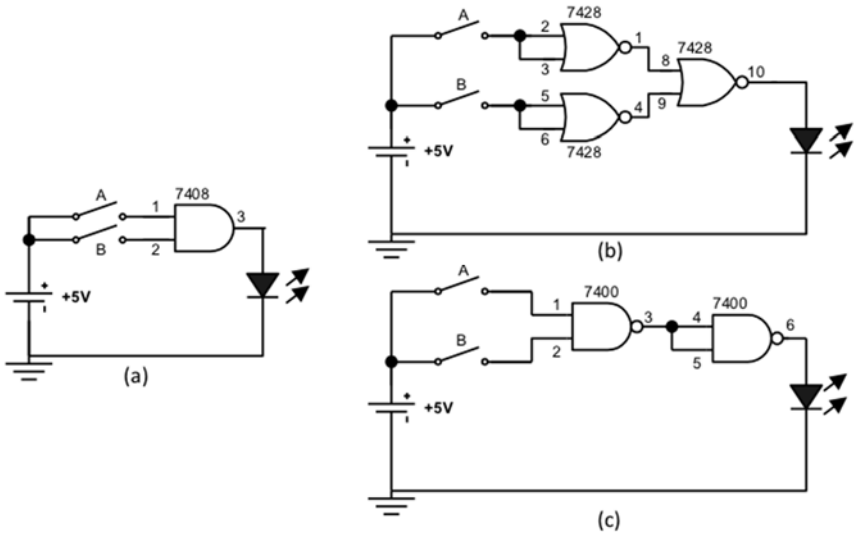


Gambar 66.
Rangkaian untuk membuktikan gerbang OR dapat dibuat
dari gerbang NAND atau gerbang NOR
menggunakan simulator *breadboard*

Lakukan eksperimen dengan memasukkan keadaan ON dan OFF melalui saklar yang ada dan amati ouputnya. Susunlah tabel kebenaran hasil eksperimen anda dan perhatikan apakah output rangkaian Gambar 65(a) sama dengan output rangkaian Gambar 65(b) dan Gambar 65(c)? Jika anda benar dalam melakukan eksperimen, maka akan diperoleh tabel kebenaran yang menunjukkan bahwa output ketiga rangkaian sama. Hal itu menunjukkan bahwa gerbang OR dapat disusun dengan gerbang NAND atau NOR.

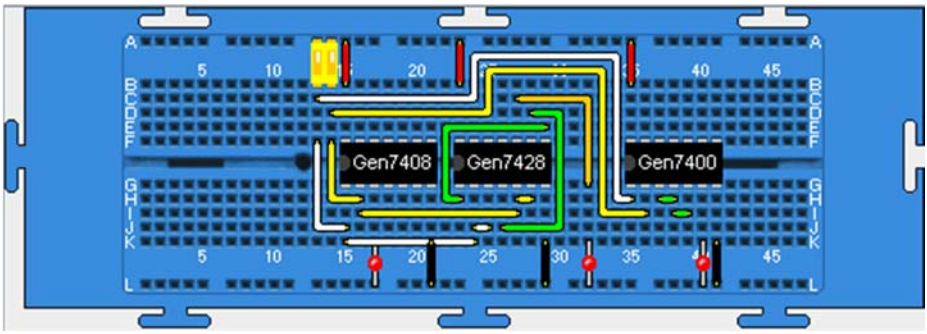
b. Menyusun Gerbang AND dari Gerbang NAND atau Gerbang NOR

Rangkaian untuk membuktikan bahwa gerbang AND dapat disusun dengan NAND atau NOR ditunjukkan pada gambar berikut ini.



Gambar 67.
Rangkaian: (a) AND asli; (b) AND menggunakan NOR;
dan (c) AND menggunakan NAND

Piranti-piranti yang dibutuhkan untuk menyusun rangkaian pada Gambar 67 adalah 3 buah IC yakni 7408 (AND), 7400 (NAND), 7428 (NOR), 2 buah LED dan sebuah saklar ganda. Salah satu bentuk susunan rangkaian dengan menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.

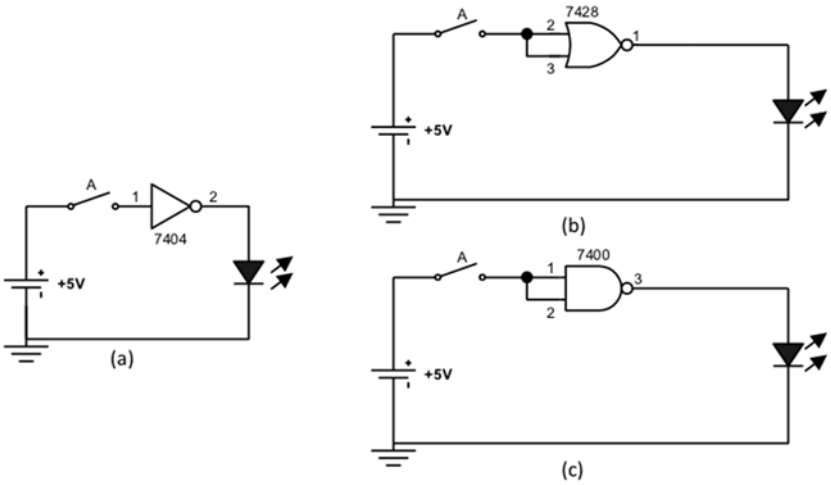


Gambar 68.
Rangkaian untuk membuktikan gerbang AND
dapat dibuat dari gerbang NAND atau gerbang NOR
menggunakan simulator *breadboard*

Coba lakukan percobaan dengan memasukkan kombinasi input yang mungkin melalui saklar, dan amati keadaan output rangkaian serta susun tabel kebenarannya. Apakah output semua rangkaian sama untuk setiap keadaan input yang diberikan? Jika anda benar dalam melakukan percobaan tersebut maka akan diperoleh keadaan semua output rangkaian sama untuk setiap input yang diberikan. Hal itu menunjukkan bahwa gerbang AND dapat dibuat dari gerbang NAND atau gerbang NOR.

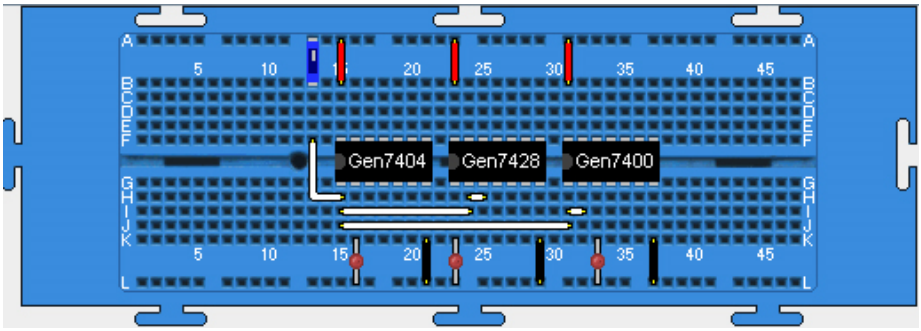
c. Menyusun Gerbang NOT dari Gerbang NAND atau Gerbang NOR

Gerbang NOT dapat disusun dengan menggunakan gerbang NAND atau gerbang NOR seperti ditunjukkan pada gambar berikut ini.



Gambar 69.
Rangkaian: (a) NOT asli; (b) NOT menggunakan NOR;
dan (c) NOT menggunakan NAND

Untuk mewujudkan rangkaian pada Gambar 69 dibutuhkan piranti-piranti: 3 buah IC yakni 7404 (NOT), 7400 (NAND), 7428 (NOR), 2 buah LED dan sebuah saklar tunggal. Siapkan semua piranti di atas papan *breadboard* dan gunakan kabel-kabel penghubung untuk mewujudkan rangkaian. Salah satu bentuk rangkaian NOT berbagai versi menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.



Gambar 70.
Rangkaian untuk membuktikan gerbang NOT
dapat dibuat dari Gerbang NAND atau Gerbang NOR

Berikan sinyal 1 dan 0 pada inputnya dan amati keadaan outputnya. Apakah output ketiganya memberikan keadaan yang sama untuk setiap keadaan input yang diberikan. Jika eksperimen dilakukan dengan benar, maka anda akan memperoleh keadaan bahwa output rangkain-rangkaian tersebut sama, yang menunjukkan bahwa gerbang NOT dapat dibuat dari gerbang NAND atau gerbang NOR.

B. Minimalisasi Rangkaian Logika

Perancangan rangkaian logika yang kompleks akan melibatkan banyak gerbang logika dasar sebagai pembentuk rangkaian yang dirancang. Untuk keperluan efisiensi, baik dari segi biaya maupun teknis perancangan, seringkali dilakukan penyederhanaan atau minimalisasi rangkaian logika. Prosedur minimalisasi merupakan salah satu azas perancangan rangkaian logika yang selalu digunakan untuk memperoleh rancangan rangkaian yang kompak dan murah. Prosedur ini dapat mencakup minimalisasi dalam jumlah gerbang maupun dalam hal jenis gerbang yang dilibatkan.

Praktikum dengan topik ini ditujukan untuk menunjukkan bahwa watak rengkaian logika yang dirancang dengan jumlah dan jenis gerbang minimum tetap memberikan watak yang sama dengan watak rangkaian aslinya (rangkaian dengan jumlah dan jenis gerbang yang banyak). Melalui

praktikum ini anda dapat meningkatkan pemahaman dan keterampilan dalam merancang rangkaian logika dengan jumlah gerbang minimum, dan merancang rangkaian logika dengan satu jenis gerbang saja.

1. Merancang Rangkaian dengan Jumlah Gerbang Minimum

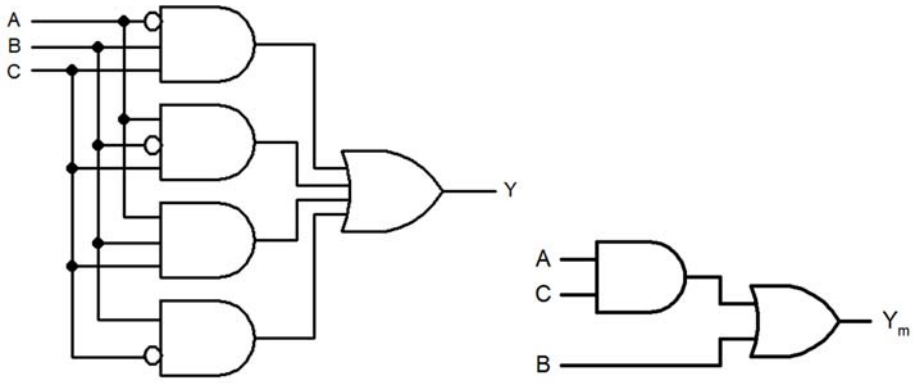
Misal diketahui rangkaian digital yang memiliki persamaan logika $Y = \overline{A}BC + A\overline{B}C + ABC + B\overline{C}$, coba lakukan minimalisasi menggunakan Aljabar Boole atau Peta Karnaugh. Jika proses minimalisasi yang anda jalankan benar, maka akan diperoleh persamaan yang lebih sederhana dengan watak yang sama yakni $Y_m = AC + B$. Secara teoritis, kita dapat membuat tabel kebenaran dari kedua rangkaian tersebut sebagai berikut.

Tabel 15.

Watak rangkaian $Y = \overline{A}BC + A\overline{B}C + ABC + B\overline{C}$ dan $Y_m = AC + B$

INPUT			OUTPUT TIAP GERBANG								OUTPUT	
A	B	C	$\overline{A}$	$\overline{B}$	$\overline{C}$	$\overline{A}BC$	$A\overline{B}C$	ABC	$B\overline{C}$	AC	Y	Y_m
0	0	0	1	1	1	0	0	0	0	0	0	0
0	0	1	1	1	0	0	0	0	0	0	0	0
0	1	0	1	0	1	0	0	0	1	0	1	1
0	1	1	1	0	0	1	0	0	0	0	1	1
1	0	0	0	1	1	0	0	0	0	0	0	0
1	0	1	0	1	0	0	1	0	0	1	1	1
1	1	0	0	0	1	0	0	0	1	0	1	1
1	1	1	0	0	0	0	0	1	0	1	1	1

Dari Tabel 15 terlihat bahwa watak Y_m (rangkaian yang lebih sederhana) sama dengan watak Y (rangkaian asli dengan banyak gerbang). Coba gambarkan kedua rangkaian tersebut! Jika anda benar dalam menggambar, maka dapat dihasilkan gambar rangkaian asli $Y = \overline{A}BC + A\overline{B}C + ABC + B\overline{C}$ dan rangkaian setelah disederhanakan $Y_m = AC + B$ seperti berikut ini.

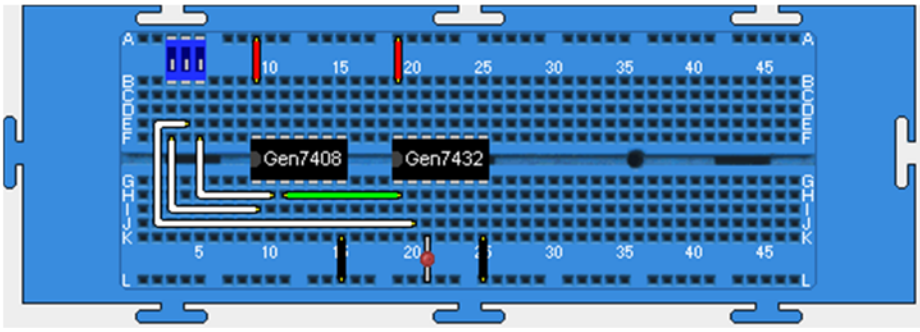


(a) (b)

Gambar 71.

Minimalisasi rangkaian: (a) $Y = \overline{A}BC + A\overline{B}C + ABC + B\overline{C}$;
(b) hasil penyederhanaan $Y_m = AC + B$

Selanjutnya, coba susun rangkaian yang telah disederhanakan yakni $Y_m = AC + B$ menggunakan simulator *breadboard*! Jika anda benar dalam menyusun rangkaian tersebut, maka susunan rangkaian dapat berbentuk seperti pada gambar berikut ini.

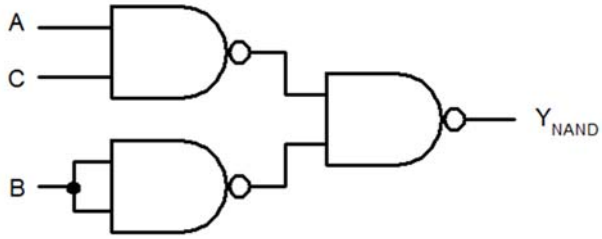


Gambar 72. Rangkaian dalam bentuk minimum $Y_m = AC + B$ menggunakan simulator *breadboard*

Coba lakukan percobaan untuk membuktikan bahwa watak rangkaian Ym secara eksperimen sama dengan watak rangkaian Y. Jika eksperimen anda benar maka akan diperoleh keadaan output seperti pada Tabel 15 di atas pada kolom yang berwarna biru.

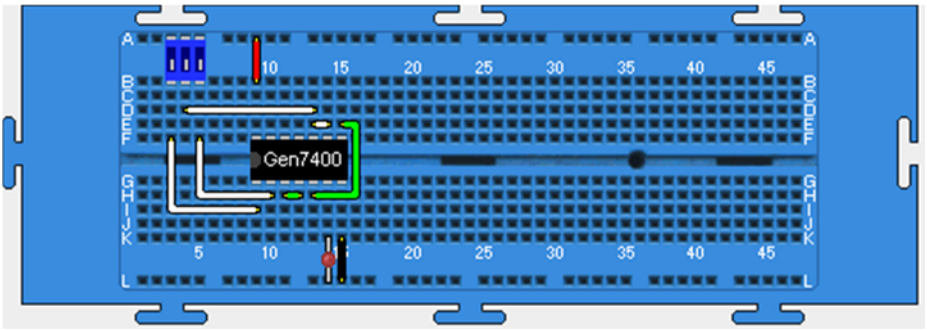
2. Merancang Rangkaian dengan Satu Jenis Gerbang Saja

Ubah hasil minimalisasi $Y = \overline{A}BC + A\overline{B}C + ABC + B\overline{C}$ yakni $Y_m = AC + B$ ke dalam bentuk NAND saja! Jika proses yang anda jalankan benar maka akan diperoleh persamaan $Y_{NAND} = \overline{\overline{AC} \cdot \overline{B}}$ dan rangkaian dalam bentuk NAND seperti pada gambar berikut ini.



Gambar 73. Rangkaian hasil minimalisai menggunakan NAND saja

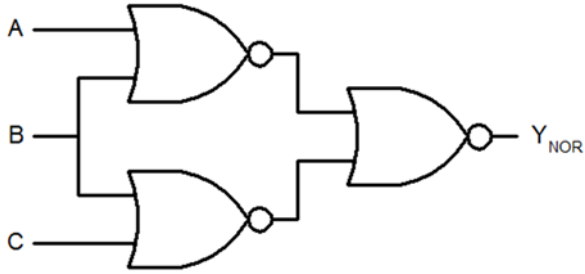
Selanjutnya susun rangkaian pada Gambar 73 menggunakan simulator *breadboard*! Jika anda benar dalam menyusun rangkaian, maka dapat diperoleh susunan rangkaian seperti berikut ini.



Gambar 74. Rangkaian dalam bentuk minimum $Y_{NAND} = \overline{\overline{AC} \cdot \overline{B}}$ menggunakan simulator *breadboard*

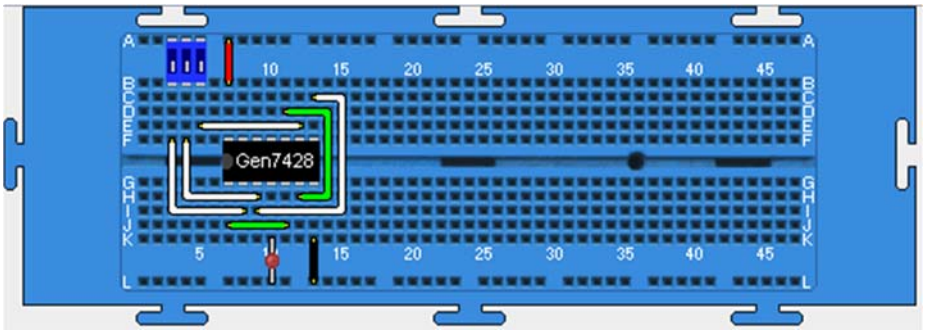
Lakukan eksperimen untuk memperoleh tabel kebenaran rangkaian tersebut! Pelaksanaan eksperimen yang benar akan menghasilkan tabel kebenaran dengan keadaan output seperti pada Tabel 15 kolom berwarna biru. Hal itu menunjukkan bahwa rangkaian yang kompleks dalam bentuk $Y = \overline{A}BC + A\overline{B}C + ABC + B\overline{C}$ (terdiri atas 3 jenis gerbang OR, AND, NOT) dengan jumlah gerbang total sebanyak 8 buah, dapat disederhanakan hanya dengan satu jenis gerbang saja yakni NAND.

Rancangan rangkaian logika hanya dengan satu gerbang saja juga dapat dilakukan dengan menggunakan gerbag NOR. Untuk membuat rangkaian tersebut, ubah persamaan $Y_m = AC + B$ menjadi bentuk NOR. Jika langkah yang anda jalankan benar, maka akan diperoleh persamaan $Y_{NOR} = \overline{\overline{(A+B)}} + \overline{\overline{(B+C)}}$. Rangkaian untuk persamaan tersebut ditunjukkan pada berikut ini.



Gambar 75. Rangkaian hasil minimalisasi menggunakan NOR saja

Susun rangkaian tersebut ke dalam simulator dengan terlebih dahulu menempatkan IC 7428, saklar *triple* dan sebuah LED di atas *breadboard.* Jika anda benar dalam menyusunnya, akan dihasilkan salah satu bentuk susunan rangkaian $Y_{NOR} = \overline{\overline{(A+B)}} + \overline{\overline{(B+C)}}$ di atas *breadboard* seperti pada gambar berikut ini.



Gambar 76.

Rangkaian dalam bentuk minimum $Y_{NOR} = \overline{\overline{(A+B)} + \overline{(B+C)}}$ menggunakan simulator *breadboard*

Coba lakukan percobaan untuk memperoleh tabel kebenaran rangkaian tersebut! Jika anda benar dalam melakukan percobaan, maka akan diperoleh tabel kebenaran dengan keadaan output seperti pada Tabel 15 kolom berwarna biru. Hal itu menunjukkan bahwa rangkaian yang kompleks dengan banyak gerbang yakni $Y = \overline{A}BC + A\overline{B}C + ABC + B\overline{C}$ (terdiri atas 3 jenis gerbang OR, AND, NOT) dapat disusun hanya dengan menggunakan satu gerbang NOR saja. Perhatikan! Jika anda menyusun rangkaian dengan menggunakan bentuk aslinya akan diperlukan 3 jenis gerbang dengan 3 buah IC, sedangkan jika disusun dalam bentuk NOR hanya memerlukan 1 buah IC.

C. Komparator dan Penjumlah Biner

Rangkaian komparator atau pembanding biner adalah salah satu rangkaian logika kombinasi, yakni rangkaian logika yang keluarannya suatu saat tidak tergantung pada nilai keluaran saat yang lalu. Komparator biner berfungsi membandingkan dua data biner pada inputnya.

Terdapat 2 jenis rangkaian komparator 1-bit yakni *non-equality comparator* dan *equality comparator*. *Non-equality comparator* merupakan rangkaian logika yang memberikan keadaan output tinggi (logika 1) jika keadaan input-inputnya berbeda, dan *equality comparator* akan

memberikan keadaan output tinggi jika keadaan input-inputnya sama. Kecuali dapat diperoleh dengan cara membangun menggunakan gerbang logika dasar, *non-equality comparator* juga dapat diperoleh dalam bentuk gerbang exclusive-OR (EX-OR) dan *equality comparator* dalam bentuk gerbang exclusive-NOR (EX-NOR), keduanya tersedia dalam bentuk IC.

Selain komparator, jenis rangkaian logika kombinasi lainnya adalah penjumlah biner (*binary adder*). Rangkaian penjumlah biner merupakan penerapan dari rangkaian komparator biner. Rangkaian ini sangat penting peranannya di dalam bidang komputer digital, karena operasi-operasi aritmatika yang lebih kompleks seperti perkalian hakekatnya merupakan penjumlahan yang diulang-ulang sehingga memerlukan rangkaian penjumlah dan pembagian adalah pengurangan yang diulang-ulang sehingga memerlukan rangkaian pengurang.

Praktikum dengan topik ini ditujukan untuk menyelidiki watak komparator 1-bit yang disusun dengan menggunakan gerbang logika dasar maupun komparator dalam bentuk IC TTL. Selain itu, melalui praktikum ini, anda juga dapat merancang dan menentukan watak *half adder* dan *full adder* 1-bit serta *parallel full* adder 4-bit.

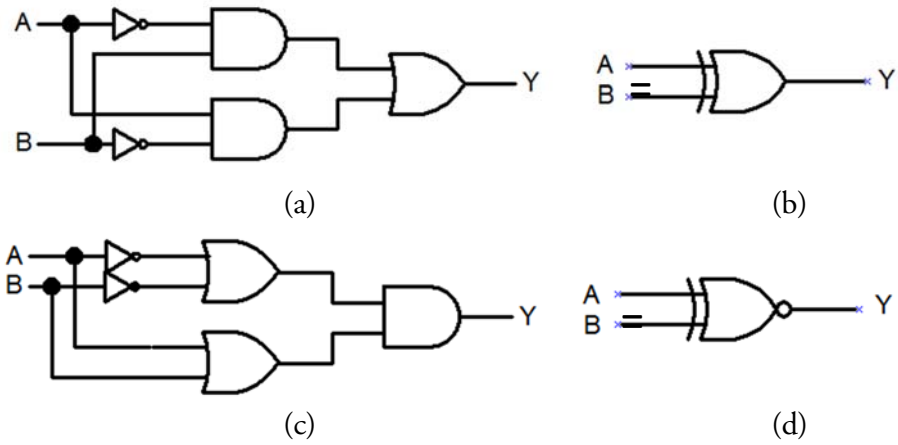
1. Menentukan Watak *Non-Equality* dan *Equality Comparator*

Berdasarkan definisinya, coba susun tabel kebenaran dan persamaan logika untuk rangkaian *non-equality* dan *equality comparator*! Jika anda benar dalam menurunkannya, akan dihasilkan tabel kebenaran dan persamaan logika sebagai berikut.

Tabel 16.
Tabel kebenaran *non-equality comparator* (XOR)
dan *equality comparator* (XNOR)

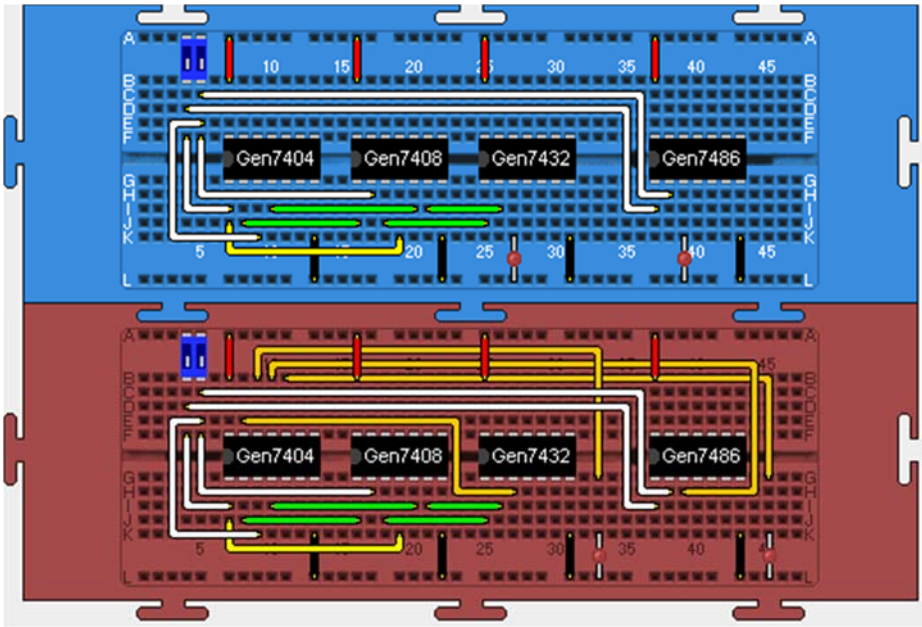
INPUT		OUTPUT	
A	B	$Y_{XOR} = \overline{A}B + A\overline{B}$	$Y_{XNOR} = (A + \overline{B})(\overline{A} + B)$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

Sesuai dengan persamaan pada Tabel 16, untuk menjalankan percobaan ini, anda dapat menggunakan rangkaian sebagai berikut.



Gambar 77.
Rangkaian *comparator*: (a) *non-equality*; (b) *XOR*;
(c) *equality;* dan (d) XNOR

Coba susun rangkaian *non-equality comparator* dan XOR pada Gambar 77(a) dan Gambar 77(b) di atas *breadboard*! Sediakan terlebih dahulu IC 7404 (NOT), 7408 (AND), IC 7432 (OR), 7486 (XOR), saklar ganda, dan 2 buah LED, serta tempatkan piranti-piranti tersebut di atas *breadboard.* Selanjutnya susun koneksi antar titik sesuai dengan rangkaian pada Gambar 77. Ambil satu *breadboard* lagi dan susun rangkaian *equality comparator* seperti pada Gambar 77(c) dan Gambar 77(d) di atasnya! Jika anda benar dalam menyusun kedua rangkaian tersebut, akan dihasilkan salah satu susunan rangkaian seperti gambar berikut ini.

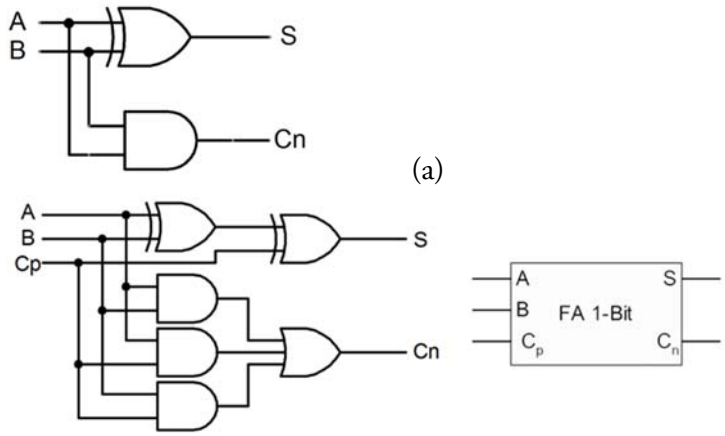


Gambar 78.
Susunan *non-equality comparator* (*breadboard* biru) dan *equality comparator* (breadboard merah)

Coba anda berikan kombinasi nilai logika yang mungkin pada input kedua rangkaian melalui saklar logika dan amati outputnya serta catat hasilnya ke dalam tabel pengamatan! Bandingkan hasilnya dengan Tabel 16. Jika percobaan anda benar, akan dihasilkan tabel kebenaran yang sama dengan tabel watak *non-equality comparator* dan *equality comparator* di atas.

2. Menentukan Watak *Half-Adder* 1-Bit dan *Full-Adder* 1-Bit

Rangkaian *half-adder* dan *full-adder* 1-bit ditunjukkan pada gambar berikut ini.



Gambar 79. Rangkaian *adder* 1-bit: (a) *half-adder*; (b) *full-adder*; dan (c) simbol *full-adder*

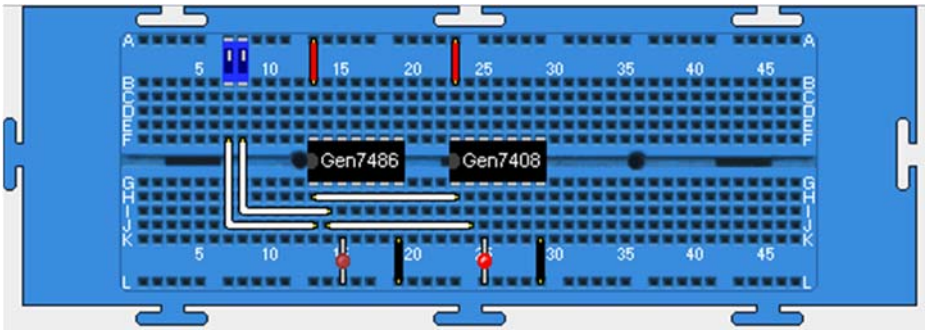
Berdasarkan rangkaian tersebut, dapat disusun tabel kebenaran untuk kedua rangkaian di atas sebagai berikut.

Tabel 17.
Watak (tabel kebenaran) *half-adder* (kiri) dan *full-adder* (kanan)

INPUT		OUTPUT	
A	B	S	Cn
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

INPUT			OUPUT	
A	B	Cp	S	Cn
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

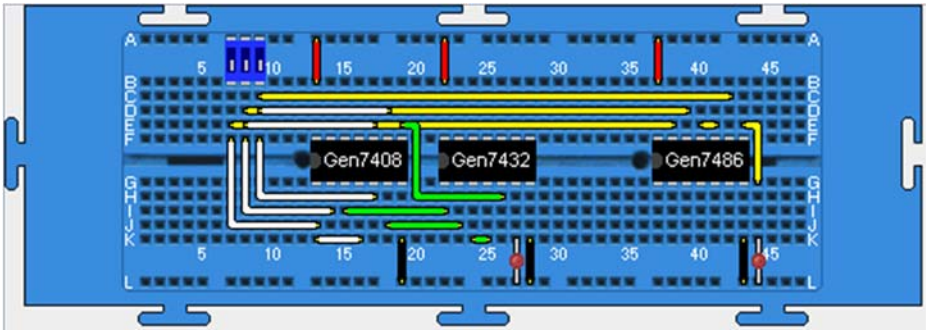
Untuk menyusun rangkaian *half-adder* pada Gambar 79, tempatkan IC 7486 (XOR), 7408 (AND). Pasang catu daya pada semua IC yang digunakan, pasang saklar logika pada input rangkaian dan indikator LED pada output rangkaian. Salah satu bentuk susunan rangkaian untuk menyelidiki watak *half-adder* menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.



Gambar 80. Rangkaian *half-adder* menggunakan simulator *breadboard*

Melalui saklar logika, berikan kombinasi nilai logika yang mungkin pada input rangkaian dan amati outputnya serta catat hasilnya ke dalam tabel pengamatan.

Selanjutnya, coba lakukan penyusunan rangkaian *full-adder* dengan terlebih dahulu menempatkan IC 7408 (AND), 7432 (OR) dan 7486 (XOR), sebuah saklar *triple,* dan 2 buah LED di atas *breadboard*. Salah satu bentuk susunan rangkaian *full-adder* 1-bit menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.

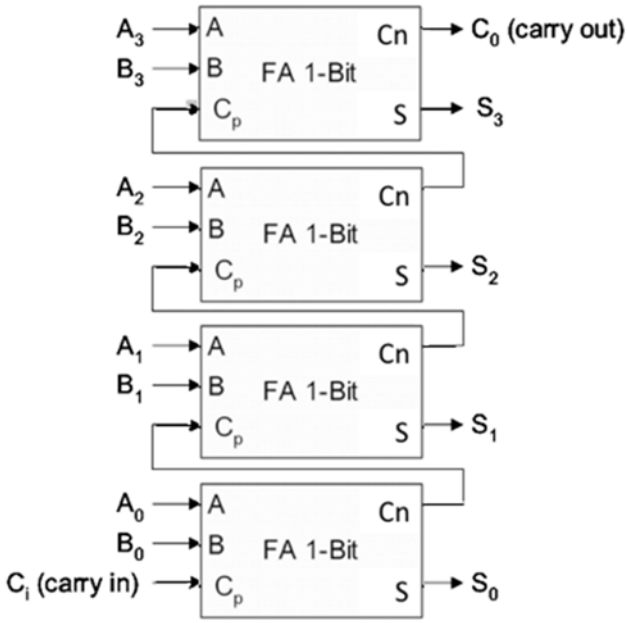


Gambar 81.
Rangkaian *full-adder* 1-bit menggunakan simulator *breadboard*

Melalui saklar logika, berikan kombinasi nilai logika yang mungkin pada input rangkaian dan amati outputnya serta catat hasilnya ke dalam tabel pengamatan. Jika percobaan untuk kedua rangkaian tersebut (*half-adder* dan *full*-adder) dilakukan dengan benar, maka akan diperoleh tabel kebenaran seperti pada Tabel 17 di atas.

3. Merancang dan Menentukan Watak *Parallel Full Adder* 4-bit

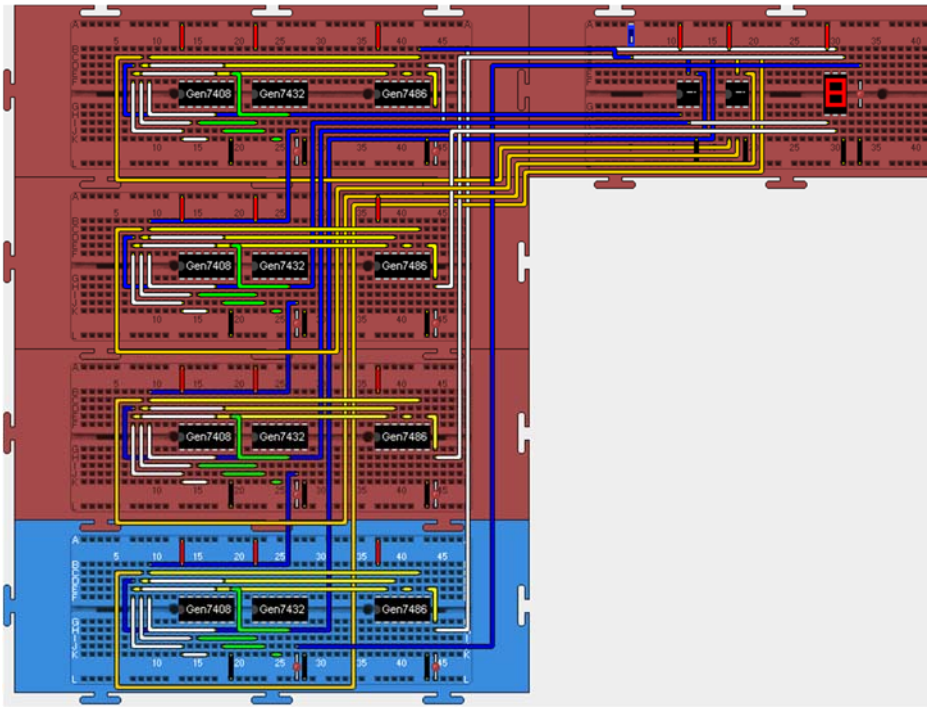
Untuk melakukan percobaan ini, anda dapat menggunakan rangkaian *parallel full adder* 4-bit yang terdiri atas 4 buah rangkaian *full adder* 1-bit seperti pada gambar berikut.



Gambar 82. *Full-adder* paralel 4-bit

Untuk menyusun rangkaian tersebut, tempatkan terlebih dahulu 4 buah rangkaian *full-adder* 1-bit (Gambar 81), sebuah saklar tunggal, dua buah *keypad*, sebuah peraga heksadesimal 7-segmen, dan sebuah LED di atas *breadboard*. Pasang catu daya pada semua piranti yang digunakan, pasang saklar logika pada *carry-in* (Ci) dan indikator LED pada output

carry (Cn), pasang *keypad* heksadesimal pada input *augend* (A) dan *addend* (B) serta peraga heksadesimal pada output *sum.* Hubungkan semua kaki IC Cn ke Cp sesuai rangkaian pada Gambar 82. Salah satu bentuk susunan rangkaian *full-adder* paralel 4-bit menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.

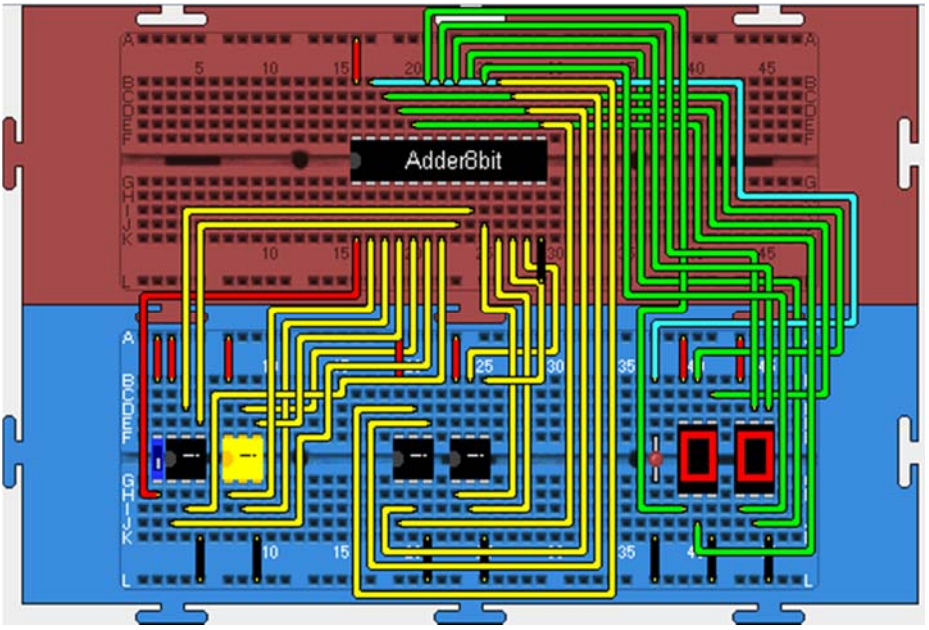


Gambar 83.
Full-adder paralel 4-bit menggunakan simulator *breadboard*

Melalui *keypad* berikan input angka pada *augend* (A) dan *addend* (B), amati dan catat output *sum* dan *carry* ke dalam tabel pengamatan. Lakukan langkah di atas untuk berbagai input *augend* (A) dan *addend* (B) dan akhiri praktik dengan memberikan nilai A=F, nilai B=1, serta Ci=0. Apakah rangkaian *full-adder* telah dapat berfungsi dengan benar? Jika percobaan dilakukan dengan benar, akan diperoleh keadaan bahwa output *sum* akan memberikan *display* hasil penjumlahan data yang

dimasukkan melalui *keypad* A dan *keypad* B, dan untuk data A=F, B=1, Ci=0 akan memberikan *sum*=0 dan *carry*=1.

Ulangi percobaan untuk rangkaian *full-adder* paralel menggunakan IC *full-adder* 8-bit, yang dapat diambil dengan cara klik **Insert>Chip> double click CPU>double click jx_york_ac_uk>double click j1> Adder8bit.** Komponen *keypad* diambil melalui cara klik **Insert>Chip> double click Components>HexKeyPad** dan untuk peraga 7-segmen diambil dengan cara **Insert>Chip>double click Components> HexDisplay.** Salah satu bentuk susunan rangkaian *full-adder* paralel 8-bit ditunjukkan gambar berikut ini.



Gambar 84.
Full-adder paralel 8-bit menggunakan simulator *breadboard*

D. Mulitiplekser dan Demultiplekser

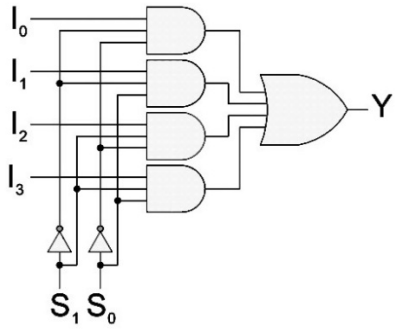
Multiplekser (sering disebut dengan MUX saja yang merupakan singkatan dari istilah aslinya yakni *multiplexer*) dan demultiplekser (biasanya disebut dengan DEMUX saja yakni singkatan dari istilah aslinya

demultiplexer) termasuk rangkaian logika kombinasi. MUX merupakan suatu rangkaian yang berfungsi memilih salah satu dari beberapa sinyal masukan untuk disalurkan melalui keluarannya dengan bantuan sinyal kendali. Pada dasarnya MUX bertugas seperti saklar pemilih sehingga disebut juga sebagai pemilih data (*data selector*). Sedangkan DEMUX adalah kebalikan dari MUX, rangkaian ini berfungsi menyalurkan suatu sinyal masukan ke salah satu dari beberapa saluran keluarannya, sehingga disebut pula sebagai penyalur data (*data distributor*).

Melalui percobaan ini anda dapat menyelidiki watak MUX dan DEMUX sederhana yang dibangun dengan menggunakan gerbang logika dasar.

1. **Menyelidiki watak rangkaian MUX 1-bit 4 ke 1**

   Rangkaian MUX 4 ke 1 untuk percobaan ini ditunjukkan pada Gambar 85 dan tabel kebenarannya pada Tabel 18.

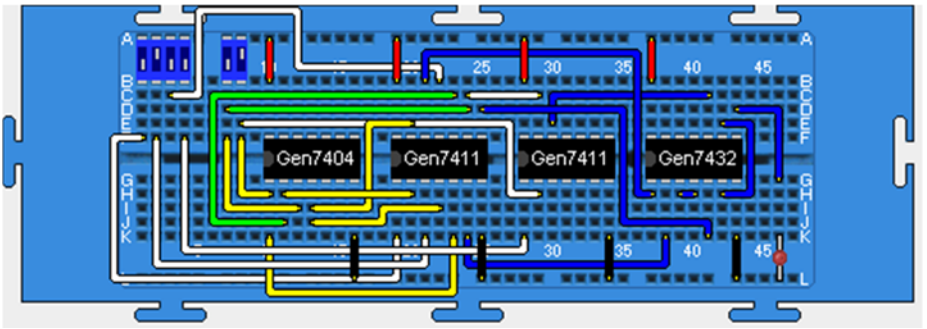


Tabel 18. Tabel kebenaran MUX 4 ke 1

PEMILIH		OUTPUT
S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

Gambar 85. Rangkaian MUX 4 ke 1

Untuk menyusun rangkaian tersebut, tempatkan terlebih dahulu di atas papan *breadboard* sebuah IC 7404 (NOT), 7432 (OR), dua buah IC 7411 (AND-3 input), 3 buah saklar *double*, dan sebuah LED. Selanjutnya, pasang catu daya pada semua IC yang digunakan, pasang saklar logika pada input rangkaian dan indikator LED pada output rangkaian. Salah satu bentuk susunan rangkaian MUX 4 ke 1 menggunakan simulator *breadboard* ditunjukkan gambar berikut ini.



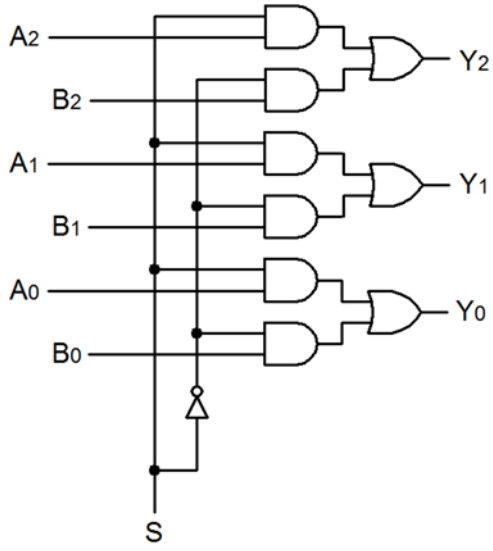
Gambar 86. MUX 4 ke 1 menggunakan simulator *breadboard*

Berikan nilai input logika pada saklar kendali S_1=0, S_0=0 (kedua saklar bernilai 0 atau dapat ditulis juga dengan S_1S_0=00). Dengan pengaturan sinyal kendali tersebut, rangkaian MUX akan memilih I_0 dan menyalurkan sinyal yang terpasang pada input tersebut ke output Y. Untuk membuktikan hal ini, coba berikan nilai logika pada saklar input I_0 bervariasi 0 dan 1, perhatikan outputnya! Sama bukan dengan keadaan I_0? Jika anda melakukan percobaan dengan benar, maka keadaan output MUX akan mengikuti keadaan input I_0, yang menunjukkan bahwa MUX telah menyalurkan data dari I_0 ke outputnya. Sebaliknya, jika anda memberikan variasi keadaan 0 dan 1 pada input-input lainnya yakni I_1, I_2, dan I_3, hal itu tidak akan memberikan efek apapun pada outputnya.

Selanjutnya, lakukan percobaan untuk sinyal kendali S_1S_0=01, S_1S_0=10 dan S_1S_0=11! Jika percobaan dilakukan dengan benar, maka akan dihasilkan tabel kebenaran MUX 4 ke 1 sesuai dengan Tabel 18 di atas.

2. Menyelidiki Watak Rangkaian MUX 3-bit 2 ke 1

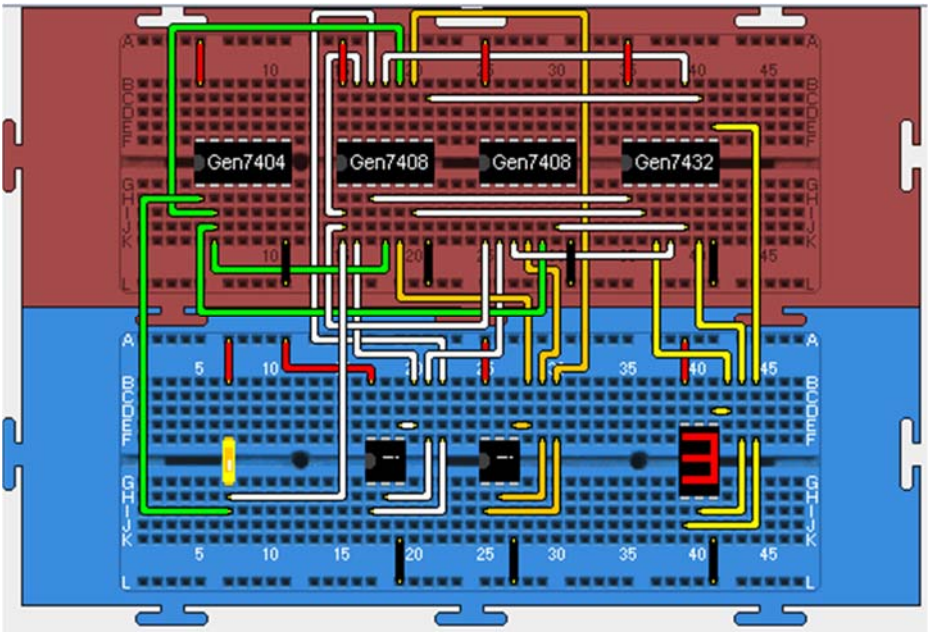
Rangkaian MUX 3-bit 2 ke 1 untuk percobaan ini dapat menggunakan susunan seperti pada gambar berikut ini.



Gambar 87. Rangkaian MUX 3 bit 2 ke 1

Coba susun rangkaian tersebut, dengan terlebih dahulu menempatkan sebuah IC 7404 (NOT), 7432 (OR), dua buah IC 7408 (AND), sebuah saklar tunggal, dua buah *keypad* dan sebuah peraga 7-segmen di atas *breadboard* ganda. Selanjutnya, pasang catu daya pada semua IC yang digunakan, pasang saklar logika pada input pemilih (S), pasang *keypad* pada input rangkaian dan peraga 7-segmen LED pada output rangkaian.

Salah satu bentuk susunan rangkaian MUX 3-bit 2 ke 1 menggunakan simulator *breadboard* ditunjukkan gambar berikut ini.

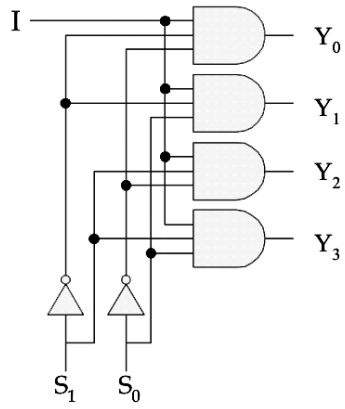


Gambar 88.
Rangkaian MUX 3 bit 2 ke 1 menggunakan simulator *breadboard*

Pada MUX ini terdapat 2 input yakni A dan B masing-masing 3-bit sehingga nilai maksimum dalam desimal yang dapat diinputkan adalah 7. Coba, berikan nilai input misalnya A=3 melalui *keypad* A (kanan) dan B=7 melalui *keypad* B (kiri)! Berikan S=0 untuk memilih input A dan amati output rangkaian serta catat hasilnya ke dalam tabel pengamatan. Jika percobaan dilakukan dengan benar, anda akan memperoleh keadaan untuk sinyal kendali S=0 data pada *keypad* A akan disalurkan ke output sehingga akan tampil angka 3 pada outputnya. Sedangkan jika anda berikan S=1 (saklar S dalam keadaan ON), data pada *keypad* B akan disalurkan ke output rangkaian sehingga akan tampil angka 7 pada peraga output.

3. Menyelidiki Watak Rangkaian DEMUX 1 ke 4

Rangkaian DEMUX 1 ke 4 menggunakan gerbang logika dasar dan wataknya ditunjukkan melalui gambar dan tabel berikut ini.

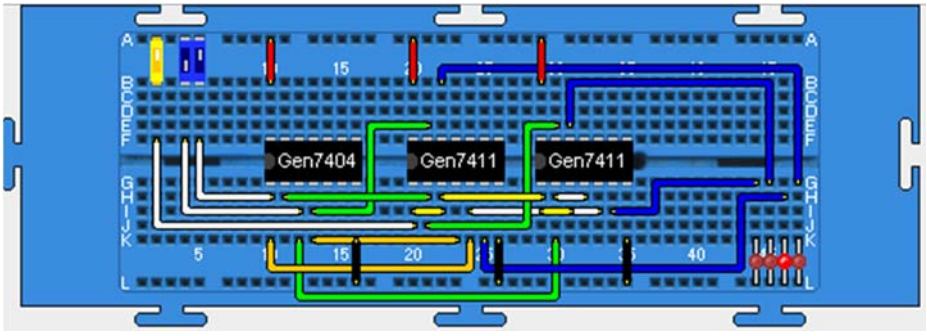


Tabel 19. Tabel Kebenaran DEMUX 1 ke 4

PEMILIH		OUTPUT			
S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	I	0	0	0
0	1	0	I	0	0
1	0	0	0	I	0
1	1	0	0	0	I

Gambar 89. DEMUX 1 ke 4

Untuk menghasilkan rangkaian DEMUX 4 ke 1, siapkan IC 7404 (NOT), 2 buah 7411 (AND-3 input), 1 buah saklar *triple* dan 4 buah LED. Susun rangkaian di atas *breadboard* sesuai Gambar 89. Salah satu bentuk susunan rangkaian DEMUX 1 ke 4 menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.



Gambar 90. DEMUX 1 ke 4 menggunakan simulator *breadboard*

Agar anda lebih memahami watak DEMUX, coba berikan pada input pengendali nilai logika S_1S_0=01, kemudian berikan nilai logika tinggi pada inputnya dan amati output rangkaian! Selanjutnya ubah-ubahlah keadaan input dari 1 ke 0 ke 1 lagi dan perhatikan outputnya! Apakah keadaan output Y_1 mengikuti keadaan inputnya? Jika percobaan dilakukan dengan benar, maka keadaan output Y_1 terlihat mengikuti keadaan input I, hal itu menunjukkan bahwa dengan memberikan sinyal kendali bernilai 1 yakni S_1S_0=01, DEMUX akan menyalurkan sinyal pada inputnya ke output Y_1. Ulangi eksperimen dengan sinyal kendali S_1S_0=00, S_1S_0=10 dan S_1S_0=11! Jika percobaan berlangsung dengan benar, maka anda akan memperoleh tabel kebenaran seperti pada Tabel 19 di atas.

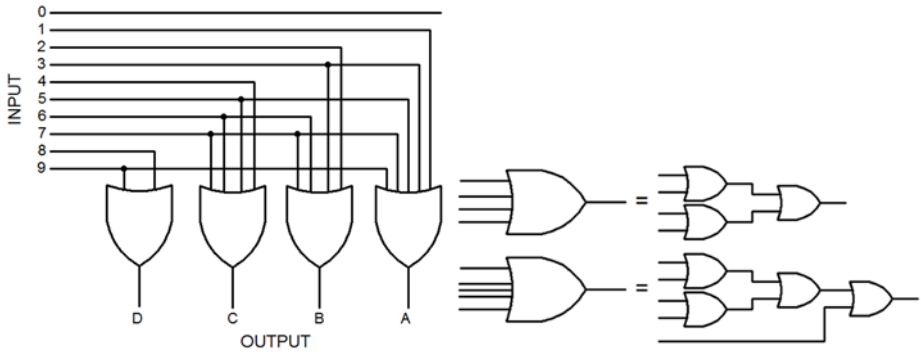
E. Enkoder dan Dekoder

Dalam suatu sistem digital, informasi atau data disajikan dalam bentuk kode biner. Untuk mengkode suatu informasi, misalnya dalam bentuk angka desimal menjadi kode biner dilakukan dengan piranti enkoder, sedangkan untuk mengenal kembali arti kode tersebut, digunakan rangkaian logika yang disebut dekoder.

Salah satu jenis sistem kode yang digunakan dalam sistem digital adalah sistem desimal dikode biner (*binary-coded decimal*) atau disingkat BCD. Dalam sistem kode ini, masing-masing digit angka desimal diganti dengan suatu kombinasi 4-bit biner. Salah satu dari jenis kode BCD adalah desimal dikode biner asli atau BCD 8421. Dalam sistem BCD 8421 ini angka 3 desimal akan dikode sebagai 0011, sedangkan angka 264 desimal akan dikode sebagai 0010 0110 0100. Untuk mengkode angka desimal tersebut menjadi kode BCD 8421 digunakan enkoder desimal ke BCD, sedangkan untuk menemukan kembali atau menafsirkan kode-kode tersebut dalam bentuk desimal, diperlukan dekoder BCD ke desimal. Percobaan dengan topik ini ditujukan untuk menyelidiki watak enkoder desimal ke BCD, dan dekoder BCD ke desimal.

1. Menyelidiki Watak Enkoder Desimal ke BCD

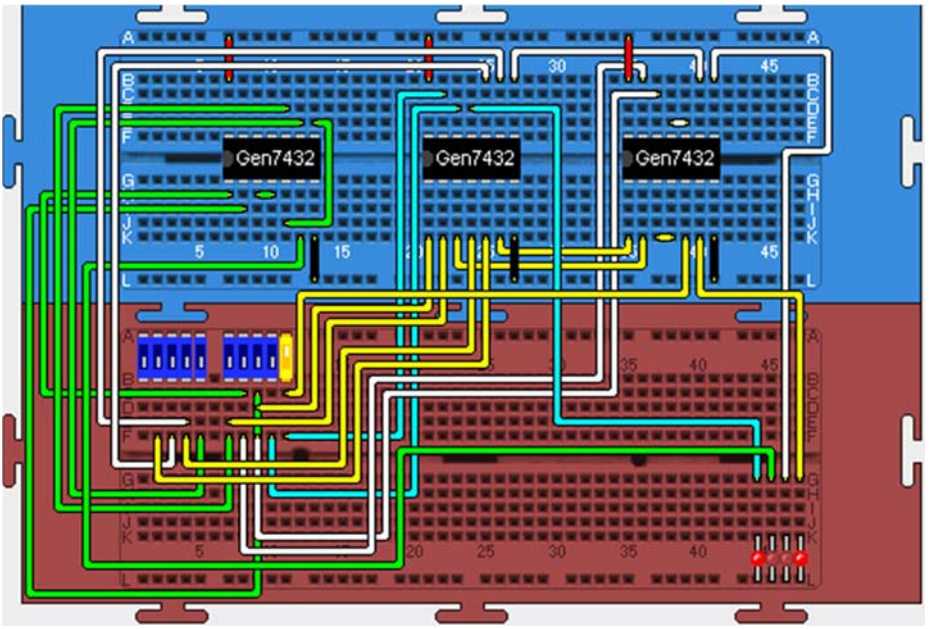
Rangkaian enkoder desimal ke BCD untuk percobaan ini dapat menggunakan susunan seperti pada gambar berikut ini.



(a) (b)

Gambar 91.
Rangkaian: (a) Enkoder desimal ke BCD; (b) konversi gerbang OR

Untuk menyusun rangkaian seperti pada Gambar 91a, perlu disediakan terlebih dahulu tiga sebuah IC 7432 (OR), 2 buah saklar quad, 1 buah saklar ganda dan empat buah LED di atas *breadboard*. Agar penyusunan rangkaian lebih mudah, gunakan *breadboard* ganda! Selanjutnya, pasang catu daya pada semua IC yang digunakan, hubungkan kesepuluh saklar logika ke input rangkaian dan keempat indikator LED ke output rangkaian. Salah satu bentuk susunan rangkaian enkoder desimal ke BCD menggunakan simulator *breadboard* ditunjukkan gambar berikut ini.



Gambar 92.
Rangkaian enkoder desimal ke BCD
menggunakan simulator *breadboard*

Kesepuluh saklar input mewakili bilangan desimal 0 (saklar paling kiri) sampai dengan bilangan 9 (saklar paling kanan). Coba Berikan nilai 1 (saklar ON) pada input paling kanan (mewakili angka 9 desimal), saklar lainnya OFF dan amati outputnya! Jika percobaan anda benar, akan diperoleh kode BCD pada output enkoder DCBA= 1001. Dengan mengulangi percobaan untuk semua saklar input yang tersedia, maka akan diperoleh tabel kebenaran enkoder desimal ke BCD seperti berikut ini.

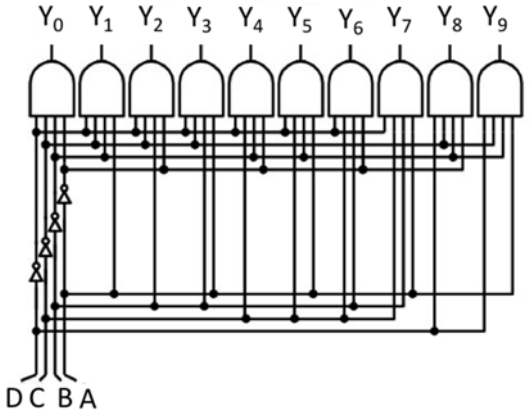
Tabel 20. Tabel Kebenaran enkoder desimal ke BCD

INPUT										OUTPUT			
0	1	2	3	4	5	6	7	8	9	D	C	B	A
1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	0	0	0	1	0	0
0	0	0	0	0	1	0	0	0	0	0	1	0	1
0	0	0	0	0	0	1	0	0	0	0	1	1	0
0	0	0	0	0	0	0	1	0	0	0	1	1	1
0	0	0	0	0	0	0	0	1	0	1	0	0	0
0	0	0	0	0	0	0	0	0	1	1	0	0	1

Berdasarkan Tabel 20, terlihat bahwa rangkaian enkoder desimal ke BCD telah berhasil mengubah (mengkode) bilangan desimal menjadi kode biner jenis BCD.

2. Menyelidiki Watak Dekoder BCD ke Desimal

Kebalikan dari enkoder, dekoder BCD ke desimal akan mengubah kembali kode BCD menjadi data asli yakni bilangan desimal. Rangkaiannya ditunjukkan pada gambar berikut ini.



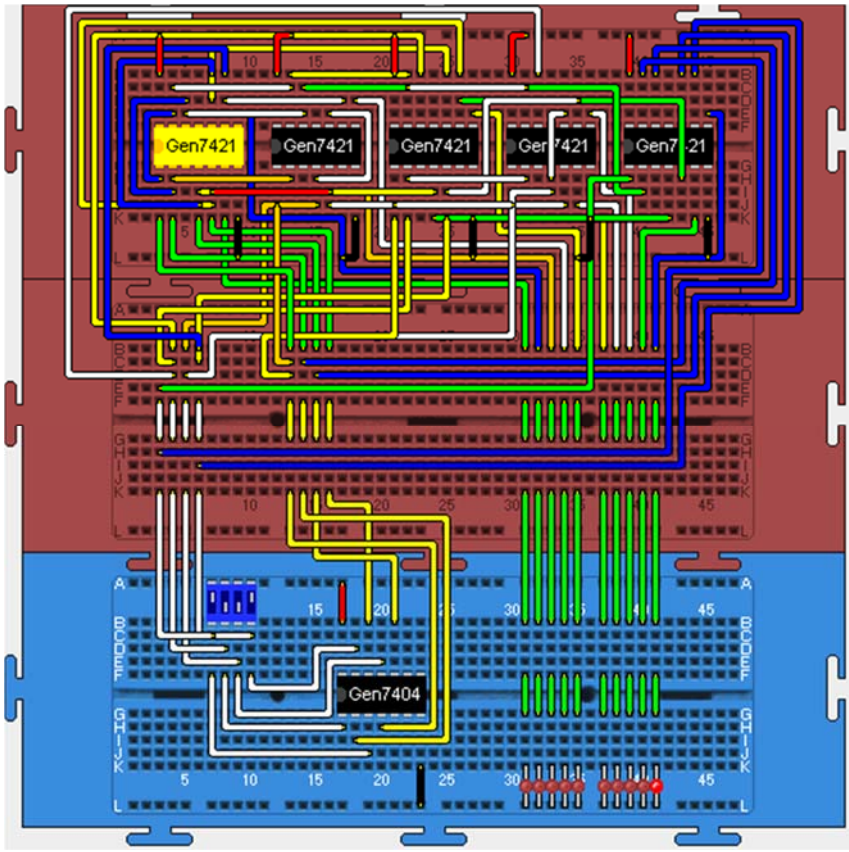
Gambar 93. Dekoder BCD ke desimal

Watak rangkaian dekoder pada Gambar 93 di atas ditunjukkan tabel kebenaran berikut ini.

Tabel 21. Tabel Kebenaran dekoder BCD ke desimal

INPUT				OUTPUT									
D	C	B	A	0	1	2	3	4	5	6	7	8	9
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1

Salah satu bentuk susunan rangkaian dekoder BCD ke desimal di atas *breadboard* ditunjukkan pada gambar berikut ini.



Gambar 94. *Decoder* BCD ke desimal menggunakan simulator *breadboard*

Untuk menghasilkan susunan tersebut, perlu disediakan 3 buah *breadboard,* 5 buah IC 7421 (AND 4-input), 1 buah IC 7404 (NOT), 1 buah saklar *quad*, dan 10 buah LED. Saklar *quad* mewakili input kode BCD dan 10 buah LED mewakili output desimal dari bilangan 0 (LED paling kiri) sampai dengan bilangan 9 (LED paling kanan). Coba anda berikan kode biner 0001 (DCBA=0001) pada input rangkaian, kemudian amati outputnya! Jika anda benar dalam melakukan percobaan, maka hasilnya LED nomor 2 dari kiri (mewakili bilangan 1) akan menyala, hal ini menunjukkan rangkaian dekoder telah melakukan penafsiran kode 0001 (kode BCD) menjadi angka 1 desimal pada outputnya. Ulangi pecobaan untuk kode BCD lainnya! Langkah-

langkah anda yang benar dalam percobaan ulang ini akan menghasilkan tabel kebenaran sesuai dengan Tabel 21 di atas.

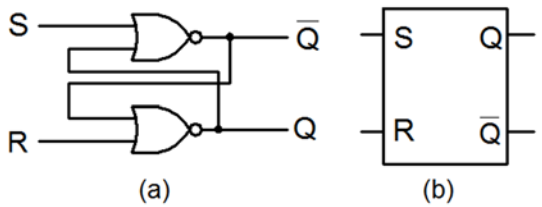
F. Flip-flop

Flip-flop merupakan elemen rangkaian logika sekuensial, yakni rangkaian yang keluarannya tergantung pada keluaran sebelumnya. Karena sifat-sifatnya, flip-flop disebut juga pengingat (memory) 1-bit. Sekumpulan flip-flop dapat membentuk sel pengingat beberapa bit. Flip-flop memegang peranan yang sangat penting di dalam perancangan rangkaian logika sekuensial. Dari flip-flop dapat dibangun rangkaian register yang merupakan pengingat dengan banyak bit, dan rangkaian pencacah (counter). Rangkaian-rangkaian tersebut merupakan dasar bagi perancangan perangkat keras komputer.

Percobaan dengan topik ini ditujukan untuk menyelidiki watak berbagai macam flip-flop yakni jenis flip-flop Set-Reset (FFSR), flip-flop JK (FFJK), flip-flop D (FFD), dan flip-flop T (FFT). Selain itu, melalui percobaan ini dapat pula diselidiki watak rangkaian *toggle* sebagai elemen pencacah, yang dibangun dari FFJK, FFD dan FFT.

1. Menentukan Watak FFSR (Set-Reset)

Rangkaian FFSR menggunakan gerbang NOR dan tabel kebenarannya ditunjukkan pada gambar serta tabel berikut ini.



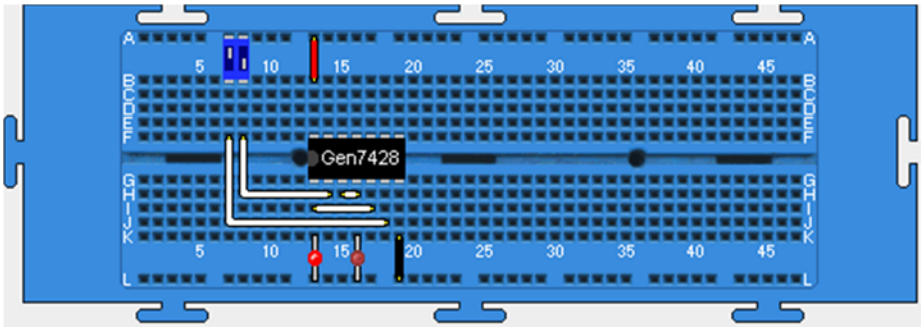
Gambar 95.
FFSR menggunakan gerbang NOR: (a) rangkaian; (b) simbol

Tabel 22. Tabel kebenaran FFSR menggunakan gerbang NOR

INPUT		OUTPUT		
S	R	Q_n	$\overline{Qn}$	KEADAAN
1	0	1	0	Set (Q_n=1)
0	1	0	1	Reset (Q_n=0)
0	0	Q_{n-1}	$\overline{Q_{n-1}}$	Tetap (Q_n=Q_{n-1})
1	1	?	?	Terlarang (Q_n=?)

Qn: output sekarang, Qn-1: output sebelumnya

Penyusunan rangkaian tersebut di atas *breadboard* dilakukan dengan menyediakan terlebih dahulu sebuah IC 7428 (NOR), sebuah saklar ganda dan 2 buah LED pada papan *breadboard.* Selanjutnya, pasang catu daya pada IC yang digunakan, pasang saklar logika pada input S dan R dan indikator LED pada output Q dan $\overline{Q}$. Salah satu bentuk susunan rangkaian FFSR menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.

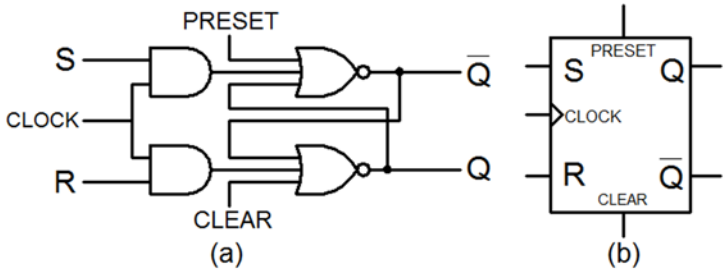


Gambar 96. Susunan FFSR menggunakan simulator *breadboard*

Coba berikan berbagai kombinasi nilai logika yang mungkin pada input S dan R, dan amati output rangkaian Q dan $\overline{Q}$ serta catat hasilnya ke dalam tabel pengamatan. Jika percobaan berlangsung dengan benar maka akan diperoleh tabel keadaan seperti Tabel 22 di atas.

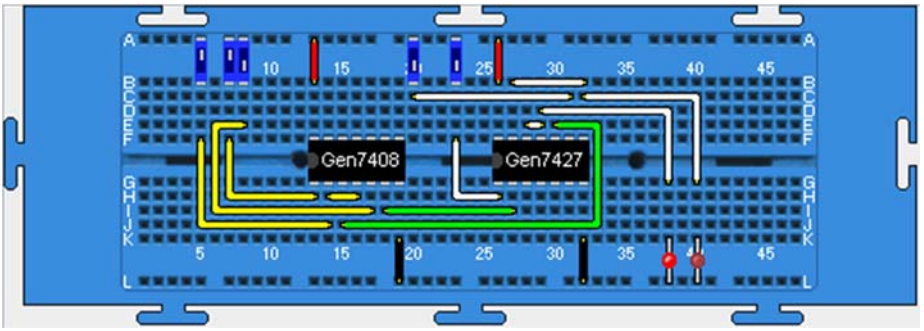
2. Menentukan Watak FFSR *Clocked*

FFSR yang lebih kompleks dilengkapi dengan fitur *clock, preset* dan *clear* seperti ditunjukkan pada gambar berikut ini.



Gambar 97. FFSR *clocked*

Penyusunan rangkaian tersebut di atas *breadboard* dapat dilakukan dengan menyediakan terlebih dahulu sebuah IC 7427 (NOR-3 input), sebuah IC 7408 (AND), 3 buah saklar tunggal, sebuah saklar ganda, dan 2 buah LED pada papan *breadboard.* Selanjutnya, pasang catu daya pada semua IC yang digunakan, pasang saklar logika pada input S, R, *Preset*, *Clear* dan *clock* serta indikator LED pada output Q dan $\overline{Q}$. Salah satu bentuk susunan rangkaian FFSR *clocked* yang dilengkapi dengan *Preset* dan *Clear* menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.

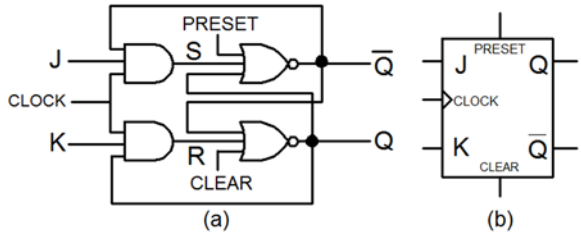


Gambar 98.
Susunan FFSR *clocked* menggunakan simulator *breadboard*

Berikan berbagai kombinasi nilai logika yang mungkin pada input S dan R, dan amati output rangkaian Q dan $\overline{Q}$ setiap saat sinyal *clock* diberikan serta catat hasilnya ke dalam tabel pengamatan. Pemberian sinyal *Clock, Preset*, *Clear* dilakukan dengan cara: saklar di-ON-kan, sesaat kemudian di-OFF-kan kembali. *Preset* untuk memberikan keadaan awal Qn-1=1 dan *Clear* untuk Qn-1=0. Eksperimen yang benar akan menghasilkan tabel kebenaran sesuai Tabel 22 di atas.

3. Menentukan Watak FFJK

FFJK merupakan perbaikan disain dari FFSR agar kinerjanya meningkat. Rangkaian dan tabel kebenarannya dapat dilihat pada uraian berikut ini.



Gambar 99. Flip-flop JK: (a) rangkaian; (b) simbol

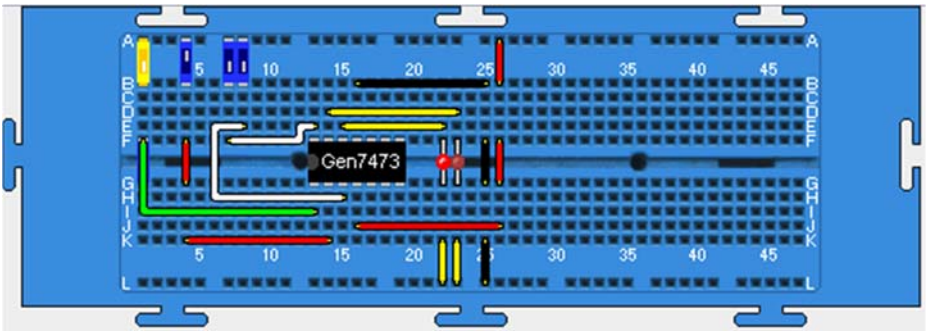
Tabel 23. Tabel kebenaran FFJK

INPUT		OUTPUT		
J	K	Q_n	$\overline{Qn}$	KEADAAN
1	0	1	0	Set (Q_n=1)
0	1	0	1	Reset (Q_n=0)
0	0	Q_{n-1}	$\overline{Q_{n-1}}$	Tetap (Q_n=Q_{n-1})
1	1	$\overline{Q_{n-1}}$	Q_{n-1}	Komplemen $Qn = \overline{Qn-1}$

Qn: output sekarang, Qn-1: output sebelumnya

Susun rangkaian FFJK dengan menempatkan terlebih dahulu IC 7473 (FFJK), saklar ganda, 2 buah saklar tunggal dan 2 buah LED pada papan *breadboard.* Pasang catu daya pada IC, hubungkan saklar logika

ke input J, K, Clear dan *clock* serta indikator LED ke output Q dan $\overline{Q}$. Salah satu bentuk susunan rangkaian FFJK menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.

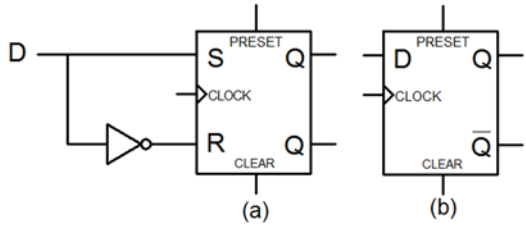


Gambar 100.
Susunan rangkaian FFJK menggunakan simulator *breadboard*

Pastikan terlebih dahulu Clear=1 (saklar ON). Untuk memulai percobaan, anda perlu memberikan keadaan awal reset (Q_{n-1}=0) yang dilakukan dengan cara meng-OFF-kan saklar *Clear* sesaat kemudian meng-ON-kannya kembali. Setiap selesai memberikan sinyal *Clear,* pertahankan agar saklar tetap ON. Berikan nilai J=1 (saklar J di-ON-kan) dan nilai K=0 (saklar K di-OFF-kan), amati outputnya! Hasilnya anda akan melihat output Q=1 dan $\overline{Q}$=0, yang menunjukkan FFJK dalam keadaan *set*. Jika anda memberikan J=K=0 akan terlihat outputnya tetap. Coba teruskan dengan memberi J=0 dan K=1, apa yang terjadi? Anda akan melihat bahwa FFJK dalam keadaan *reset* yakni Q=0 dan $\overline{Q}$ =1. Sekarang apa yang terjadi jika anda berikan J=K=1? Terlihat bahwa FFJK akan memberikan output yang keadaannya bergantian antara Q dan $\overline{Q}$ untuk setiap *clock* yang terjadi.

4. Menentukan watak FFD

FFD dapat dikembangkan dari FFSR seperti ditunjukkan apada gambar berikut ini.



Gambar 101. FFD: (a) rangkaian; (b) simbol

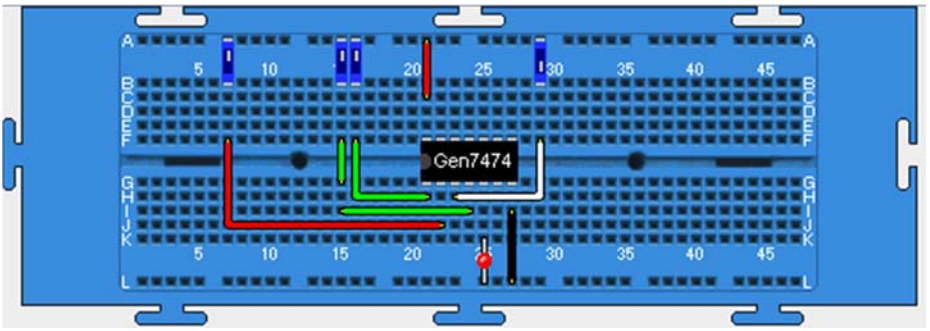
Tabel 24. Tabel kebenaran FFD

INPUT	OUTPUT		
D	Q_n	$\overline{Qn}$	KEADAAN
0	0	1	Reset (Q_n=0)
1	1	0	Set (Q_n=1)

Qn: output sekarang, Qn-1: output sebelumnya

Rangkaian ini dapat direalisasikan dengan menyiapkan terlebih dahulu sebuah IC 7474 (FFD), 2 buah saklar tunggal, 1 buah saklar ganda dan sebuah LED pada papan *breadboard.* Anda dapat meneruskan dengan memasang catu daya pada IC, memasang saklar logika ke input D, Preset, Clear dan *clock* serta indikator LED ke output Q. Salah satu bentuk susunan rangkaian FFD menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.

Untuk memulai percobaan, pastikan terlebih dahulu saklar Preset dan Clear pada posisi ON, keduanya harus bernilai logika 1. Pemberian kondisi awal (Qn-1=0) dilakukan dengan meng-OFF-kan saklar Clear, sesaat kemudian meng-ON-kannya. Kondisi awal Qn-1=1 dapat diatur dengan meng-OFF-kan saklar Preset, sesaat kemudian meng-ON-kannya.

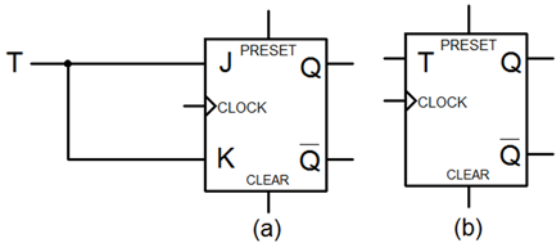


Gambar 102.
Susunan Flip-flop D menggunakan simulator *breadboard*

Berikan nilai input D=1 (saklar D di-ON-kan), amati outputnya! Melalui langkah ini anda akan melihat output Q=1 yang menunjukkan FFD menyimpan data 1 (set). Jika anda memberikan input D=0 dengan meng-OFF-kan saklar D, akan diperoleh keadaan Q=0 yang menunjukkan bahwa FFD menyimpan data 0 (reset).

5. Menentukan watak FFT

Rangkaian FFT dapat direalisasikan menggunakan FFJK dengan menghubungkan input J dan K menjadi input T. Uraian berikut ini menunjukkan gambar dan tabel kebenaran FFT.



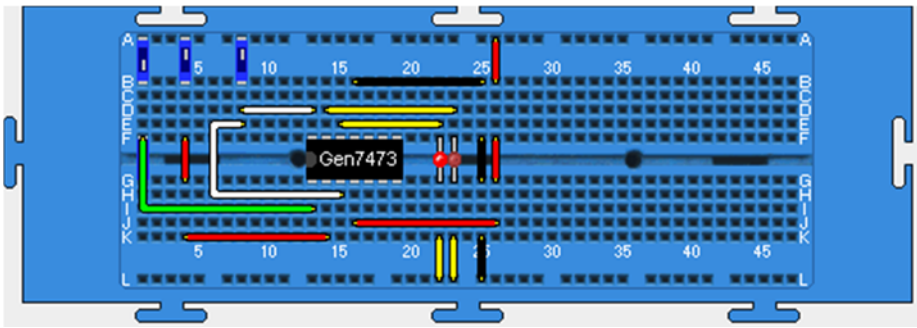
Gambar 103. Flip-flop T: (a) rangkaian; (b) simbol

Tabel 25. Tabel kebenaran FFT

INPUT	OUTPUT		
T	Q_n	$\overline{Qn}$	KEADAAN
0	Q_{n-1}	$\overline{Q_{n-1}}$	Tetap(Q_n=Q_{n-1})
1	$\overline{Qn-1}$	Q_{n-1}	Komplemen $Qn = \overline{Qn-1}$

Qn: output sekarang, Qn-1: output sebelumnya

Rangkaian ini dapat disusun dengan terlebih dahulu menyediakan sebuah IC 7473 (FFJK), 3 buah saklar tunggal, dan dua buah LED di atas papan *breadboard*. Pasang catu daya pada IC, hubungkan saklar logika ke input T, *Clear* dan *Clock* serta indikator LED ke output Q dan $\overline{Q}$. Salah satu bentuk susunan rangkaian FFT menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.

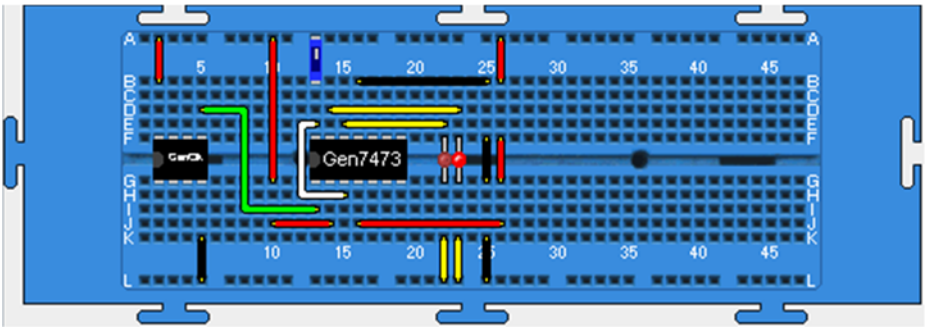


Gambar 104. Susunan FFT menggunakan simulator *breadboard*

Jalankan simulasi dan ON-kan saklar *Clear* untuk memastikan input *Clear* selalu bernilai 1. Coba berikan nilai output awal Qn-1=0 dengan cara saklar *Clear* di-OFF-kan, sesaat kemudian di-ON-kan kembali. Berikan 0 pada input T dengan meng-OFF-kan saklar input T, diikuti pemberian *Clock*. Amati output rangkaian Q dan $\overline{Q}$. Anda akan melihat bahwa untuk T=0, setiap *Clock* yahng diberikan akan memberikan output yang tetap, sama dengan output sebelumnya (Qn=Qn-1). Sekarang berikan T=1 dan sinyal *Clock*, serta ulangi langkah tersebut! Apa yang terjadi? Anda akan melihat bahwa setiap saat sinyal *clock*

diberikan untuk T=1, output rangkaian berganti dari 0 ke 1 dan seterusnya.

Selanjutnya, coba ganti saklar *clock* dengan generator *clock,* dan hubungkan *Clear* ke +Vcc, sehingga rangkaian menjadi seperti berikut ini.

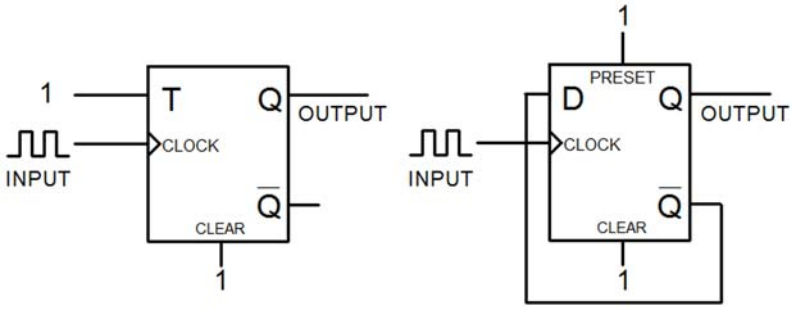


Gambar 105. Flip-flop T dengan input generator *clock*

Jalankan simulasi, berikan input J=K=1 dan atur **kecepatan simulasi (Sim Speed) hingga level 8,5**, amati output rangkaian! Selalu berubah bukan, untuk setiap *clock* yang diberikan?

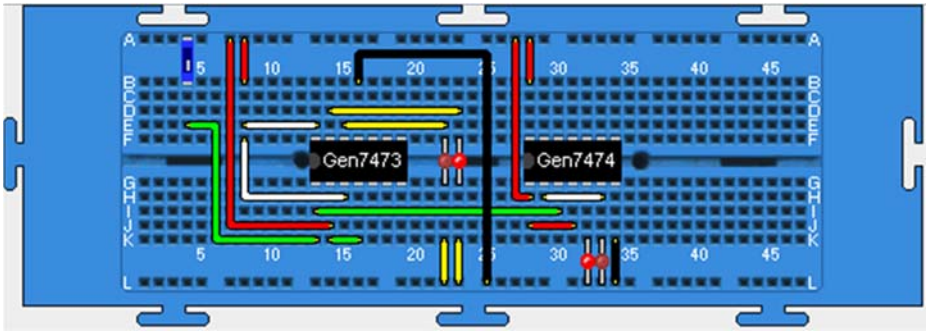
6. Menentukan watak rangkaian *toggle*

Rangkaian *toggle* dapat diimplementasikan dengan menggunakan FFT atau FFD seperti ditunjukkan pada gambar berikut ini.



Gambar 106. Rangkaian *Toggle* dengan FFT dan FFD

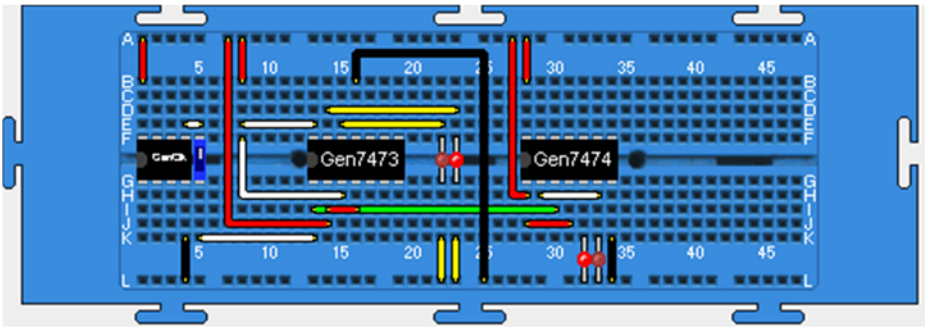
Untuk menyusun kedua rangkaian tersebut, sediakan terlebih dahulu IC 7473 (FFJK), IC 7474 (FFD), saklar tunggal, generator *clock,* dan empat buah LED di atas papan *breadboard.* Pasang catu daya pada semua IC yang digunakan, hubungkan input J, K, ke logika 1 (catu daya +), saklar tunggal ke input *clock* FFJK dan FFD, serta indikator LED ke output dari FFJK maupun FFD. Untuk FFD, hubungkan input D ke output $\overline{Q}$. Gambar berikut ini menunjukkan bentuk susunan rangkaian *toggle* menggunakan *breadboard.*



Gambar 107.
Toggle dengan FFT dan FFD menggunakan simulator *breadboard*

Berikan *clock* beberapa kali dengan cara meng-ON-kan saklar, sesaat kemudian meng-OFF-kannya kembali, dan amati output rangkaian Q dan $\overline{Q}$ setiap saat sinyal *clock* diberikan! Anda akan melihat bahwa setiap *clock* yang diberikan, output rangkaian akan berubah dari keadaan sebelumnya (*toggle*).

Sekarang coba ganti saklar *clock* dengan generator *clock,* sehingga rangkaian menjadi seperti berikut ini.



Gambar 108. *Toggle* dengan FFT dan FFD dengan input generator *clock*

Jalankan simulasi dan atur **kecepatan simulasi (Sim Speed) hingga level 8,5**, amati output rangkaian! Anda akan melihat kedua rangkaian *toggle* akan memberikan keadaan output yang bergantian 1 dan 0 untuk setiap *clock* yang diberikan.

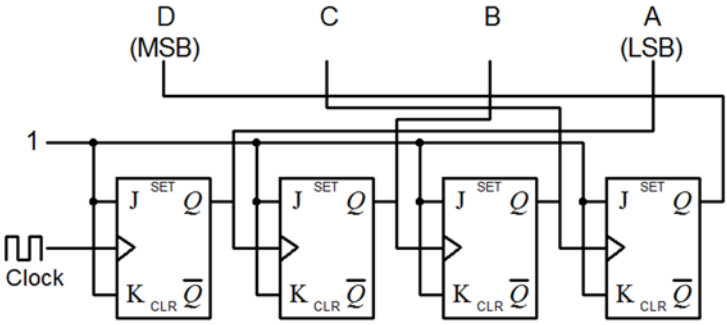
G. Pencacah

Pencacah (*counter*) merupakan salah satu rangkaian logika sekuensial, dan dibangun dengan menggunakan beberapa flip-flop. Sebuah flip-flop mempunyai 2 keadaan yakni keadaan 0 (RESET) dan keadaan 1 (SET), sehingga sederetan n buah flip-flop mempunyai 2^n keadaan yang berbeda. Dalam penggunaannya sebagai pencacah pulsa, setiap satu keadaan dari 2^n keadaan, digunakan untuk menyatakan jumlah pulsa yang masuk pencacah. Dengan demikian hubungan antara flip-flop yang satu dengan yang lain sedemikian rupa sehingga keadaannya akan berubah secara berurutan setiap kali ada pulsa masuk.

Pencacah terdiri atas *asynchronous counter* (pencacah tak sinkron) dan *synchronous counter* (pencacah sinkron). Percobaan dengan topik ini ditujukan untuk menyelidiki watak pencacah yang dibangun dari FFJK, FFD, maupun yang tersedia dalam bentuk IC.

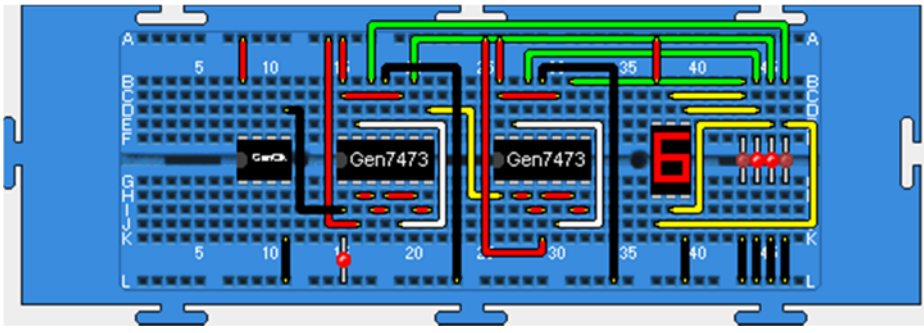
1. Menentukan Watak Rangkaian Pencacah Asinkron Modulo-16

Pencacah asinkron modulo-16 dapat direalisasikan dengan FFJK seperti gambar berikut ini.



Gambar 109. Pencacah modulo-16 menggunakan FFJK

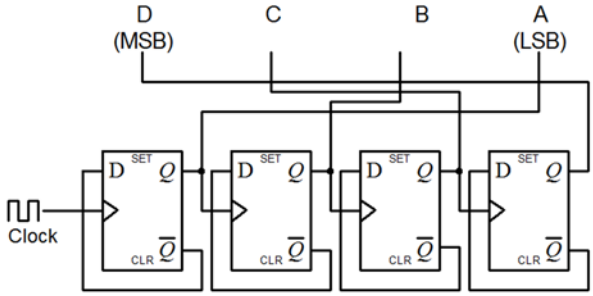
Untuk menyusun rangkaian pada Gambar 109, siapkan terlebih dahulu dua buah IC 7473 (FFJK), sebuah generator *clock*, sebuah peraga 7-segmen, dan lima buah LED pada papan *breadboard.* Selanjutnya, hubungkan catu daya ke semua IC yang digunakan, generator *clock* dan LED ke input *clock*, serta pasang peraga heksadesimal dan 4 buah LED pada output pencacah. Salah satu bentuk susunan rangkaian pencacah modulo-16 dengan FFJK menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.



Gambar 110. Pencacah modulo-16 menggunakan FFJK

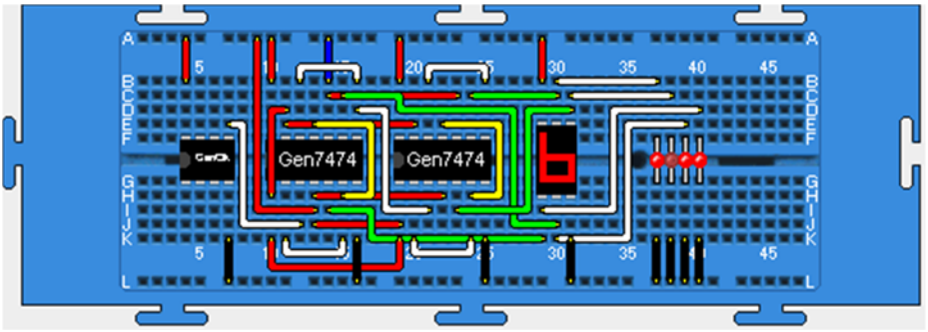
Jalankan simulasi dan atur **kecepatan simulasi (Sim Speed) hingga level 8,5**! Amati input pencacah melalui LED dan output pencacah melalui peraga heksadesimal! Anda akan melihat output rangkaian menampilkan angka 0 sampai dengan F heksadesimal setiap *clock* yang diberikan. Hal itu berarti rangkaian telah melakukan pencacahan pulsa *clock* yang masuk sebanyak 16 pulsa (0 sampai dengan F) dan setelah 16 kali pencacahan, rangkaian akan reset kembali ke 0 dan seterusnya.

Anda juga dapat menyusun rangkaian pencacah asinkron modulo-16 dengan menggunakan FFD. Rangkaiannya ditunjukkan pada gambar berikut ini.



Gambar 111. Pencacah Modulo-16 dengan FFD

Salah satu bentuk susunan rangkaian pencacah modulo-16 dengan FFD menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini. Dalam hal ini IC yang digunakan adalah 7474 yang menyediakan FFD.

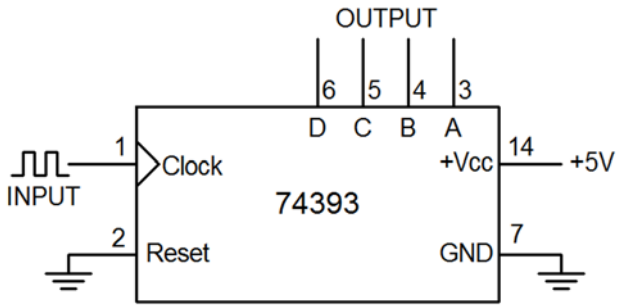


Gambar 112. Pencacah modulo-16 dari FDD

Coba lakukan eksperimen yang sama dengan eksperiman seperti pada pencacah modulo-16 menggunakan FFJK di atas. Hasilnya sama bukan?

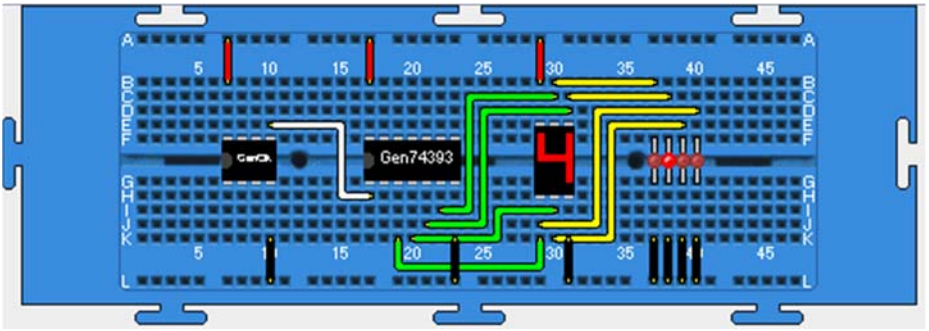
2. Menyelidiki Watak Pencacah IC 74393

Salah satu IC yang menyediakan fungsi pencacah adalah seri 74393. Dengan menggunakan IC ini anda dapat menyusun pencacah modulo-16 dan pencacah dengan modulo-modulo di bawahnya seperti modulo-12 dan modulo-6. Berikut ini adalah gambar rangkaian pencacah modulo-16 menggunakan IC 74393.



Gambar 113. Pencacah modulo-16 menggunakan IC 74393

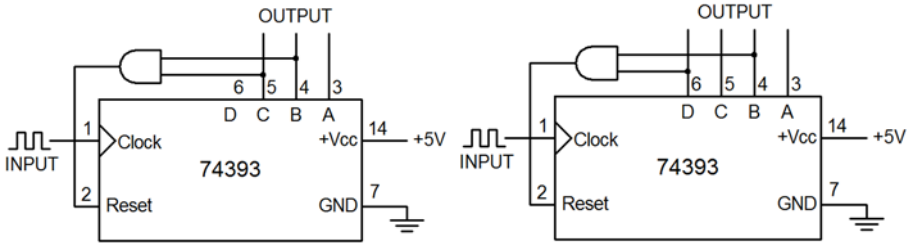
Untuk menyusun rangkaian tersebut, sediakan sebuah IC 74393, sebuah generator *clock*, sebuah peraga 7-segmen, dan empat buah LED. Tempatkan piranti-piranti tersebut di atas papan *breadboard.* Selanjutnya, pasang catu daya pada semua IC yang digunakan, pasang generator *clock* pada input, pasang indikator LED dan peraga heksadesimal pada output pencacah. Salah satu bentuk susunan rangkaian pencacah modulo-16 dengan IC 74393 menggunakan simulator *breadboard* ditunjukkan pada gambar berikut ini.



Gambar 114. Pencacah modulo-16 dengan IC 74393
menggunakan simulator *breadboard*

Jalankan simulasi dan atur **kecepatan simulasi (Sim Speed) hingga level 8,5**! Amati output rangkaian pencacah pada indikator LED maupun peraga heksadesimal! Anda akan melihat rangkaian pencacah menampilkan angka 0 sampai dengan F heksadesimal (16 pencacahan), dan akan kembali 0 setelah mencapai angka F.

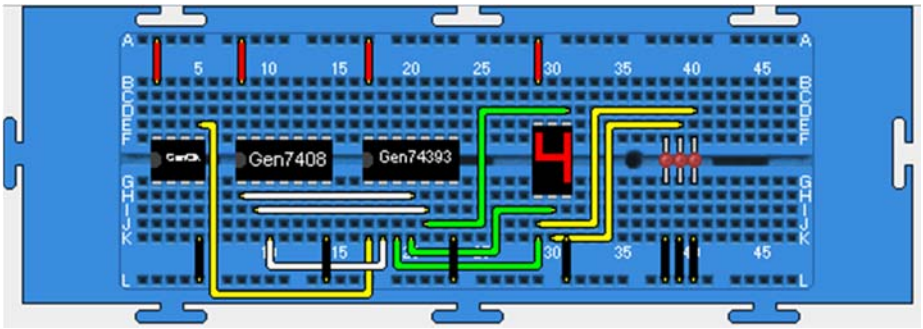
Coba lakukan prosedur yang sama untuk pencacah modulo-6 dan modulo-12 menggunakan IC 74393. Rangkaian keduanya ditunjukkan pada gambar berikut ini.



(a) (b)

Gambar 115.
Pencacah menggunakan IC 74393:
(a) modulo-6; dan (b) modulo-12

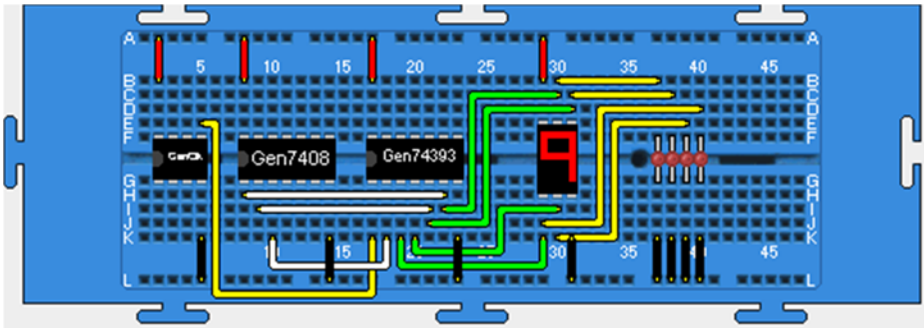
Susunan rangkaian pencacah modulo-6 dengan IC 74393 menggunakan simulator *beadboard* ditunjukkan pada gambar berikut ini.



Gambar 116. Pencacah modulo-6 dengan IC 74393 menggunakan simulator *breadboard*

Atur kecepatan simulasi hingga level 8,5! Jika simulasi dijalankan, anda akan melihat bahwa rangkaian pencacah modulo-6 melakukan pencacahan *clock* sebanyak 6 buah yakni *clock* ke-0 sampai dengan *clock* ke-5, dan pada *clock* ke-6 output pencacah akan *reset* kembali ke-0.

Untuk pencacah modulo-12 dengan IC 74393, susunan rangkaiannya di atas *breadboard* ditunjukkan gambar berikut ini.

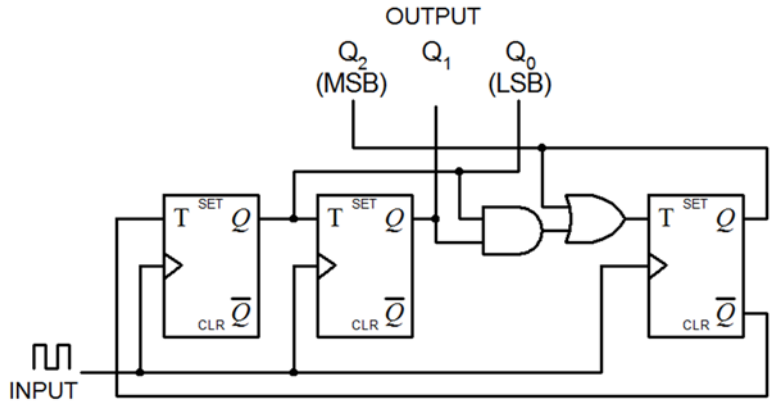


Gambar 115. Pencacah modulo-12 dengan IC 74393 menggunakan simulator *breadboard*

Jika anda menjalankan simulasi, akan terlihat bahwa pencacah modulo-12 melakukan pencacahan *clock* sebanyak 12 buah dari *clock* ke-0 sampai dengan *clock* ke-11 (B heksadesimal), dan pada *clock* ke-12 output pencacah *reset* kembali ke nilai 0.

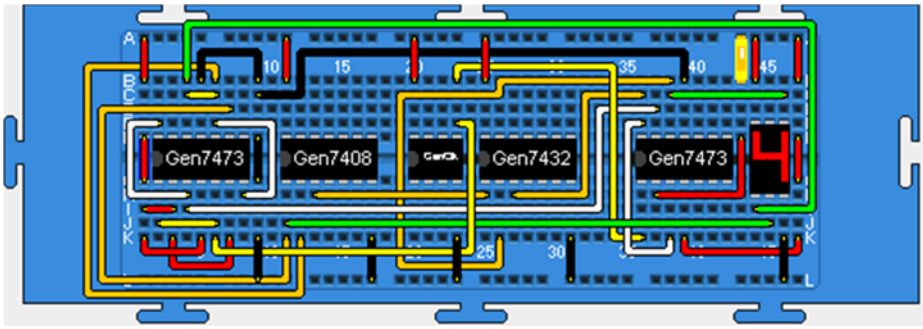
3. Menyelidiki Watak Sinkron Modulo-5

Rancangan pencacah sinkron modulo-5 menggunakan FFJK ditunjukkan pada gambar berikut ini.



Gambar 116. Pencacah sinkron modulo-5 menggunakan FFJK

Untuk menyusun pencacah pada Gambar 116, siapkan di atas papan *breadboard* piranti: 2 buah IC 7473 (FFJK), IC 7408 (AND), IC 7432 (OR), saklar tunggal, generator *clock*, dan peraga heksadesimal. Selanjutnya, hubungkan titik-titik yang ada sesuai dengan rangkaian di atas. Perlu diingat, anda harus memasang saklar tunggal yang dihubungkan ke pin *Clear* dari FFJK paling kanan untuk memicu flip-flop agar rangkaian mulai bekerja. Salah satu bentuk susunan rangkaian tersebut menggunakan simulator *breadboar* ditunjukkan pada gambar berikut ini.



Gambar 117.
Pencacah sinkron modulo-5 menggunakan simulator *breadboard*

Jalankan simulasi dan atur **kecepatan simulasi (Sim Speed) hingga level 8,5**! Untuk memicu rangkaian agar dapat mulai bekerja, berikan sinyal *Clear* pada FFJK paling kanan dengan meng-OFF-kan saklar, dan sesaat kemudian meng-ON-kannya kembali. Amati output rangkaian pencacah pada peraga heksadesimal! Anda akan melihat bahwa rangkaian melakukan pencacahan *clock* ke-0 sampai dengan *clock* ke-4, dan pada *clock* ke-5 outputnya akan reset kembali ke-0.

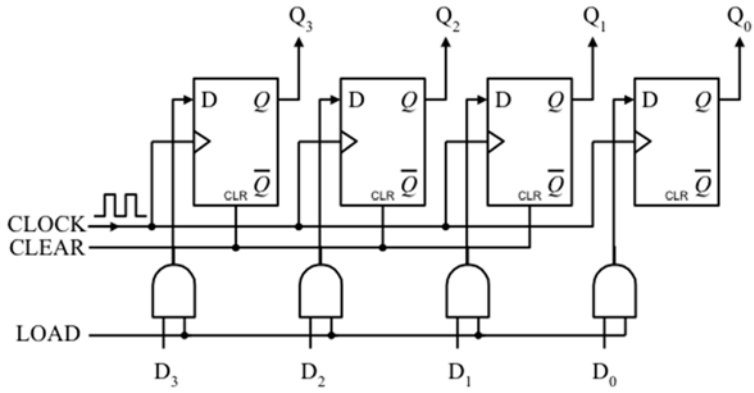
H. Register

Register merupakan memori n-bit. Jika sebuah flip-flop dapat digunakan untuk menyimpan data 1 bit, maka sederetan dari n flip-flop dapat digunakan untuk menyimpan data n-bit. Ada dua cara untuk memasukkan atau mengeluarkan data ke atau dari suatu register, yakni serial dan paralel. Pada cara serial, data dimasukkan atau dikeluarkan bit demi bit berganti-ganti lewat satu saluran, sedangkan pada cara paralel, n-bit dimasukkan atau dikeluarkan secara bersamaan lewat n saluran. Berdasarkan mekanisme pemasukan dan pengeluaran data tersebut, terdapat 4 macam register yakni *serial in-serial out* (SISO), *serial in-parallel out (*SIPO), *parallel in-parallel out* (PIPO), dan *parallel in-serial out* (PISO). Dalam keempat jenis register tersebut, terdapat jenis register geser kanan (*right shift register*) dan register geser kiri (*left shift register*).

Percobaan ini ditujukan untuk mempelajari cara kerja dan watak beberapa jenis register khususnya SIPO dan PIPO.

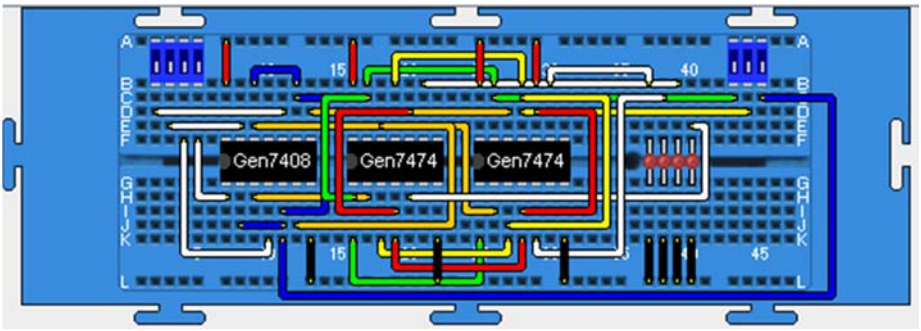
1. Menentukan Watak Register PIPO 4-bit Menggunakan FFD (IC 7474)

Rangkaian register PIPO 4-bit dapat direalisasikan dengan menggunakan flip-flop D (IC 7474) seperti ditunjukkan berikut ini.



Gambar 118. Rangkaian PIPO 4-bit dengan FFD

Rangkaian tersebut dapat diwujudkan dengan menempatkan dua buah IC 7474 (FFD), tiga buah saklar *double*, dan empat buah LED di atas *breadboard.* Selanjutnya, pasang catu daya pada semua IC yang digunakan, pasang saklar logika pada input *clock, clear* dan data ($D_3D_2D_1D_0$), pasang indikator LED pada output register (DCBA). Salah satu bentuk susunan rangkaian PIPO 4-bit menggunakan simulator *breadboard,* ditunjukkan pada gambar berikut ini.

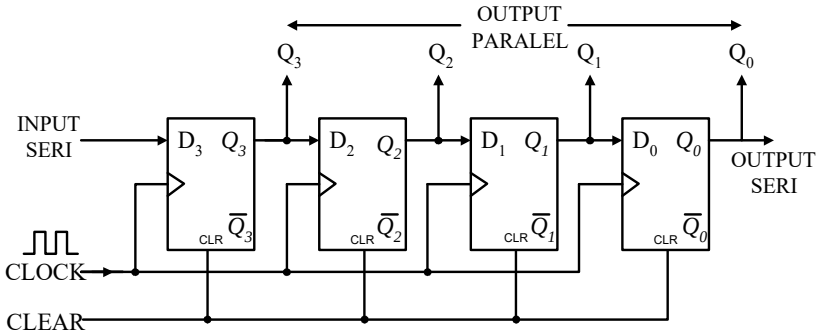


Gambar 119. Rangkaian PIPO 4-bit dengan simulator *breadboard*

Untuk mengoperasikan register tersebut, pastikan *Clear* bernilai 1 (saklar di-ON-kan) dan LOAD=1. Coba berikan data paralel pada inputnya ($D_3D_2D_1D_0$) misalnya 1010, kemudian sinyal *clock* (saklar di-ON-kan, sesaat kemudian di-OFF-kan). Apa yang terjadi? Anda akan melihat data 1010 telah tersimpan pada output register. Bersihkan output register dengan memberi sinyal CLEAR (saklar di-OFF-kan, sesaat kemudian di-ON-kan kembali). Pasang input data 1110, CLEAR=1, LOAD=0, berikan *clock* dan amati outputnya. Apa yang terjadi? Anda akan melihat register tidak dapat menyimpan data karena LOAD=0. Coba sekarang berikan LOAD=1, dan sinyal *clock*, maka anda akan melihat register dapat menyimpan data 1110 pada outputnya.

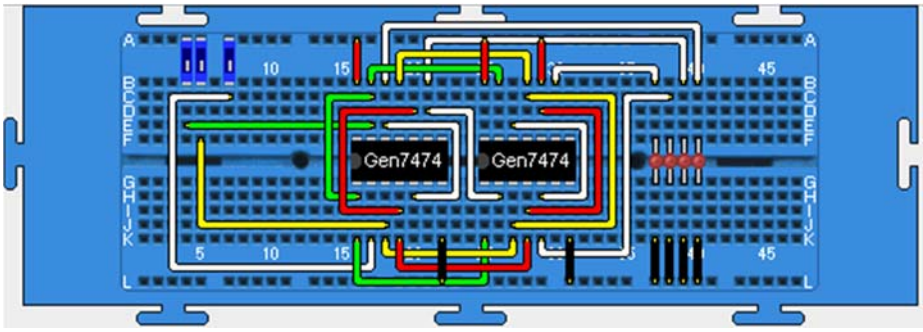
2. Menentukan Watak Register SIPO 4-bit Menggunakan FFD (IC 7474)

Rangkaian register SIPO 4-bit menggunakan flip-flop D (IC 7474) ditunjukkan pada gambar berikut ini.



Gambar 120. Rangkaian SIPO 4-bit menggunakan FFD

Coba susun rangkaian tersebut dengan terlebih dahulu menempatkan dua buah IC 7474 (FFD), tiga buah saklar tunggal, dan empat buah LED di atas *breadboard.* Selanjutnya, pasang catu daya pada semua IC yang digunakan, pasang saklar logika pada input *clock, clear* dan data serial, pasang indikator LED pada output register (DCBA). Salah satu bentuk susunan rangkaian SIPO 4-bit menggunakan simulator *breadboard,* ditunjukkan pada gambar berikut ini.



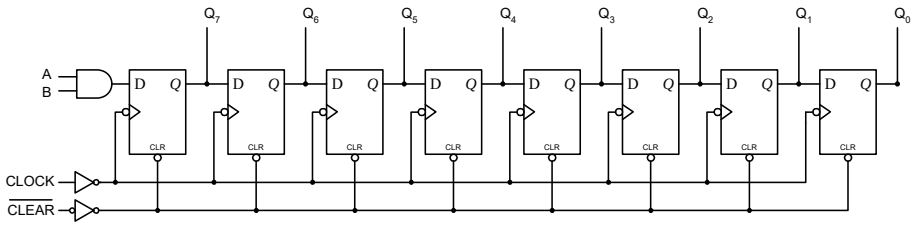
Gambar 121.
Rangkaian SIPO 4-bit menggunakan simulator *breadboard*

Untuk mempelajari cara kerja rangkaian, coba pasang data serial misalnya 1011 pada inputnya dengan cara:

a. ON-kan saklar *clear* (saklar paling kiri)
b. Pasang logika 1 pada input serial (ON-kan saklar paling kanan)
c. ON-kan saklar *clock* (saklar tengah), sesaat kemudian OFF-kan

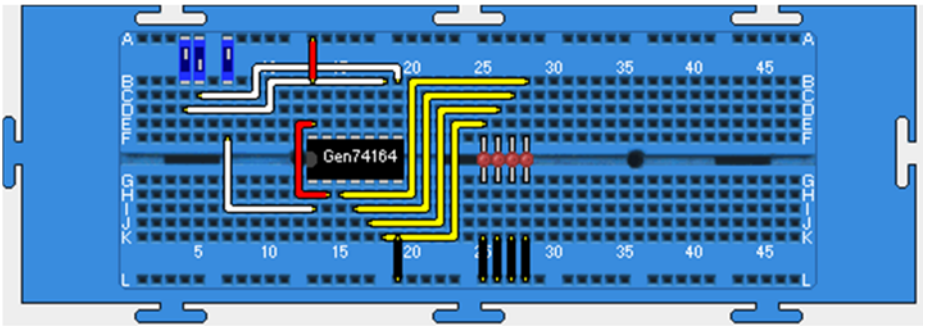
Sampai tahap ini anda telah memasukkan data seri 1 ke dalam register, terlihat outputnya 0001. Selanjutnya ulangi langkah b dan c di atas untuk data berikutnya. Masukkan data mulai dari MSB (most significant bit atau bit paling kiri) berturut-turut sampai terakhir bit paling kanan. Jika anda ingin membersihkan isi register OFF-kan saklar *clear* (paling kiri).

Selain dapat dibangun dengan menggunakan FFD, SIPO juga tersedia dalam bentuk IC 74164. Rangkaian internal IC 74164 ditunjukkan pada gambar berikut ini.



Gambar 122. Rangkaian internal IC 74164

Salah satu bentuk susunan rangkaian SIPO 4-bit dengan IC 74164 menggunakan simulator *breadboard,* ditunjukkan pada gambar berikut ini.



Gambar 121. Susunan SIPO 4-bit dengan IC 74164

Coba selidiki cara kerjanya dengan prosedur yang sama dengan SIPO 4-bit dengan FFD di atas! Jika eksperimen yang anda jalankan benar, maka akan diperoleh wakak yang sama antara register SIPO menggunakan IC 74164 dengan register SIPO dari FFD di atas.

Daftar Pustaka

Bailey, C. & Freeman, M. J. "A Java Bread-Board Simulator: Digital Circuit Simulation with an Open-Source Toolset". *IADIS International Journal on Computer Science and Information System,* Volume VV, 1, 2010, hlm. 13-25.

Freeman, M. 2010. *Getting Started with Java Breadboard in Windows*. Heslington: Department of Computer Science, The University of York.

Glass, N. 2002. *Java Digital Breadboard Simulator: A Simulator for an Educational Electronics Environment*.

Muchlas. 2013. *Dasar-dasar Rangkaian Digital*. Yogyakarta: UAD Press.

_____. 2015. "Developing a Teaching Model Using an Online Collaboration Approach for a Digital Technique Practical Work". *The Turkish Online Journal of Educational Technology,* 14(3), hlm. 63-69.

Muchlas & Novianta, M. A. 2015. "An Online Lab for Digital Electronics Course Using Information Technology Supports". *International Conference on Science in Information Technology (ICSITech)*, hlm. 299-302.

Profil Penulis

Dr. Muchlas, M. T.

Sejak 1987, ia aktif sebagai dosen pada Program Studi Pendidikan Fisika Universitas Ahmad Dahlan. Satu dekade kemudian, ia meniti karir akademik pada program studi Teknik Elektro meminati bidang Elektronika Digital dan Mikroprosesor, dilanjutkan menjadi staf pengajar pada Magister Pendidikan Vokasional di universitas yang sama untuk mata kuliah Teori & Strategi Belajar Pendidikan Teknologi dan Kejuruan.

Dr. Chris Bailey

Sejak 1999, ia menjadi dosen pada Departemen Ilmu Komputer, *University of York*, Inggris, mengajar mata kuliah bidang Mikroelektronika dan Arsitektur Komputer. Sebelumnya, ia bekerja di Departemen Teknik Elektro dan Elektronika, *University of Teesside*, pada berbagai peran selama tujuh tahun, kemudian menjadi Dosen Senior pada tahun 1997 di universitas yang sama.

Dr. Michael Freeman

Saat ini ia menjalani profesi sebagai dosen di Departemen Ilmu Komputer di *University of York*, Inggris, dengan minat pada bidang Arsitektur Perangkat Keras khususnya prosesor teks berkecepatan tinggi. Minat lainnya adalah sistem-sistem berbasis prosesor *multi-soft-core* dengan sistem operasi yang mendukung piranti-piranti FPGA untuk aplikasi robotika. Sebelum menjadi dosen, ia pernah menekuni pekerjaan sebagai *electronics engineer* dan *research assistant*.

Simulator Breadboard

Perangkat Pembelajaran Teknik Digital

Perkembangan teknologi informasi dan komunikasi yang sangat pesat saat ini, telah mempengaruhi seluruh aspek penyelenggaraan pendidikan, termasuk tatanan laboratorium pembelajaran. Kegiatan praktikum yang semula diselenggarakan menggunakan laboratorium *real*, sekarang banyak diimplementasikan dengan virtual lab. *Trend* penggunaannya semakin meningkat setelah banyak riset membuktikan bahwa perangkat ini dapat membantu meningkatkan efektivitas pembelajaran.

Simulator breadboard merupakan salah satu perangkat virtual lab yang dapat menyediakan alat dan bahan maya yang diperlukan dalam mendukung praktikum teknik digital. Perangkat lunak ini dapat melakukan simulasi watak rangkaian digital yang disusun menggunakan *breadboard* virtual.

Bagian awal buku ini mendeskripsikan secara lengkap cara pengoperasian *simulator breadboard* yang didahului dengan penjelasan penggunaan papan rangkaian tersebut dalam bentuk real. Pada bagian ini dijelaskan pula persyaratan perangkat keras dan perangkat lunak yang diperlukan, serta cara melakukan instalasinya. Selanjutnya, buku ini menguraikan kelengkapan yang disediakan oleh simulator berupa piranti IC (*integrated circuits*), komponen input dan output, serta kabel penghubung. Buku ini memberikan pula panduan kepada pembaca dalam pembuatan rangkaian baru, perbaikan rangkaian, penggunaan papan ganda, dan pembukaan banyak berkas rangkaian dalam satu layar.

Pada bagian akhir, buku ini menyediakan tidak kurang dari 35 buah rangkaian virtual untuk mendukung praktikum teknik digital yang meliputi topik: (1) watak gerbang logika dasar dan universal; (2) minimalisasi rangkaian logika; (3) komparator dan penjumlah biner; (4) multiplekser dan demultiplekser; (5) enkoder dan dekoder; (6) flip-flop; (7) pencacah; serta (8) register.

Buku ini sangat sesuai digunakan oleh dosen, mahasiswa, instruktur, guru dan siswa sekolah menengah atas maupun sekolah menengah kejuruan (SMK) sebagai salah satu rujukan dalam membantu mempelajari dasar-dasar teknik digital.

UAD PRESS
(Anggota IKAPI dan APPTI)
Kampus II Universitas Ahmad Dahlan
Jl. Pramuka No.42, Pandeyan, Kec. Umbulharjo,
Daerah Istimewa Yogyakarta 55161
E-mail: uadpress@uad.ac.id
HP/WA: 088239499820

ISBN 602-0737-80-2
9 786020 737805