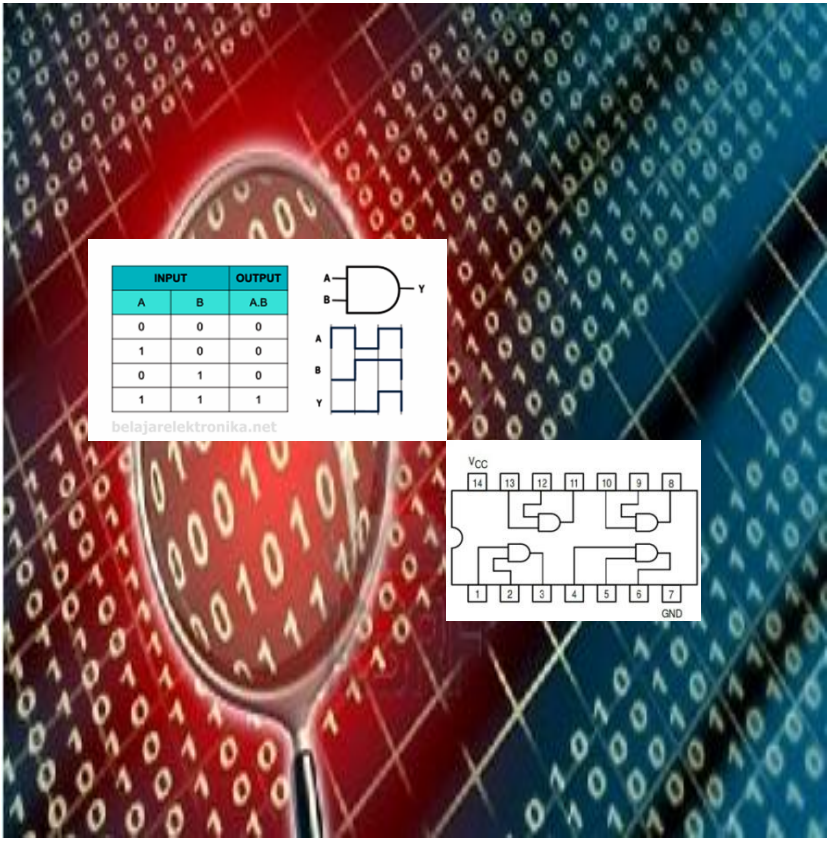
SISTEM DIGITAL

Oleh : Dr. Hindarto, S.Kom, MT

(Digunakan di lingkungan sendiri, sebagai buku ajar mata kuliah Sistem Digital)



FAKULTAS SAINS DAN TEKNOLOGI

UNIVERSITAS MUHAMMADIYAH SIDOARJO

2019

KATA PENGANTAR

Alhamdulillah kami panjatkankehadirat Allah SWT, serta sholawat dan salam tercurahkan kepada junjungan nabi kita nabi Muhammad SAW. Buku Sistem digital ini dimaksudkan sebagai pegangan kegiatan belajar mengajar di Fakultas Teknik Umsida, khususnya untuk program studi Informatika. Pada buku ini diuraikan tentang sitem bilangan, rangkaian logika Dasar, teknik pencacah, dan register, cara kerja dan aplikasinya.

Penulis menyadari bahwa pada penyusunan buku Sistem Digital ini jauh dari sempurna, baik dari segi penyusunan, bahasan, ataupun penulisannya. Oleh karena itu penulis mengharapkan kritik dan saran yang sifatnya membangun, guna menjadi acuan dalam bekal pengalaman untuk lebih baik dimasa yang akan datang. Semoga buku ini dapat dipergunakan sebagai salah satu acuan, petunjuk maupun pedoman bagi pembaca.

Sidoarjo, 20 Oktober 2018

Penulis

DAFTAR ISI

BAB 1
DASAR SISTEM BILANGAN

1. Dasar sistem bilangan

Pengertian dan Macam Sistem Bilangan Komputer atau Number System adalah Suatu cara untuk mewakili besaran dari suatu item fisik. Sistem Bilangan menggunakan suatu bilangan dasar atau basis (base / radix) yang tertentu. Dalam hubungannya dengan komputer, ada 4 Jenis Sistem Bilangan yang dikenal yaitu : Desimal (Basis 10), Biner (Basis 2), Oktal (Basis 8) dan Hexadesimal (Basis 16). Berikut penjelesan mengenai 4 Sistem Bilangan ini.

a. Bilangan Desimal (Basis 10)

Desimal (Basis 10) adalah Sistem Bilangan yang paling umum digunakan dalam kehidupan sehari-hari. Sistem bilangan desimal menggunakan basis 10 dan menggunakan 10 macam simbol bilangan yaitu : 0, 1, 2, 3, 4, 5, 6, 7, 8 dan 9. Sistem bilangan desimal dapat berupa integer desimal (decimal integer) dan dapat juga berupa pecahan desimal (decimal fraction).

Untuk melihat nilai bilangan desimal dapat digunakan perhitungan seperti berikut, misalkan contoh bilangan desimal adalah 8598. Ini dapat diartikan :

Absolute Value

Position Value

$1 \times 10^3 = 1000$

$5 \times 10^2 = 500$

$9 \times 10^1 = 90$

$8 \times 10^0 = 8$ +

1598

Dalam gambar diatas disebutkan Absolut Value dan Position Value. Setiap simbol dalam sistem bilangan desimal memiliki Absolut Value dan Position Value. Absolut value adalah Nilai Mutlak dari masing-masing digit bilangan. Sedangkan Position Value adalah Nilai Penimbang atau bobot dari masing-masing digit bilangan tergantung dari letak posisinya yaitu bernilai basis di pangkatkan dengan urutan posisinya. Untuk lebih jelasnya perhatikan tabel 1.1 dibawah ini.

Table 1.1 Tabel Posisi bilangan decimal

Posisi Digit (dari kanan)	Position Value
1	$10^0 = 1$
2	$10^1 = 10$
3	$10^2 = 100$
4	$10^3 = 1000$
5	$10^4 = 10000$

Dengan begitu maka bilangan desimal 8598 bisa diartikan sebagai berikut :

1598_{10} = (1 x 1000) + (5 x 100) + (9 x 10) + (8 x 1)

Sistem bilangan desimal juga bisa berupa pecahan desimal (decimal fraction), misalnya : 183,75 yang dapat diartikan :

$1 \times 10^2 = 100$

$2 \times 10^1 = 20$

$3 \times 100 = 3$

$7 \times 10^{-1} = 0,7$

$5 \times 10^{-2} = 0,05$ +

123,75

Contoh :

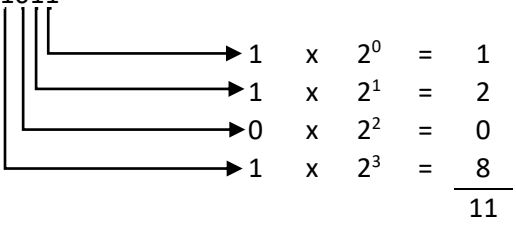
1. $125_{10} = 1 \times 10^2 + 2 \times 10^1 + 5 \times 10^0$

2. $0{,}21_{10} = 0 \times 10^0 + 2 \times 10^{-1} + 11 \times 10^{-2} + 6 \times 10^1 + 1 \times 10^{-2}$
3. $3407{,}108_{10} = 3 \times 10^3 + 4 \times 10^2 + 7 \times 10^0 + 1 \times 10^{-1} + 8 \times 10^{-3}$

b. Bilangan Biner (Basis 2)

Biner (Basis 2) adalah Sistem Bilangan yang terdiri dari 2 simbol yaitu 0 dan 1. Bilangan Biner ini dipopulerkan oleh John Von Neumann. Contoh Bilangan Biner 1001, Ini dapat di artikan (Di konversi ke sistem bilangan desimal) menjadi sebagai berikut :

1011



Position Value dalam sistem Bilangan Biner merupakan perpangkatan dari nilai 2 (basis), seperti pada tabel 1.2 berikut ini :

Table 1.2 Tabel Posisi bilangan Biner

Posisi Digit (dari kanan)	Position Value
1	$2^0 = 1$
2	$2^1 = 2$
3	$2^2 = 4$
4	$2^3 = 8$
5	$2^4 = 16$

Berarti, Bilangan Biner 1001 perhitungannya adalah sebagai berikut :
$1011_2 = (1 \times 8) + (0 \times 4) + (1 \times 2) + (1 \times 1)$

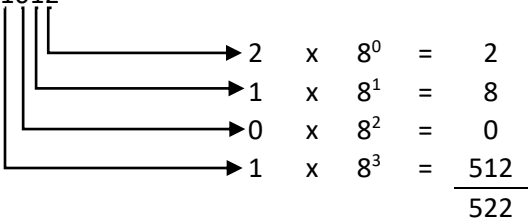
Contoh :

1. 1001_2 $= 1 \times 2^3 + 1 \times 2^0$
   $= 8 + 1$
   $= 9_{10}$
2. 0,100 $= 0 \times 2^0 + 1 \times 2^{-1} + 0 \times 2^{-2} + 0 \times 2^{-3}$
   $= 0 + 0,5 + 0 + 0$
   $= 0,500$
3. 10,01 $= 1 \times 2^1 + 1 \times 2^{-2}$
   $= 2 + 0,25$
   $= 2,25_{10}$

c. Bilangan Oktal (Basis 8)

Oktal (Basis 8) adalah Sistem Bilangan yang terdiri dari 8 Simbol yaitu 0, 1, 2, 3, 4, 5, 6, 7. Contoh Oktal 1024, Ini dapat di artikan (Di konversikan ke sistem bilangan desimal) menjadi sebagai berikut :

1012



Position Value dalam Sistem Bilangan Oktal merupakan perpangkatan dari nilai 8 (basis), seperti pada tabel 1.3 berikut ini :

Table 1.3 Tabel Posisi bilangan Oktal

Posisi Digit (dari kanan)	Position Value
1	$8^0 = 1$
2	$8^1 = 8$
3	$8^2 = 64$
4	$8^3 = 512$
5	$8^4 = 4096$

Berarti, Bilangan Oktal 1022 perhitungannya adalah sebagai berikut :
$1012_8 = (1 \times 512) + (0 \times 64) + (1 \times 8) + (2 \times 1)$

Contoh :

$547{,}35_8 = 5 \times 8^2 + 4 \times 8^1 + 7 \times 8^0 + 3 \times 8^{-1} + 5 \times 8^{-2}$
$= 320 + 32 + 7 + 0{,}375 + 0{,}078125$
$= 359{,}453125_{10}$

d. Bilangan Hexadesimal (Basis 16)

Hexadesimal (Basis 16), Hexa berarti 6 dan Desimal berarti 10 adalah Sistem Bilangan yang terdiri dari 16 simbol yaitu 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A(10), B(11), C(12), D(13), E(14), F(15). Pada Sistem Bilangan Hexadesimal memadukan 2 unsur yaitu angka dan huruf. Huruf A mewakili angka 10, B mewakili angka 11 dan seterusnya sampai Huruf F mewakili angka 15.

Contoh Hexadesimal F3D4, Ini dapat di artikan (Di konversikan ke sistem bilangan desimal) menjadi sebagai berikut :

F3D9

→	9	x	16^0	=	9
→	13	x	16^1	=	208
→	3	x	16^2	=	768
→	15	x	16^3	=	61440
					62425

Position Value dalam Sistem Bilangan Hexadesimal merupakan perpangkatan dari nilai 16 (basis), seperti pada tabel berikut ini :

Table 1.4 Tabel Posisi bilangan hexadecimal

Posisi Digit (dari kanan)	Position Value
1	$16^0 = 1$
2	$16^1 = 16$
3	$16^2 = 256$
4	$16^3 = 4096$
5	$16^4 = 65536$

Berarti, Bilangan Hexadesimal F3DA perhitungannya adalah sebagai berikut :

$F3D9_{16} = (15x16^3) + (3x16^2) + (13x16^1) + (9x16^0)$

$= 61440 + 768 + 208 + 9$

$= 624425$

Contoh :

$584AE9_{16} = 5 \times 16^5 + 8 \times 16^4 + 4 \times 16^3 + 10 \times 16^2 + 14 \times 16^1 + 9x\ 16^0$

$= 5242880 + 524288 + 16384 + 2560 + 224 + 9$

$= 5786345_{10}$

2. Konversi Bilangan

Konversi bilangan adalah proses mengubah bentuk bilangan satu ke bentuk bilangan lain yang memiliki nilai yang sama. Misal: nilai bilangan desimal 12 memiliki nilai yang sama dengan bilangan octal 15. Nilai bilangan biner 10100 memiliki nilai yang sama dengan 24 dalam octal dan seterusnya.

Konversi bilangan biner, octal atau hexadesimal menjadi bilangan desimal.

Konversi dari bilangan biner, octal atau hexa menjadi bilangan desimal memiliki konsep yang sama. Konsepnya adalah bilangan tersebut dikalikan basis bilangannya yang dipangkatkan 0,1,2 dst

dimulai dari kanan. Untuk lebih jelasnya silakan lihat contoh konversi bilangan di bawah ini.

a. **Konversi bilangan octal ke desimal.**

Cara mengkonversi bilangan octal ke desimal adalah dengan mengalikan satu-satu bilangan dengan 8 (basis octal) pangkat 0 atau 1 atau 2 dst dimulai dari bilangan paling kanan. Kemudian hasilnya dijumlahkan. Misal, 137(octal) = $(7x8^0) + (3x8^1) + (1x8^2)$ = 7+24+64 = 95 (desimal).

Contoh :

136_8 = ... $_{10}$

1	x	8^2	=	64	
3	x	8^1	=	24	
6	x	8^0	=	6	+
Nilai Dalam Desimal				**94**	

b. **Konversi bilangan biner ke desimal.**

Cara mengkonversi bilangan biner ke desimal adalah dengan mengalikan satu-satu bilangan dengan 2 (basis biner) pangkat 0 atau 1 atau 2 dst dimulai dari bilangan paling kanan. Kemudian hasilnya dijumlahkan. Misal, 11000(biner) = $(1x2^0) + (0x2^1) + (0x2^2) + (1x2) + (0x2^2)$ = 0+0+0+8+16 = 24 (desimal).

Contoh :

11000_2 = ... $_{10}$

1	x	2^4	=	16
1	x	2^3		8

0	x	2^2		0	
0	x	2^1	=	0	
0	x	2^0	=	0	+
Nilai Dalam Desimal				**24**	

c. **Konversi bilangan hexadesimal ke desimal.**

Cara mengkonversi bilangan biner ke desimal adalah dengan mengalikan satu-satu bilangan dengan 16 (basis hexa) pangkat 0 atau 1 atau 2 dst dimulai dari bilangan paling kanan. Kemudian hasilnya dijumlahkan. Misal, 7A9A(hexa) = ($Ax16^0$) + ($9x16^1$) + ($Ax16^2$)+($7x16^3$) = 10+144+2560+28672 = 31386 (desimal).

Contoh :

$7A9A_{16}$ = ... 10

7	x	16^3	=	28672	
A=10	x	16^2	=	2560	
9	x	16^1	=	144	
A=10	x	16^0	=	10	+
Nilai Dalam Desimal				**31386**	

Konversi bilangan desimal menjadi bilangan biner, octal atau hexadesimal.

Konversi dari bilangan desimal menjadi biner, octal atau hexadesimal juga memiliki konse yang sama. Konsepnya bilangan desimal harus dibagi dengan basis bilangan tujuan, hasilnya dibulatkan kebawah dan sisa hasil baginya (remainder) disimpan. Ini dilakukan terus menerus hingga hasil bagi < basis bilangan tujuan. Sisa bagi ini kemudian diurutkan dari yang paling akhir hingga yang paling awal dan inilah yang merupakan hasil konversi bilangan tersebut.

d. **Konversi bilangan desimal ke biner.**

Cara konversi bilangan desimal ke biner adalah dengan membagi bilangan desimal dengan 2 dan menyimpan sisa bagi per seitap pembagian terus hingga hasil baginya < 2. Hasil konversi adalah urutan sisa bagi dari yang paling akhir hingga paling awal.

Contoh :

124(desimal) = ... (biner)

124/2 = 62 sisa bagi 0

62/2= 31 sisa bagi 0

31/2=15 sisa bagi 1

15/2=7 sisa bagi 1

7/2=3 sisa bagi 1

3/2=1 sisa bagi 1

Hasil konversi: 1111100

Atau dengan Cara lain yang mencari bilangan dari basis 10 ke basis 2 (dua)

Contoh :

Ubahlah bilangan 98 basis 10 ke dalam basis-2 yang setara !

97_{10} $= \sum(N \times n^a) = \sum(N \times 2^a)$

$= N \times 64 + n \times 32 + N \times 2^1$

$=1x2^6+1x2^5+1x2^1$ (semua posisi belum diperhitungkan)

$= 1 \times 2^6 + 1 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0x\ 2^1 + 1 \times 2^0$

$= 1100001$

$= 1100001_2$

Perhatikan bahwa 0 ditempatkan dalam posisi 2^4, 2^3, 2^2, dan 2^0 karena semua posisi harus diperhitungkan.

e. **Konversi bilangan desimal ke octal.**

Cara konversi bilangan desimal ke octal adalah dengan membagi bilangan desimal dengan 8 dan menyimpan sisa bagi per seitap pembagian terus hingga hasil baginya < 8. Hasil konversi adalah urutan sisa bagi dari yang paling akhirhingga paling awal.

Contoh 1 :

1326_{10} = ... $_8$

1326/8 =165 sisa 6

165/8 = 20 sisa 5

20/8 = 2 sisa 4

Bilangan Octal dari 1326_{10} adalah 2456

Atau dengan Cara lain yang mencari bilangan dari basis 10 ke basis 8 (delapan)

Contoh :

Ubahkah bilangan 1367_{10} ke dalam basis- 8 yang setara !

1367_{10} $= \sum(N \times n^a)$

$= \sum(N \times 8^a)$

$= N \times 512\ N \times 64 + Nx8$

$= 2 \times 8^3 + 5 \times 8^2 + 3 \times 8^1$

(semua posisi belum diperhitungkan)

$= 2 \times 8^3 + 5 \times 8^2 + 2 \times 8^1 + 7 \times 8^0$

$= 2527_8$.

Perhatikan bahwa 0 ditempatkan dalam posisi 8^0 karena semua posisi harus diperhitungkan.

f. **Konversi bilangan desimal ke hexadesimal.**

Cara konversi bilangan desimal ke octal adalah dengan membagi bilangan desimal dengan 16 dan menyimpan sisa bagi per seitap

pembagian terus hingga hasil baginya < 16. Hasil konversi adalah urutan sisa bagi dari yang paling akhir hingga paling awal. Apabila sisa bagi diatas 9 maka angkanya diubah, untuk nilai 10 angkanya A, nilai 11 angkanya B, nilai 12 angkanya C, nilai 13 angkanya D, nilai 14 angkanya E, nilai 15 angkanya F.

Contoh :

23601_{10} = ... $_{16}$

23601/16 = 1475 sisa 1

1475/16 = 92 sisa 3

92/16 = 5 sisa 12 = C

Bilangan Hexa dari 23601_{16} adalah 5C31

Atau dengan Cara lain yang mencari bilangan dari basis 10 ke basis 16

Ubahlah bilangan 19007_{10} ke dalam heksadesimal yang setara !

19007_{10} $= \sum(N \times n^a)$

$= \sum(N \times 16^a)$

$= N \times 4096 + N \times 256 + N \times 16 + N \times 16^0$

$= 4 \times 16^3 + A \times 16^2 + 3 \times 16^1 + 15x\ 16^0$

(semua posisi belum diperhitungkan)

$= 4A3E_{16}$

g. **Konversi bilangan octal ke biner.**

Konversi bilangan octal ke biner caranya dengan memecah bilangan octal tersebut persatuan bilangan kemudian masing-masing diubah kebentuk biner tiga angka. Maksudnya misalkan kita mengkonversi nilai 2 binernya bukan 10 melainkan 010. Setelah itu hasil seluruhnya diurutkan kembali.

Contoh:

146_8 = ... $_2$

1	4	6
binernya	binernya	binernya
001	100	110

Bilangan Biner dari 146_8 adalah : 001100110_2

h. **Konversi bilangan biner ke octal.**

Konversi bilangan biner ke octal sebaliknya yakni dengan mengelompokkan angka biner menjadi tiga-tiga dimulai dari sebelah kanan kemudian masing-masing kelompok dikonversikan kedalam angka desimal dan hasilnya diurutkan. Contoh :

Contoh:

11001100_2 = ... $_8$

11	001	100
oktalnya	oktalnya	oktalnya
3	1	4

Bilangan oktal dari 11001100_2 adalah :314_8

i. **Konversi bilangan hexadesimal ke biner.**

Sama dengan cara konversi bilanga octal ke biner, bedanya kalau bilangan octal binernya harus 3 buah, bilangan desimal binernya 4 buah. Misal kita konversi 2 hexa menjadi binerhasilnya bukan 10 melainkan 0010.

Contoh :

$A7E_{16}$ = ... $_2$

A	7	E
binernya	binernya	binernya
1010	0111	1110

Bilangan Biner dari $A7E_{16}$ adalah :101001111110_2

j. **Konversi bilangan biner ke hexadesimal.**

Teknik yang sama pada konversi biner ke octal. Hanya saja pengelompokan binernya bukan tiga-tiga sebagaimana pada bilangan octal melainkan harus empat-empat.

Contoh :

11001100_2 = ... $_{16}$

1100	1100
hexanya	hexanya
C	C

Bilangan Hexa dari 11001100_2 adalah : CC_{16}

Konversi bilangan hexadesimal ke octal dan sebaliknya

k. **Konversi bilangan octal ke hexadesimal.**

Teknik mengonversi bilangan octal ke hexa desimal adalah dengan mengubah bilangan octal menjadi biner kemudian mengubah binernya menjadi hexa. Ringkasnya octal->biner>hexa.

Contoh :

724_8 = ... $_{16}$

7	2	4	
binernya	binernya	binernya	
111	010	100 ↓	Dijadikan 4 digit
1	1101	0100	
Hexanya	Hexanya	Hexanya	
1	D	4	

Bilangan Biner dari 724_8 adalah :$1D4_{16}$

l. **Konversi bilangan hexadesimal ke octal.**

Begitu juga dengan konversi hexa desimal ke octal yakni dengan mengubah bilangan hexa ke biner kemudian diubah menjadi bilangan octal. Ringkasnya hexa->biner->octal.

Contoh :

$1D4_{16}$ = ... $_8$

1	D	4	
binernya	binernya	binernya	
0001	1101	0100 ↓	Dijadikan 4 digit
111	010	100	
Hexanya	Hexanya	Hexanya	
7	2	4	

Bilangan Biner dari $1D4_{16}$ adalah : 724_8

Diantara fungsi konversi bilangan diantaranya adalah untuk menghitung maksimum usable host pada blok IP address.

3. Operasi Bilangan Biner

3.1 Operasi Penjumlahan

Ada beberapa hal umum yang harus diketahui dalam penjumlahan bilangan biner yaitu sebagai berikut :

0 + 0 = 0

0 + 1 = 1

1 + 0 = 1

1 + 1 = 10 → 0 + carry 1 ditempatkan pada posisi berikutnya

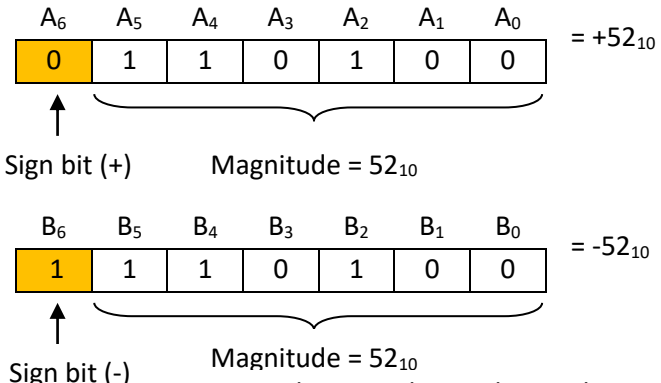
1 + 1 + 1 = 11 → 1 + carry 1 ditempatkan pada posisi berikutnya

Contoh :

	011 (3)		1001 (9)		11.011 (3.375)
+	110 (6)	+	1111(15)	+	10.110 (2.750)
	1001 (9)		11000 (24)		110.001 (6.125)

Bilangan Bertanda

Sebagian besar komputer digital menangani bilangan negative sebagai bilangan positif, sehinga diperlukan sign (tanda) bilangan + atau -. Tanda tersebut diwakili oleh satu bit yang disebut sebagai sign bit. Dimana 0 merupakan tanda positif dan 1 merupakan tanda negative. Bit tanda ini menempati posisi bit paling kiri atau pada bagian MSB seperti pada gambar 1.1 dibawah.



Gambar 1.1 Bilangan bertanda

Sign bit digunakan untuk menyatakan bilangan positif dan negative yang disimpan dalam bentuk bilangan biner. Pada Gambar 2.1 diatas terlihat bahwa bilangan tersebut terdiri dari 1 sign bit dan 6 magnitude bit. Magnitude bit merupakan bilangan biner yang nilainya sama dengan bilangan decimal yang mewakilinya. Prinsip ini dikenal dengan nama **signmagnitude system** untuk menyatakan bilangan biner bertanda.

System yang umum digunakan untuk menyatakan bilangan biner bertanda ini adalah 2's complement system. Komplemen 2 ini digunakan untuk menyatakan bilangan bertanda, karena untuk melakukan operasi pengurangan, sebenarnya operasi yang dilakukan adalah penjumlahan.

Langkah langkah melakukan komplemen 2 :

1's-Complement Form

Komplemen 1 dari sebuah bilangan biner merupakan diperoleh dari perubahan setiap 0 menjadi 1, dan 1 menjadi 0.

Contoh :

```
1 0 1 1 0 1  original binary number
↓ ↓ ↓ ↓ ↓ ↓
0 1 0 0 1 0  complement each bit to form 1's complement
```

Komplemen 1 dari **101101 adalah 010010**

2's Complement Form

Komplemen 2 dari sebuah bilangan biner diperoleh dari hasil komplemen 1 ditambah dengan 1 pada posisi LSB.

```
1 0 1 1 0 1      binary equivalent of 45
0 1 0 0 1 0      complement each bit to form 1's complement
+          1     add 1 to form 2's complement
-----------
 0 1 0 0 1 1     2's complement of original binary number
```

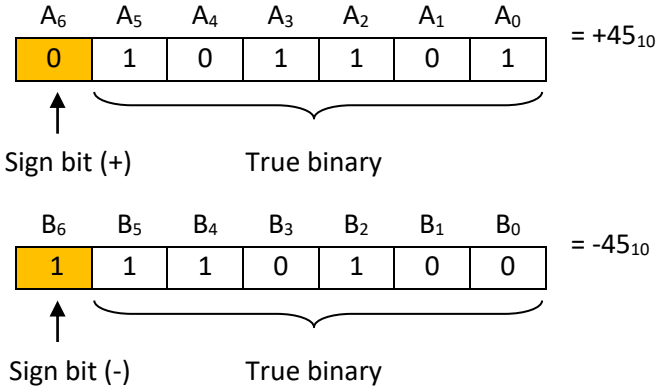
Sehingga 010011 merupakan komplemen 2 dari 101101.

Representing Signed Numbers Using 2's Complement

Sistem komplemen 2 digunakan untuk menyatakan bilangan bertanda.

- Jika bilangan positif, magnitude dinyatakan dalam bentuk nilai bilangan biner asli dan sign bit adalah 0 ditempatkan pada bagian MSB.
- Jika bilangan negative, maka magnitude merupakan bentuk komplemen 2, dan sign bit adalah 1 ditempatkan pada bagian MSB.

2 ketentuan diatas dapat dilihat pada Gambar 2.2 dibawah.



Gambar 1.2 Bilangan bertanda denan komplemen 2

Penjumlahan Complement Dua

Case I: Two Positive Numbers.

Menjumlahkan 2 bilangan positif sama seperti penjumlahan bilangan biner diatas, Contoh +9 dan +4

+9 → 0 1001 (augend)
+4 → 0 0100 (addend)
0 1101 (sum = +13)
Sign bits

Case II: Positive Number and Smaller Negative Number.

Contoh : +9 dan -4

Langkah 1 : mencari nilai complemen 2 dari -4

+4	0 1 0 0
C'1	1 0 1 1
	1
C'2	1 1 0 0

Langkah 2 : Menjumlahkan +9 dengan C'2 -4

+9 →	0	1001 (augend)
-4 →	1	1100 (addend)
1	0	0101 (sum = +13)

This carry disregarded, the result is 00101 (sum = +5)

Pada kasus ini, sign bit dari addend (penambah) adalah 1. Hasil dari penjumlahan menghasilkan sebuah carry pada bagian akhir, dan carry ini diabaikan, sehingga hasil akhirnya adalah 0 0101 (+5)

Case III: Positive Number and Larger Negative Number.

Contoh : -9 dan +4

Langkah 1 : mencari nilai complemen 2 dari -9

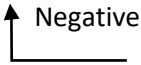
+9	1 0 0 1
C'1	0 1 1 0
	1
C'2	0 1 1 1

Langkah 2 menjumlahkan c'2 -9 dengan +4

-9 → 10111

+4 → 00100

11011 (sum = -5)



Dari hasil diperoleh 1 1011, dimana sign bit nya adalah 1 sehingga bilangannya adalah negative, dan magnitudenya merupakan hasil komplemen dua yaitu 1011, sehingga bilangan aslinya adalah:

C'2	1 0 1 1	
C'1	0 1 0 0	
	1+	
Asli	0 1 0 1	→ 5

Case IV: Two Negative Numbers

Contoh : -9 dan -4

Langkah 1 : mencari nilai complemen 2 dari -9

+9	1 0 0 1
C'1	0 1 1 0
	1
C'2	0 1 1 1

Langkah 2 : mencari nilai complemen 2 dari -4

+4	0 1 0 0
C'1	1 0 1 1
	1
C'2	1 1 0 0

Langkah 3 menjumlahkan

-9 → 10111

-4 → 11100

1 10011 (sum = -5)

Negative sign bit

This carry disregarded, the result is 10011(sum=-13)

Dari hasil diperoleh 1 1 0011, dimana 1 bit carry diabaikan, sign bit nya adalah 1 sehingga bilangannya adalah negative, dan magnitudenya merupakan hasil komplemen dua yaitu 0011, sehingga bilangan aslinya adalah :

C'2	0 0 1 1
C'1	1 1 0 0
	1+
Asli	1 1 0 1 → 13

Case V: Equal and Opposite Numbers

Contoh : +9 dan -9

Langkah 1 : mencari nilai complemen 2 dari -9

+9	1 0 0 1
C'1	0 1 1 0
	1
C'2	0 1 1 1

Langkah 2 menjumlahkan

-9 → 10111

+9 → 01001

0 1 00000

disregarded, the result is 00000(sum=+0)

3.2 Perkalian Bilangan Biner

Perkalian bilangan biner dilakukan dengan cara yang sama dengan perkalian bilangan decimal. Proses menjadi lebih sederhana karena kita hanya mengalikan digit 1 atau 0, dan tidak melibatkan digit lainnya.

Contoh : 9 x 11

1001 ← multiplicand = 9_{10}

1011 ← multiplier = 11_{10}

1001

1001 } Partial products

0000

1001

1100011 ➔ Final product = 99_{10}

Pada sebagian mesin digital, penjumlahan hanya dapat dilakukan antara 2 bilangan biner pada satu waktu, sehingga selama perkalian, maka penjumlahan tidak dilakukan seluruhnya pada satu waktu, tetapi penjumlahan dilakukan untuk 2 partial product per satuan waktunya. Pertama ditambah dengan kedua, hasilnya ditambah dengan ketiga, dan seterusnya seperti ilustrasi dibawah ini.

Add { 1001 ← first partial product
1001 ← second partial product shifted left

Add { 11001 ← sum of first two partial product
0000 ← third partial product shifted left

Add { 011011 ← first partial product
1001 ← second partial product shifted left
1100011 ← sum of four partial product,wich equal final total product

3.3 Pembagian Bilangan Biner

Proses Pembagian satu bilangan biner (dividend) dengan bilangan biner lainnya (divisor) sama dengan pembagian pada bilangan decimal. Prosesnya lebih sederhana dalam bilangan biner karena nilai yang dilibatkan hanya 0 atau 1.

Contoh :

```
   0011              0010.1
11/1001           100/1010.0
  011               100
  0011                100
   11                 100
    0                   0
```

4. Kode Bilangan

4.1 Kode BCD

Seperti telah diterangkan dalam uraian mengenai sistem bilangan oktal dan heksadesimal di bagian depan, untuk menyatakan 1 angka desimal diperlukan 4 angka biner. Tetapi dengan 4 bit sebenarnya dapat dinyatakan 16 macam simbol yang berbeda sehingga kesepuluh simbol dalam bilangan desimal dapat dinyatakan dengan beberapa himpunan (set) kode yang berbeda. Perlu dibedakan dengan tegas antara pengkodean dan konversi. Kalau suatu bilangan dikonversikan ke bilangan lain maka kedua bilangan itu mempunyai harga/nilai. Sebagai contoh, kalau angka 8 desimal dikonversikan ke biner, maka satu- satunya pilihan adalah 1000. Tetapi kalau angka 8 ini dikodekan ke biner, ada bermacam-macam kode yang dapat dibentuk, walaupun hanya terdiri atas 4 bit. Dari bermacam-macam kode untuk angka-angka desimal, kode BCD (singkatan dari *Binary Coded Decimal*) merupakan kode yang paling sederhana karena kode itu sendiri merupakan konversi dari desimal ke biner.

Sistem BCD digunakan untuk menampilkan digit desimal sebagai kode biner 4 bit. Kode ini berguna untuk menampilkan angka numerik dari 0 sampai 9 seperti pada jam digital atau voltmeter. Untuk mengubah nilai BCD ke biner, ubah tiap digit desimal ke 4 bit biner.

Contoh soal:

1. Konversikan bilangan desimal 683_{10} ke nilai BCDnya.

Penyelesaian: 6 8 3

0110 1000 0011 = $0110\ 1000\ 0011_{BCD}$

Jadi $\mathbf{683_{10} = 0110\ 1000\ 0011_{BCD}}$

2. Konversikan bilangan BCD $1001\ 0101\ 0111_{BCD}$ ke nilai desimalnya.

Penyelesaian: 1001 0101 0111

9 5 7 = 957_{10}

Jadi $\mathbf{1001\ 0101\ 0111_{BCD} = 957_{10}}$

Tabel dibawah ini merupakan perbandingan sistem bilangan yang biasanya digunakan dalam sistem komputer dan elektronika digital.

Tabel 1.5 kode BCD

Desimal	Biner	Oktal	Heksadesimal	BCD
0	0000 0000	00	00	0000 0000
1	0000 0001	01	01	0000 0001
2	0000 0010	02	02	0000 0010
3	0000 0011	03	03	0000 0011
4	0000 0100	04	04	0000 0100
5	0000 0101	05	05	0000 0101
6	0000 0110	06	06	0000 0110
7	0000 0111	07	07	0000 0111
8	0000 1000	10	08	0000 1000
9	0000 1001	11	09	0000 1001

10	0000 1010	12	0A	0001 0000
11	0000 1011	13	0B	0001 0001
12	0000 1100	14	0C	0001 0010
13	0000 1101	15	0D	0001 0011
14	0000 1110	16	0E	0001 0100
15	0000 1111	17	0F	0001 0101
16	0001 0000	20	10	0001 0110
17	0001 0001	21	11	0001 0111
18	0001 0010	22	12	0001 1000
19	0001 0011	23	13	0001 1001
20	0001 0100	24	14	10 0000

4.2 **Kode Gray**

Dalam kode Gray, setengah bagian atas, yaitu untuk kode desimal 5-9, merupakan bayangan cermin dari pada setengah bagian bawah, yaitu kode untuk desimal 0-4, kecuali untuk bit 3 (bit ke 4 dari kanan). Sifat ini disebut reflective. Di samping itu, seperti dapat dilihat pada Tabel 4.2 di depan, kode Gray juga mempunyai sifat bahwa kode untuk desimal yang berturutan berbeda hanya pada 1 bit. Sifat ini sangat penting dalam pengubahan sinyal-sinyal mekanis atau listrik ke bentuk digital. Sebagai contoh, kalau tegangan yang dikenakan pada suatu voltmeter digital berubah dari 3 volt ke 4 volt (dalam biner dari 0011 ke 0100), maka ada kemungkinan bit 2 (bit ke 3 dari kanan) akan berubah lebih dulu dari bit-bit yang lain sehingga akan memberikan penunjukan sementara 0111 (= 7) yang jelas salah. Dengan penggunaan kode Gray kesalahan seperti ini tidak akan terjadi.

Kode Gray biasanya digunakan sebagai data yang menunjukkan posisi dari suatu poros mesin yang berputar

Cara mengubah bilangan desimal ke kode Gray:

Contoh : Ubah bilangan desimal 13 ke kode Gray !

13			Desimal	
+	+	+		abaikan bawaannya
1	1	0	1	
1	0	1	1	kode Gray

4.3 **Kode ASCII**

Untuk mendapatkan informasi keluar masuknya data di komputer, dibutuhkan informasi seluruh alamat huruf dan simbol yang digunakan untuk pemrosesan data selain perwakilan dari bilangan tersebut. Informasi ini berupa nama, alamat, dan keterangan yang harus dimasukan dan dikeluarkan pada format pembacaan di sistem komputer. Oleh karena itu, dibutuhkan suatu kode khusus untuk mewakili semua data alfanumeris (huruf, simbol dan bilangan). Kode tersebut disebut juga kode ASCII (*American Standard Code for Information Interchange*), dinyatakan dalam bit biner. Selain angka dan huruf, kode ini juga menampung karakter pengendali seperti EOF (*End of File*) sebagai tanda akhir file dan EOL (*End of Line*) sebagai tanda akhir baris. Kode ini merupakan kode yang paling banyak digunakan untuk pertukaran informasi. Tujuh bit kode ASCII akan menghasilkan 128 kode kombinasi yang berbeda. Menggunakan tabel ASCII, dapat diperoleh kode ASCII huruf “P” yaitu 0101 0000.

Tabel. 1.6 Kode Alfanumerik ASCII, EBCDIC, dan Hollerith

Tanda	ASCII	EBCDI	Kartu	Tanda	ASCII	EBCDI	Kartu
NUL	00	00	12,0,9,8,1	@	40	7C	8,4
SOH	01	01	12, 9, 1	A	41	C1	12,1
STX	02	02	12, 9, 2	B	42	C2	12,2
ETX	03	03	12, 9, 3	C	43	C3	12,3
BOT	04	37	9,7	D	44	C4	12,4
ENQ	05	2D	0, 9,8,5	E	45	C5	12,5
ACK	06	2E	0, 9,8,6	F	46	C6	12,6
BEL	07	2F	0,9,8,7	G	47	C7	12,7
BS	08	16	11,9,4	H	48	C8	12,8
HT	09	05	11,9,5	I	49	C9	12,9
LF	0A	25	0,9,5	J	4A	D1	11,1
VT	0B	0B	12,9,8,3	K	4B	D2	11,2
FF	0C	0C	12,9,8,4	L	4C	D3	11,3
CR	0D	0D	12,9,8,5	M	4D	D4	11,4
S0	0E	0E	12,9,8,6	N	4E	D5	11,5
S1	0F	0F	12,9,8,7	O	4F	6	11,6
DLE	10	10	12,11,9,8,1	P	50	D7	11,7
DC1	11	11	11,9,1	Q	51	D8	11,8
DC2	12	12	11,9,2	R	52	D9	11,9
DC3	13	13	11,9,3	S	53	E2	0,2
DC4	14	35	9,8,4	T	54	E3	0,3
NAK	15	3D	9,8,5	U	55	E4	0,4
SYN	16	32	9,2	V	56	E5	0,5
ETB	17	26	0,9,6	W	57	E6	0,6
CAN	18	18	11,9,8	X	58	E7	0,7
EM	19	19	11,9,8,1	Y	59	E8	0,8

SUB	1A	3F	9,8,7	Z	5A	E9	0,9
ESC	1B	24	0,9,7	[	5B	AD	12,8,2
FS	1C	1C	11,9,8,4	\	5C	15	0,8,2
GS	1D	1D	11,9,8,5	]	5D	DD	11,8,2
RS	1E	1E	11,9,8,6	^	5E	5F	11,8,7
US	1F	1F	11,9,8,7	_	5F	6D	0,8,5
blank	20	40	no punch	‘	60	14	8,1
!	21	5A	12,8,7	A	61	81	12,0,1
“	22	7F	8,7	B	62	82	12,0,2
#	23	7B	8,3	C	63	83	12,0,3
$	24	5B	11,8,3	D	64	84	12,0,4
%	25	6C	0,8,4	E	65	85	12,0,5
&	26	50	12	F	66	86	12,0,6
‘	27	7D	8,5	G	67	87	12,0,7
(	28	4D	12,8,5	H	68	88	12,0,8
)	29	5D	11,8,5	I	69	89	12,0,9
*	2A	5C	11,8,4	J	6A	91	12,11,1
+	2B	4E	12,8,6	K	6B	92	12,11,2
,	2C	6B	0,8,3	L	6C	93	12,11,3
–	2D	60	11	M	6D	94	12,11,4
.	2E	4B	12,8,3	N	6E	95	12,11,5
/	2F	61	0,1	O	6F	96	12,11,6
0	30	F0	0	P	70	97	12,11,7
1	31	F1	1	Q	71	98	12,11,8
2	32	F2	2	R	72	99	12,11,9
3	33	F3	3	S	73	A2	11,0,2
4	34	F4	4	T	74	A3	11,0,3
5	35	F5	5	U	75	A4	11,0,4

6	36	F6	6	V	76	A5	11,0,5
7	37	F7	7	W	77	A6	11,0,6
8	38	F8	8	X	78	A7	11,0,7
9	39	F9	9	Y	79	A8	11,0,8
:	3A	7A	8,2	Z	7A	A9	11,0,9
;	3B	5E	11,8,6	(	7B	8B	12,0
<	3C	4C	12,8,4	|	7C	4F	12,11
=	3D	7E	8,6	)	7D	9B	11,0
>	3E	6E	0,8,6	~	7E	4A	11,0,1
?	3F	6F	0,8,7	DEL	7F	07	12,9,7

4.4 **Kode Excess-3 (XS-3)**

Excess-3 artinya kelebihan tiga. Sesuai dengan namanya, penetapannyadiperoleh dari penambahan 3 pada nilai binernya. Tabel 1.7.berikut inimenunjukkan kode XS-3.

Tabel 1.7. Kode Excess-3

Desimal	Kode Excess-3
0	0011
1	0100
2	0101
3	0110
4	0111
5	1000
6	1001
7	1010
8	1011
9	1100

Seperti halnya dengan kode BCD, kode XS-3 ini hanya menggunakansepuluh dari enam belas kombinasi yang ada. Enam kelompok bit yang tidak dipakai adalah 0000, 0001, 0010, 1101, 1110, dan 1111

Contoh:

1. Kodekan bilangan decimal 129 ke system XS-3.

Jawab :

```
   1     2     9
  0001  0010  1001    Setara binernya
+ 0011  0011  0011    Tambah tiga
  0100  0101  1100
```

Jadi, $129_{(10)} = 010001011100_{(XS3)}$

2. Kembalikan kode XS-3 0110 1001 1100 1000 menjadi bilangandesimal.

Jawab :

```
  0110  1001  1100  1000
  6     9     12    8     Setara desimalnya
- 3     3     3     3     Dikurang tiga
  3     7     9     5
```

Jadi, $0110\,1001\,1100\,1000_{(XS-3)} = 3795_{(10)}$

Kode XS-3 ini dirancang untuk mengatasi kesulitan kode BCD dalam perhitungan aritmatika. Penjumlahan dengan menggunakan kode XS3 dapat dilakukan dengan mengikuti aturan berikut :

1. Penjumlahan mengikuti aturan penjumlahan biner biasa
2. a. Jika hasil penjumlahan untuk suatu kelompok menghasilkan suatu simpanan desimal, tambahkan 0011 ke kelompok tersebut.

b. Jika hasil penjumlahan untuk setiap kelompok tidak menghasilkan simpanan desimal, kurangkan 0011 dari kelompok tersebut.

Contoh 1:

1. Jumlahkan bilangan decimal 63 dengan 26 dengan menggunakan system penjumlahan kode XS-3.

Jawab :

63 → 1001 0110

26+ → 0101 1001

+89 → 1110 1111 penjumlahan biner biasa

-0011 0011–

1011 1100

Penjumlahan contoh di atas tidak mempunyai simpanan decimal. Untuk proses penjumlahan yang mempunyai simpanan desimal dapat dilihat pada contoh 2.

Contoh 2:

2. Jumlahkan bilangan decimal 38 dengan 29 dengan menggunakan system penjumlahan kode XS-3.

Jawab :

38 → 0110 1011

29+ → 0101 1100

+67 → 1100 0111 penjumlahan biner biasa

-0011 0011+

1001 1010

Latihan Soal

Konversikan bilangan dibawah ini:

1. Ubahlah bilangan berikut ini kedalam bentuk Desimal
   A. 3A (16) = … (10)
   B. 377 (8) = … (10)
   C. 101010 (2) = … (10)
2. Ubahlah bilangan berikut ini kedalam bentuk Hexa Desimal
   A. 25 (10) = … (16)
   B. 276 (8) = … (16)
   C. 1010001 (2)= … (16)
3. Ubahlah bilangan berikut ini kedalam bentuk Biner
   A. BC (16) = … (2)
   B. 169 (10) = … (2)
   C. 654 (8) = … (2)
4. Ubahlah bilangan berikut ini kedalam bentuk Oktal
   A. 1FA (16) = … (8)
   B. 654 (10) = … (8)
   C. 1001100 (2)= … (8)
5. Ubahlah bilangan berikut ini kedalam bentuk Biner, Oktal, Desimal Atau Hexa
   A. 1111011_2 = ... $_8$ = ... $_{10}$ = ... $_{16}$
   B. 3563_8 = ... $_2$ = ... $_{10}$ = ... $_{16}$
   C. 7865_{10} = ... $_2$ = ... $_8$ = ... $_{16}$
   D. $1ACF_{16}$ = ... $_2$ = ... $_8$ = ... $_{10}$
   E. $1AFD_{16}$ = ... $_2$ = ... $_8$ = ... $_{10}$
6. Jumlahkan bilangan biner dibawah ini :
   A. $1011011_2 + 100011_2$ = ...
   B. $111011_2 + 100011_2 + 101010_2$ = ...
7. Jumlahkan bilangan Oktal dibawah ini :
   A. $2341_8 + 1456_8$ = ...
   B. $2241_8 + 1416_8 + 2351_8$ = ...

8. Jumlahkan bilangan Hexa dibawah ini :
   A. $134F_{16} + 241B_{16}$ = ……
   B. $334F_{16} + 261B_{16} + 33F1_{16}$ = ……
9. Jumlahkan bilangan Biner, Oktal, Desimal atau Hexa dibawah ini :
   A. $101010_2 + 1347_8$ = …… 10
   B. $3347_8 + 2619_{10}$ = …… 10
   C. $A34D_{16} + 2619_{10}$ = …… 10
   D.
10. Kalikan bilangan biner, Oktal, Desimal dan Hexa dibawah ini :
   A. $10101_2 \times 10101_2$ = …… 2
   B. $321_8 \times 265_8$ = …… 8
   C. $1F_{16} \times 65_{16}$ = …… 16

BAB 2
GERBANG LOGIKA DASAR

2.1 Gerbang Logika Dasar

Gerbang Logika atau dalam bahasa Inggris disebut dengan *Logic Gate* adalah dasar pembentuk Sistem Elektronika Digital yang berfungsi untuk mengubah satu atau beberapa Input (masukan) menjadi sebuah sinyal Output (Keluaran) Logis. Gerbang Logika beroperasi berdasarkan sistem bilangan biner yaitu bilangan yang hanya memiliki 2 kode simbol yakni **0** dan **1** dengan menggunakan Teori Aljabar Boolean. Gerbang Logika yang diterapkan dalam Sistem Elektronika Digital pada dasarnya menggunakan Komponen-komponen Elektronika seperti Integrated Circuit (IC), Dioda, Transistor, Relay, Optik maupun Elemen Mekanikal.

2.2 Jenis-jenis Gerbang Logika Dasar dan Simbolnya

Terdapat 7 jenis Gerbang Logika Dasar yang membentuk sebuah Sistem Elektronika Digital, yaitu :

1. Gerbang AND
2. Gerbang OR
3. Gerbang NOT
4. Gerbang NAND
5. Gerbang NOR
6. Gerbang X-OR (Exclusive OR)
7. Gerbang X-NOR (Exlusive NOR)

Tabel yang berisikan kombinasi-kombinasi Variabel Input (Masukan) yang menghasilkan Output (Keluaran) Logis disebut dengan **"Tabel Kebenaran"** atau **"Truth Table".** Input dan Output pada Gerbang

Logika hanya memiliki 2 level. Kedua Level tersebut pada umumnya dapat dilambangkan dengan :

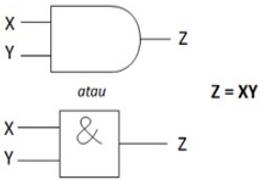
a) HIGH (tinggi) dan LOW (rendah)
b) TRUE (benar) dan FALSE (salah)
c) ON (Hidup) dan OFF (Mati)
d) 1 dan 0

Contoh Penerapannya ke dalam Rangkaian Elektronika yang memakai Transistor TTL (Transistor-transistor Logic), maka 0V dalam Rangkaian akan diasumsikan sebagai "LOW" atau "0" sedangkan 5V akan diasumsikan sebagai "HIGH" atau "1". Berikut ini adalah Penjelasan singkat mengenai 7 jenis Gerbang Logika Dasar beserta Simbol dan Tabel Kebenarannya.

1. Gerbang AND (AND Gate)

Gerbang AND memerlukan 2 atau lebih Masukan (Input) untuk menghasilkan hanya 1 Keluaran (Output). Gerbang AND akan menghasilkan Keluaran (Output) Logika 1 jika semua masukan (Input) bernilai Logika 1 dan akan menghasilkan Keluaran (Output) Logika 0 jika salah satu dari masukan (Input) bernilai Logika 0. Simbol yang menandakan Operasi Gerbang Logika AND adalah tanda titik (".") atau tidak memakai tanda sama sekali.
Contohnya : Z = X.Y atau Z = XY.

Simbol dan Tabel Kebenaran Gerbang AND (AND Gate)



Gambar 2.1 Simbol Gerbang AND

Tabel 2.1 Tabel Kebenaran Gerbang Logika AND

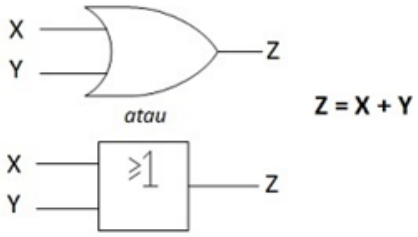
INPUT		OUTPUT
X	Y	Z
0	0	0
0	1	0
1	0	0
1	1	1

2. Gerbang OR (OR Gate)

Gerbang OR memerlukan 2 atau lebih Masukan (Input) untuk menghasilkan hanya 1 Keluaran (Output). Gerbang OR akan menghasilkan Keluaran (Output) 1 jika salah satu dari Masukan (Input) bernilai Logika 1 dan jika ingin menghasilkan Keluaran (Output) Logika 0, maka semua Masukan (Input) harus bernilai Logika 0.

Simbol yang menandakan Operasi Logika OR adalah tanda Plus (“+”). Contohnya : Z = X + Y.

Simbol dan Tabel Kebenaran Gerbang OR (OR Gate)



Gambar 2.2 Simbol Gerbang OR

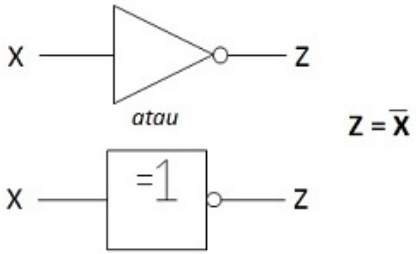
Tabel 2.2 Tabel Kebenaran Gerbang Logika OR

INPUT		OUTPUT
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	1

3. Gerbang NOT (NOT Gate)

Gerbang NOT hanya memerlukan sebuah Masukan (Input) untuk menghasilkan hanya 1 Keluaran (Output). Gerbang NOT disebut juga dengan Inverter (Pembalik) karena menghasilkan Keluaran (Output) yang berlawanan (kebalikan) dengan Masukan atau Inputnya. Berarti jika kita ingin mendapatkan Keluaran (Output) dengan nilai Logika 0 maka Input atau Masukannya harus bernilai Logika 1. Gerbang NOT biasanya dilambangkan dengan simbol minus ("-") di atas Variabel Inputnya.

Simbol dan Tabel Kebenaran Gerbang NOT (NOT Gate)



Gambar 2.3 Simbol Gerbang NOT

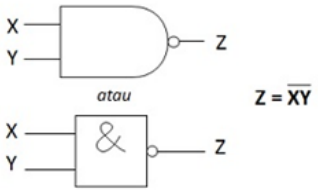
Tabel 2.3 Tabel Kebenaran Gerbang Logika NOT

INPUT	OUTPUT
X	Z
0	1
1	0

4. Gerbang NAND (NAND Gate)

Arti NAND adalah NOT AND atau BUKAN AND, Gerbang NAND merupakan kombinasi dari Gerbang AND dan Gerbang NOT yang menghasilkan kebalikan dari Keluaran (Output) Gerbang AND. Gerbang NAND akan menghasilkan Keluaran Logika 0 apabila semua Masukan (Input) pada Logika 1 dan jika terdapat sebuah Input yang bernilai Logika 0 maka akan menghasilkan Keluaran (Output) Logika 1.

Simbol dan Tabel Kebenaran Gerbang NAND (NAND Gate)



Gambar 2.4 Simbol Gerbang NAND

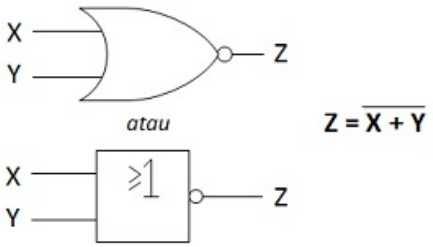
Tabel 2.4 Tabel Kebenaran Gerbang Logika NAND

INPUT		OUTPUT
X	Y	Z
0	0	1
0	1	1
1	0	1
1	1	0

5. Gerbang NOR (NOR Gate)

Arti NOR adalah NOT OR atau BUKAN OR, Gerbang NOR merupakan kombinasi dari Gerbang OR dan Gerbang NOT yang menghasilkan kebalikan dari Keluaran (Output) Gerbang OR. Gerbang NOR akan menghasilkan Keluaran Logika 0 jika salah satu dari Masukan (Input) bernilai Logika 1 dan jika ingin mendapatkan Keluaran Logika 1, maka semua Masukan (Input) harus bernilai Logika 0.

Simbol dan Tabel Kebenaran Gerbang NOR (NOR Gate)



Gambar 2.5 Simbol Gerbang NOR

Tabel 2.5 Tabel Kebenaran Gerbang Logika NOR

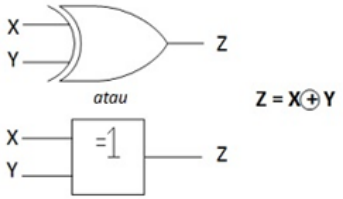
INPUT		OUTPUT
X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	0

6. Gerbang X-OR (X-OR Gate)

X-OR adalah singkatan dari Exclusive OR yang terdiri dari 2 Masukan (Input) dan 1 Keluaran (Output) Logika. Gerbang X-OR akan

menghasilkan Keluaran (Output) Logika 1 jika semua Masukan-masukannya (Input) mempunyai nilai Logika yang berbeda. Jika nilai Logika Inputnya sama, maka akan memberikan hasil Keluaran Logika 0.

Simbol &Tabel Kebenaran Gerbang X-OR (X-ORGate)



Gambar 2.6 Simbol Gerbang X-OR

Tabel 2.6 Tabel Kebenaran Gerbang Logika X-OR

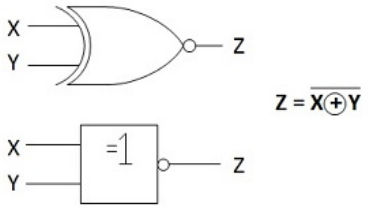
INPUT		OUTPUT
X	Y	Z
0	0	0
0	1	1
1	0	1
1	1	0

7. Gerbang X-NOR (X-NOR Gate)

Seperti Gerbang X-OR, Gerban X-NOR juga terdiri dari 2 Masukan (Input) dan 1 Keluaran (Output). X-NOR adalah singkatan dari Exclusive NOR dan merupakan kombinasi dari Gerbang X-OR dan Gerbang NOT. Gerbang X-NOR akan menghasilkan Keluaran (Output) Logika 1 jika semua Masukan atau Inputnya bernilai Logika yang sama dan akan menghasilkan Keluaran (Output) Logika 0 jika semua

Masukan atau Inputnya bernilai Logika yang berbeda. Hal ini merupakan kebalikan dari Gerbang X-OR (Exclusive OR).

Simbol dan Tabel Kebenaran Gerbang X-NOR (X-NOR Gate)



Gambar 2.7 Simbol Gerbang X-NOR

Tabel 2.7 Tabel Kebenaran Gerbang Logika X-NOR

INPUT		OUTPUT
X	Y	Z
0	0	1
0	1	0
1	0	0
1	1	1

2.3 Rangkaian Terintegrasi

Rangkaian Terintegrasi (Integrated Circuit/IC), sering juga disebut sirkuit terpadu, terdiri dari beberapa transistor, resistor, dll yang terinterkoneksi satu sama lain dalam package (paket) kecil dengan terminal-terminal sambungan. IC itu sudah lengkap, hanya memerlukan sambungan input dan output dan sebuah tegangan supply untuk bisa berfungsi. Cara lain, beberapa komponen eksternal harus dihubungkan agar IC itu bisa beroperasional.

Gambar 2.8 Rangkaian Terintegrasi

IC dapat diklasifikasikan menurut fungsinya, sebagai IC analog atau IC digital. IC analog (juga disebut IC linier) bisa berupa amplifier, voltage regulator, dll. IC digital biasanya berisi transistor-transistor baik yang dalam keadaan swicthed-off atau dalam keadaan switched-on. Gerbang-gerbang logika, counting circuits, dan yang semacam terdapat dalam bentuk IC. Metoda lain mengklasifikasi IC adalah menurut ukurannya small-scale integration (SSI), medium-scale integration (MSI), large-scale integration (LSI), dan very large-scale integration (VLSI). Selang ini dari kurang 12 rangkaian individual per paket sampai di atas 50.000 rangkaian dalam sebuah paket tunggal. Teknik-teknik yang digunakan dalam manufaktur IC lebih dari satu metoda klasifikasi. Teknik manufaktur yang utama adalah fabrikasi monolitik, thin-film, thick-film, dan hybrid. IC Monolitik Di dalam IC Monolitik semua komponen difabrikasi dengan proses difusi pada chip silikon tunggal. Interkoneksi di antara komponen dilakukan di atas permukaan struktur, dan pengawatan sambungan eksternal dihubungkan

dengan terminal. Rangkaian Terintegrasi (integrated circuit/IC) adalah realisasi secara fisik dari komponen-komponen diskrit yang terpisah tapi merupakan satu kesatuan yang berada di atas atau di dalam suatu substrat yang membentuk sebuah rangkaian terintegrasi yang bekerja dengan fungsi khusus. Bahan dasar sebuah substrat adalah semikonduktor kristal tunggal yang dipotong-potong menjadi beberapa keping wafer. Ukuran sekeping wafer mempunyai tebal 0,2 mm dan diameter 2 cm sampai 12 cm. Di atas keping wafer ini kemudian dibuat rangkaian-rangkaian yang diinginkan. Sekeping wafer dibagi menjadi sejumlah chip yang berukuran 10 mm x 10 mm. Chip-chip ini selanjutnya dirakit menjadi sebuah package (kemasan). Implementasi rangkaian logika ke dalam wafer silikon merupakan seni tersendiri. Pemahaman tentang langkah-langkah pengolahan silikon sangat diperlukan agar diperoleh keyakinan dalam mendisain pola rangkaian. Ada aturan disain (design rule) agar sesuai dengan toleransi peralatan proses fabrikasi. Disainer rangkaian terintegrasi menggambar pola berdasarkan aturan itu. Menggambar pola rangkaian dapat dilakukan secara manual atau dibantu komputer. Tujuannya agar disainer dapat menggunakan fasilitas proses fabrikasi dengan baik dalam merealisasikan rangkaian terintegrasi. Divais dapat direalisasikan menjadi rangkaian terintegrasi dengan beberapa teknologi, antara lain teknologi bipolar dan teknologi MOS. Teknologi bipolar mempunyai keterbatasan untuk rangkaian yang padat. Teknologi MOS berkembang untuk rangkaian terintegrasi padat seperti VLSI. Langkah pengolahan dasar yang dpakai untuk fabrikasi beberapa divais silikon, seperti dioda, transistor, dan IC, dapat dikategorikan sebagai berikut.

1. Ion implantation
2. Diffusion
3. Oxidation
4. Photolithography

5. Chemical-vapor depostion (termasuk epitaxy)
6. Metalization

Berawal dengan wafer silikon kristal tunggal, pengolahan yang tercantum di atas tadi dapat dipakai untuk menghasilkan divais diskrit yang berfungsi (yaitu, dioda dan transistor individual) dan IC. Divais atau IC ini dalam bentuk wafer, dengan puluhan, ratusan, atau bahkan ribuan divais atau IC pada wafer silikon yang sama. Wafer itu kemudian harus dipotong-potong untuk mendapatkan dice atau chip. Chip ini kemudian dijadikan kapsul (encapsulated) atau dikemas (packaged), dengan bermacam-macam kemasan dengan metoda pengemasan yang ada. Ada tiga tujuan dasar pengemasan.

1. membuat kapsul pada chip untuk melindungi chip dari pengaruh lingkungan,
2. memberikan kemudahan akses ke beberapa bagian dari chip melalui struktur pin sedemikian rupa sehingga divais dapat dengan mudah ditancapkan (plug) pada atau dihubungkan pada bagian yang lain dari suatu sistem, dan
3. memberi fasilitas heat transfer untuk keluar dari divais ke udara.

Proses fabrikasi dasar yang tercantum di atas tadi biasanya diaplikasikan berkali-kali secara berturut-turut, terutama dalam kasus IC, dimana sebanyak 20 pengulangan dari langkah-langkah fotolitografi, oksidasi, implantasi ion, dan difusi yang bisa dilakukan.

Persiapan Wafer Silikon

Material awal untuk pengolahan divais silikon adalah wafer silikon kristal tunggal yang jenis dan doping-nya yang sesuai kondukvitasnya. Urutan persiapan wafer silikon terdiri dari langkah-langkah dasar berikut ini.

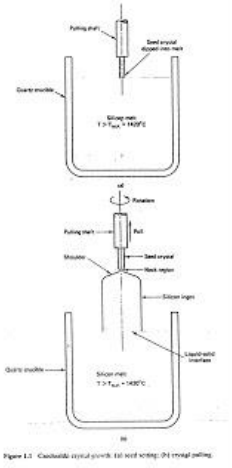
1. Pertumbuhan kristal dan doping

2. Ingot trimming dan grinding
3. Ingot slicing
4. Wafer polishing dan etching
5. Wafer cleaning

Pertumbuhan Kristal

Dalam proses pertumbuhan kristal, ingot silikon kristal tunggal dengan tingkat dan jenis doping yang cukup untuk diproduksi. Material awal untuk pertumbuhan kristal adalah silikon polikristalin yang tingkat kemurniannya tinggi yang disebut semicoductor grade silicon. Tingkat kemurnian dari material ini dalam selang lebih dari 99,9999999 atau kermurnian 9-nines. Ini terkait dengan suatu konsentrasi impuriti sebesar lebih dari 1 part per billion atoms (>1 ppba), yaitu kurang dari satu atom impuriti untuk setiap miliar (10^9) atom silikon. Banyaknya atom per satuan isi adalah 5.0×10^{22} cm^{-3}, sehingga konsentrasi impuriti sebesar 1 ppba berkait dengan kerapatan sebesar 5×10^{13} cm^{-3}. Kebanyakan dari kerapatan impuritas residual ini merupakan impuritas akseptor seperti boron, dan resistivitas yang terkait dengan kerapaatan impuritas di atas tadi hampir 300 ohm-cm. Polyscrystaline silicon dengaan kosentrasi impuritas kurang dari 0,1 ppba itu ada. Proses pertumbuhan kristal Czochralski merupakan proses yang paling sering digunakan untuk memproduksi ingot silikon kristal tunggal. Polycrystalline silicon bersama dengan sejumlah dopant yang cukup atau silikon yang didoping dimasukkan ke dalam sebuah quartz crucible, yang kemdian dimasukkan sebuah tungku pertumbuhan kristal. Bahan itu kemudian dipanasi sampai pada suhu sedikit di atas titik cair/leleh silikon yaitu 1420^oC. Sebuah batang kecil silikon kristal tunggal yang disebut sebuah seed crystal kemudian dicelupkan ke dalam cairan/lelehan silikon dan perlahan-lahan ditarik, seperti pada gambar di bawah ini. Konduksi panas pada seed crystal itu akan

menurunkan suhu lelehan silikon yang terhubung dengan seed crystal itu sedikit di bawah titik leleh silikon.



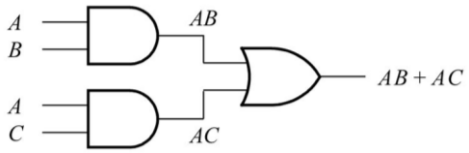
Gambar 2.9 Proses pertumbuhan kristal Czochralski

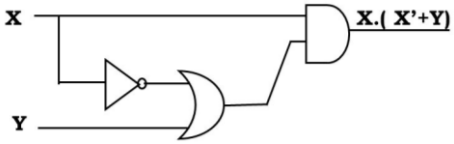
Contoh :

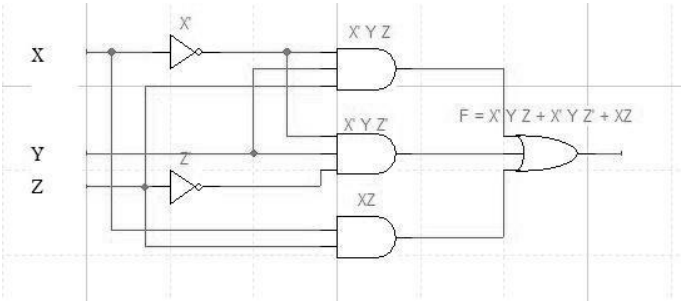
Buatlah rangkaian logika dari

1. F = AB + AC
2. F = X.(X'+Y)
3. F = X'YZ + X'YZ' + XZ

Jawab

1. 

2. 

3. 

Soal Latihan :

Buatlah rangkaian logika dari fungsi dibawah ini :

1. F = A.B.C + AC'
2. F = X.(X'.Z + Y')
3. F = X'.Y.Z + X'.Y.Z' + X.Z' + X.Y.Z'

BAB 3
RANGKAIAN KOMBINASIONAL

Rangkaian kombinasional terdiri dari gerbang logika yang memiliki output yang selalu tergantung pada kombinasi input yang ada. Rangkaian kombinasional melakukan operasi yang dapat ditentukan secara logika dengan memakai sebuah fungsi boolean.

3.1 Aljabar Boolean

Aljabar Boolean atau dalam bahasa Inggris disebut dengan Boolean Algebra adalah matematika yang digunakan untuk menganalisis dan menyederhanakan Gerbang Logika pada Rangkaian-rangkaian Digital Elektronika. Boolean pada dasarnya merupakan Tipe data yang hanya terdiri dari dua nilai yaitu “True” dan “False” atau “Tinggi” dan “Rendah” yang biasanya dilambangkan dengan angka “1” dan “0” pada Gerbang Logika ataupun bahasa pemrograman komputer. Aljabar Boolean ini pertama kali diperkenalkan oleh seorang Matematikawan yang berasal dari Inggris pada tahun 1854.

Dengan menggunakan Hukum Aljabar Boolean ini, kita dapat mengurangi dan menyederhanakan Ekspresi Boolean yang kompleks sehingga dapat mengurangi jumlah Gerbang Logika yang diperlukan dalam sebuah rangkaian Digital Elektronika.

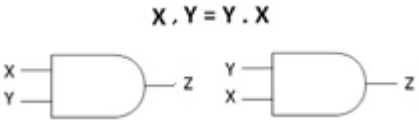
Berikut 6 tipe Hukum yang berkaitan dengan Hukum Aljabar Boolean.

1. Hukum Komutatif (Commutative Law) :
   Hukum Komutatif menyatakan bahwa penukaran urutan variabel atau sinyal Input tidak akan berpengaruh terhadap Output Rangkaian Logika.

Contoh :

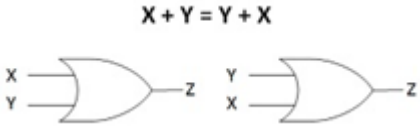
Perkalian (Gerbang Logika AND)

X.Y = Y.X



Gambar 3.1 Hukum Komutatif Perkalian (Gerbang logika AND)

Penjumlahan (Gerbang Logika OR)
X+Y = Y+X



Gambar 3.2 Hukum Komutatif Penjumlahan (Gerbang logika OR)

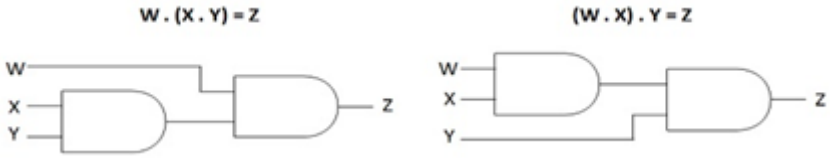
Catatan : Pada penjumlahan dan perkalian, kita dapat menukarkan posisi variabel atau dalam hal ini adalah sinyal Input, hasilnya akan tetap sama atau tidak akan mengubah keluarannya.

2. Hukum Asosiatif (Associative Law)

Hukum Asosiatif menyatakan bahwa urutan operasi logika tidak akan berpengaruh terhadap Output Rangkaian Logika.

Contoh :

- Perkalian (Gerbang Logika AND)
  W . (X . Y) = (W . X) . Y



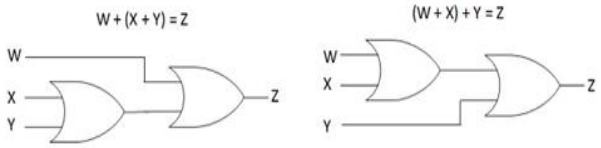
Gambar 3.3 Hukum Asosiatif Perkalian (Gerbang Logika AND)

Tabel 3.1 Hukum Asosiatif Perkalian (Gerbang Logika AND)

INPUT			OUTPUT
W	X	Y	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

INPUT			OUTPUT
W	X	Y	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

- Penjumlahan (Gerbang Logika OR)
  W + (X + Y) = (W + X) + Y



Gambar 3.4 Hukum Asosiatif Penjumlahan (Gerbang Logika OR)

Tabel 3.2 Hukum Asosiatif Perkalian (Gerbang Logika AND)

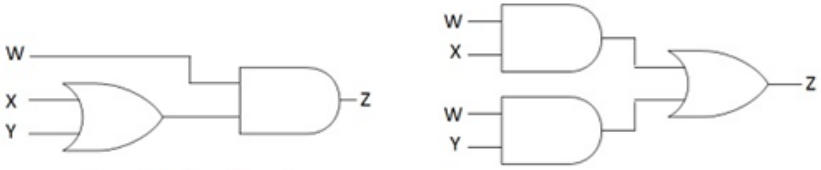
INPUT			OUTPUT
W	X	Y	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

INPUT			OUTPUT
W	X	Y	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

3. Hukum Distributif

Hukum Distributif menyatakan bahwa variabel-variabel atau sinyal Input dapat disebarkan tempatnya atau diubah urutan sinyalnya, perubahan tersebut tidak akan mempengaruhi Output Keluarannya.

$$W \times (X + Y) = WX + WY$$



Gambar 3.5 Hukum Distributif

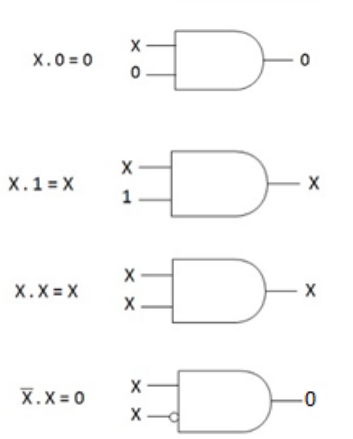
Tabel 3.2 Hukum Distributif

INPUT			OUTPUT
W	X	Y	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

INPUT			OUTPUT
W	X	Y	Z
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

4. Hukum AND (AND Law)

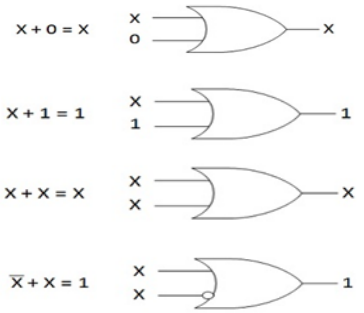
Disebut dengan Hukum AND karena pada hukum ini menggunakan Operasi Logika AND atau perkalian. Berikut ini contohnya.



Gambar 3.6 Hukum AND

5. Hukum OR (OR Law)

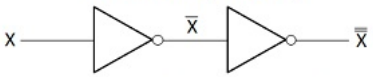
Hukum OR menggunakn Operasi Logika OR atau Penjumlahan. Berikut ini adalah Contohnya :



Gambar 3.7 Hukum OR

6. Hukum Inversi (Inversion Law)

Hukum Inversi menggunakan Operasi Logika NOT. Hukum Inversi ini menyatakan jika terjadi Inversi ganda (kebalikan 2 kali) maka hasilnya akan kembali ke nilai aslinya.



Gambar 3.8 Hukum Invers

Jadi, jika suatu Input (masukan) diinversi (dibalik) maka hasilnya akan berlawanan. Namun jika diinversi sekali lagi, hasilnya akan kembali ke semula.

Contoh. Diketahui fungsi Booelan *f*(*x*, *y*, *z*) = *xy z*', nyatakan fungsi Boolean dalam tabel kebenaran.

Penyelesaian:

X	*Y*	*Z*	*f*(*x*, *y*, *z*) = *xy z*'
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0

3.2. Teorema Demorgan

Teorema ini ada 2 yaitu :

$$\overline{(x + y)} = \overline{x} \cdot \overline{y} \longrightarrow \text{Teori 1}$$
$$\overline{(x \cdot y)} = \overline{x} + \overline{y} \longrightarrow \text{Teori 2}$$

Teori 1 diatas menyatakan bahwa penjumlahan dua variable (OR) yang diinvers, maka hasil nya sama dengan setiap variable diinvers dan di AND kan. Teori 2 diatas menyatakan bahwa penjumlahan dua variable (AND) yang diinvers, maka hasil nya sama dengan setiap variable diinvers dan di OR kan.

Contoh :

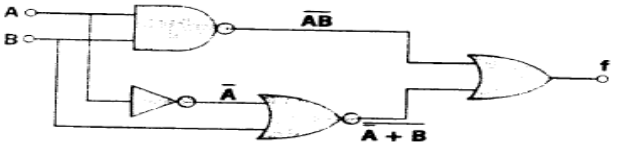
Example 1

$$z = \overline{A + \overline{B} \cdot C}$$
$$= \overline{A} \cdot (\overline{\overline{B} \cdot C})$$
$$= \overline{A} \cdot (\overline{\overline{B}} + \overline{C})$$
$$= \overline{A} \cdot (B + \overline{C})$$

Example 2

$$\omega = \overline{(A + BC) \cdot (D + EF)}$$
$$= (\overline{A + BC}) + (\overline{D + EF})$$
$$= (\overline{A} \cdot \overline{BC}) + (\overline{D} \cdot \overline{EF})$$
$$= [\overline{A} \cdot (\overline{B} + \overline{C})] + [\overline{D} \cdot (\overline{E} + \overline{F})]$$
$$= \overline{A}\overline{B} + \overline{A}\overline{C} + \overline{D}\overline{E} + \overline{D}\overline{F}$$

Contoh :



Sub output dicatat pada diagram. Fungsi keseluruhan dapat disederhanakan sebagai berikut.

F $= \overline{A\bar{B}} + \overline{\bar{A} + B}$

$= (\bar{A} + \bar{B}) + A\bar{B}$ (DeMorgan's rule)

$= \bar{A} + \bar{B}(1 + A)$ (Distribution)

$= \bar{A} + \bar{B}$ $(1 + 4 = 1)$

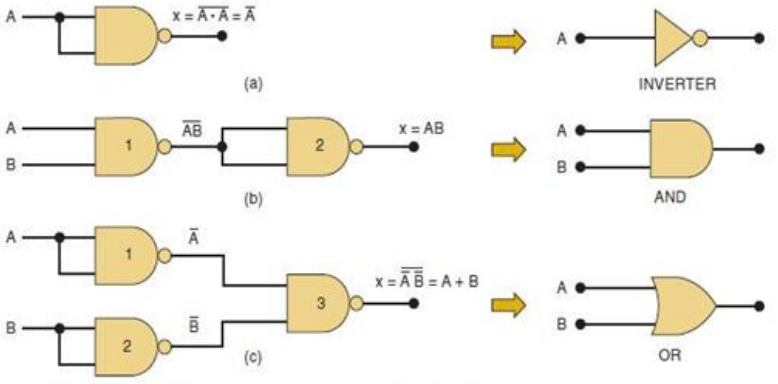
$= \overline{A\bar{B}}$ (DeMorgan's rule)

A	B	$\overline{A\bar{B}}$	$\overline{\bar{A} + B}$	F
0	0	1	0	1
0	1	1	0	1
1	0	1	1	1
1	1	0	0	0

3.3 Universality dari gerbang Nand dan gerbang Nor

Gerbang Nand

Semua persamaan Boolean terdiri dari berbagai kombinasi dari operasi dasar OR, AND dan INVERT. Setiap persamaan dapat diimplementasikan menggunakan gerbang OR, gerbang AND dan INVERTER. Sehingga memungkinkan untuk mengimplementasikan setiap persamaan logika menggunakan NAND gate, karena Gerbang NAND merupakan kombinasi yang dapat digunakan dalam operasi Boolean OR, AND dan INVERT.



Gambar 3.9 Gerbang Nand dapat digunakan untuk Implentasi Fungsi Boolean

Gambar 3.9 (a).

Gerbang NAND memiliki 2 input dimana kedua inputnya berasal dari variable A. dalam hal ini Gerbang NAND berindak sebagai INVERTER karena outputnya.

$$x = \overline{A \cdot A} = \overline{A}.$$

Gambar 3.9 (b).

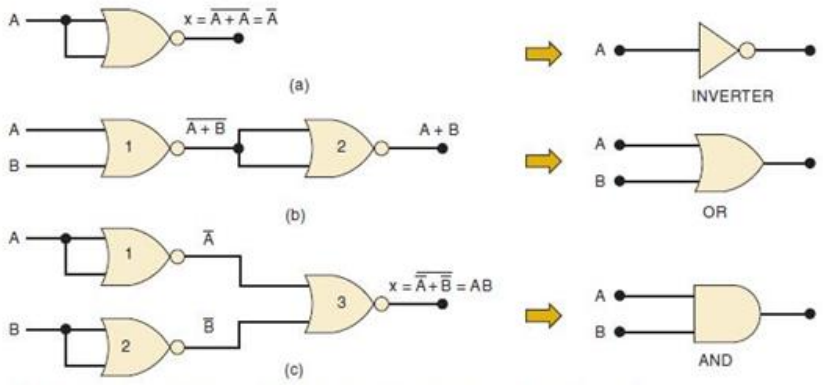
Dua Gerbang NAND yang terhubung sehingga menghasilkan operasi AND. Gerbang NAND kedua berfungsi sebagai INVERTER untuk merubah seperti fungsi AND yang diinginkan.

Gambar 3.9 (c).

Operasi OR dapat diimplementasikan menggunakan gerbang NAND yang dihubungkan seperti Gambar diatas. Gerbang NAND 1 dan 2 digunakan sebagaiINVERTER untuk membalik input, sehingga hasil akhir output $\overline{\overline{x}} \equiv A \,.\, B$ dapat disederhanakan menjadi $x = A + B$ menggunakan teori demorgan.

Gerbang NOR

Seperti halnya Gerbang NAND, dapat digunakan gerbang NOR.



Gambar 3.10 Gerbang Nor dapat digunakan untuk Implentasi Fungsi Boolean

Gambar 3.10 (a).

Gerbang NOR memiliki 2 input dimana kedua inputnya berasal dari variable A. dalam hal ini Gerbang NOR betrindak sebagai INVERTER karena outputnya $x = A \;.\; A = A$

Gambar 3.10 (b).

2 Gerbang NOR yang terhubung sehingga menghasilkan operasi OR. Gerbang NOR kedua berfungsi sebagai INVERTER untuk merubah seperti fungsi OR yang diinginkan.

Gambar 3.10 (c).

Operasi AND dapat diimplementasikan menggunakan gerbang NOR yang dihubungkan seperti Gambar diatas. Gerbang NOR 1 dan 2 digunakan sebagai INVERTER untuk membalik input, sehingga hasil akhir output $\overline{\overline{x} = A}\,.\,B$ dapat disederhanakan menjadi $x = A\,.\,B$ menggunakan teori demorgan.

Soal Latihan :

Diketahui fungsi Booelan *f*(*x, y, z*) = *x.y' z + x'*. y. z + x.y' + y.z , nyatakan fungsi Boolean dalam tabel kebenaran dan gambar rangkain logika.

BAB 4
PETA KARNAUGH

4.1 K-Map (Karnaugh Map)

Karnaugh Map atau yang biasanya disebut dengan K-Map adalah suatu teknik penyederhanaan fungsi logika dengan cara pemetaan. K-Map terdiri dari kotak-kotak yang jumlahnya terdiri dari jumlah variable dan fungsi logika atau jumlah inputan dari rangkaian logika yang sedang kita hitung. Rumus untuk menentukan jumlah kotak pada K-Map adalah 2^n n adalah banyaknya variabel / inputan.

Langkah – langkah pemetaan K-Map secara umum :

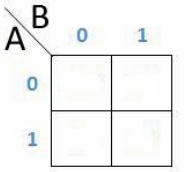
1. Menyusun aljabar Boolean terlebih dahulu
2. Menggambar rangkaian digital
3. Membuat Table Kebenarannya
4. Merumuskan Tabel Kebenarannya
5. Lalu memasukkan rumus Tabel Kebenaran ke K-Map (Kotak-kotak)

4.2 Penyederhanaa K-Map

1. Penyederhanaan Dua Variabel

Catatan : Bar = ‘

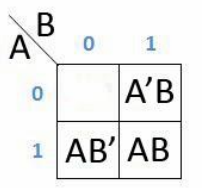
Tabel dari K-Map 2 variabel adalah seperti dibawah ini.



Contoh :

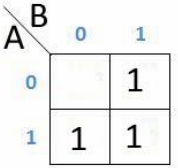
H = AB + A’B+AB’

Maka cara pengerjaanya seperti dibawah ini.

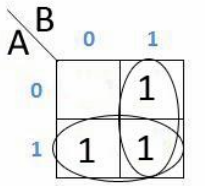


Bar / ‘ biasanya ditulis kedalam angka 0 sedangkan angka 1 adalah tanpa Bar / ‘

Dan dapat dipermudah lagi menjadi dibawah ini.



Yang dapat disederhanakan dalam K-Map hanya 2 / kelipatan 2 dari kotak yang berdempetan dan sedangkan jika seperti kotak diatas maka penyderhanaannya menjadi :



Yaitu terletak pada kotak 01 + 11 dan 10 + 11 yaitu cara penyederhanaan dengan cara menulis angka yang sama (1 lingkaran) dan menerjemahkannya kedalam bentuk huruf seperti A dan B.

Caranya :

01

11

1 yang sama adalah angka 1 yang dibelakang jadi jika letaknya dibelakang (kedua) adalah B (B diambil dari tabel K-Map Diatas) jika yang sama angka 0 pada urutan kedua adalah B’ diatas sudah disebutkan bahwa angka 0 = Bar/’

10

11

1 yang sama adalah angka 1 yang didepan jadi jika letaknya didepan (pertama) adalah A (A diambil dari tabel K-Map Diatas) jika yang sama angka 0 pada urutan kedua adalah A' diatas sudah disebutkan bahwa angka 0 = Bar/'

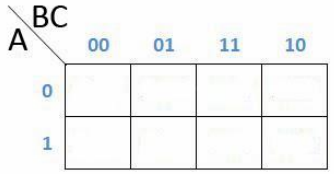
Jadi kesimpulan dari contoh diatas adalah dari rumus :

H = AB + A'B + AB' dapat disederhanakan menggunakan K-Map menjadi BA / AB (boleh dibalik menurut abjad tetapi harus 1 teman atau tidak dapat dibalik dengan huruf yang dipisahkan dengan penjumlahan atau pengurangan)

2. Penyederhanaan Tiga Variabel

Catatan : Bar = '

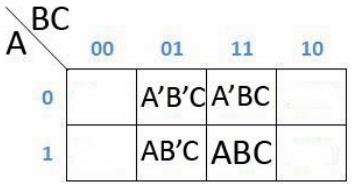
Tabel dari K-Map 3 variabel adalah seperti dibawah ini.



Contoh :

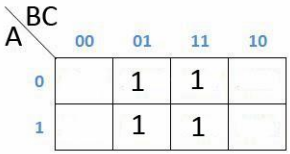
H = ABC + A'BC+A'B'C+AB'C

Maka cara pengerjaanya seperti dibawah ini.

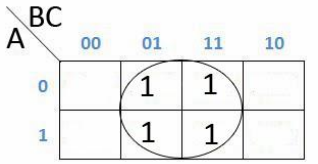


Bar / ‘ biasanya ditulis kedalam angka 0 sedangkan angka 1 adalah tanpa Bar / ‘

Dan dapat dipermudah lagi menjadi dibawah ini.



Yang dapat disederhanakan dalam K-Map hanya 2 / kelipatan 2 dari kotak yang berdempetan dan sedangkan jika seperti kotak diatas maka penyderhanaannya menjadi :



Cara diatas adalah langsung mesederhanakan 4 kotak, sebenarnya dapat disederhanakan menjadi 2 kotak 2 kotak tetapi terlalu lama dan kita hanya menyingkat waktu saja menjadi 4 kotak langsung, terletak pada kotak 001 + 011+101 +111 yaitu cara penyederhanaan dengan cara menulis angka yang sama (1 lingkaran) dan menerjemahkannya kedalam bentuk huruf seperti A, B, C.

Caranya :

011

011

101

111

1 yang sama adalah angka 1 yang dibelakang jadi jika letaknya dibelakang (keempat) adalah C (C diambil dari tabel K-Map Diatas). Jika yang sama angka 0 pada urutan keempat adalah C’ diatas sudah disebutkan bahwa angka 0 = Bar/’

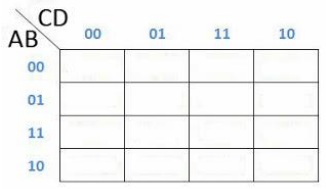
Jadi kesimpulan dari contoh diatas adalah dari rumus :

H = ABC + A’BC+A’B’C+AB’C dapat disederhanakan menggunakan K-Map menjadi C.

3. Penyederhanaan 4 variabel

Catatan : Bar = ‘

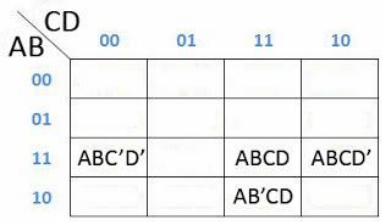
Tabel dari K-Map 4 variabel adalah seperti dibawah ini



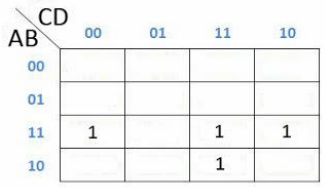
Contoh :

H = ABCD + ABCD’+AB’CD+ABC’D’

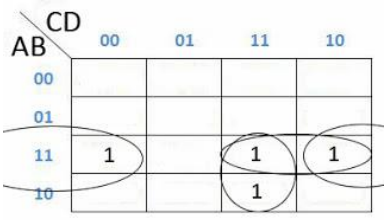
Maka cara pengerjaanya seperti dibawah ini.



Bar / ‘ biasanya ditulis kedalam angka 0 sedangkan angka 1 adalah tanpa Bar / ‘ Dan dapat dipermudah lagi menjadi dibawah ini :



Yang dapat disederhanakan dalam K-Map hanya 2 / kelipatan 2 dari kotak yang berdempetan dan sedangkan jika seperti kotak diatas maka penyderhanaannya menjadi :



Yaitu terletak pada kotak 1111 + 1011 dan 1111 + 1110 dan 1110 + 1100. Cara diatas menyederhanakannya dapat dari sisi paling kanan dengan sisi paling kiri dalam 1 baris.

Cara penyederhanaan dengan cara menulis angka yang sama (1 lingkaran) dan menerjemahkannya kedalam bentuk huruf seperti A, B, C, D.

Caranya :

1111

1011

1 11 yang sama adalah angka 1 yang pertama, ketiga, dan keempat adalah A, C, dan D (A, C, dan D diambil dari tabel K-Map Diatas) jika yang sama angka 0 pada urutan kedua adalah A' dst diatas sudah disebutkan bahwa angka 0 = Bar/'

1111

1110

111 yang sama adalah angka 1 yang pertama, kedua, dan ketiga adalah A, B, C (A, B, C diambil dari tabel K-Map Diatas) jika yang sama angka 0 pada urutan kedua adalah A' dst diatas sudah disebutkan bahwa angka 0 = Bar/'

1110

1100

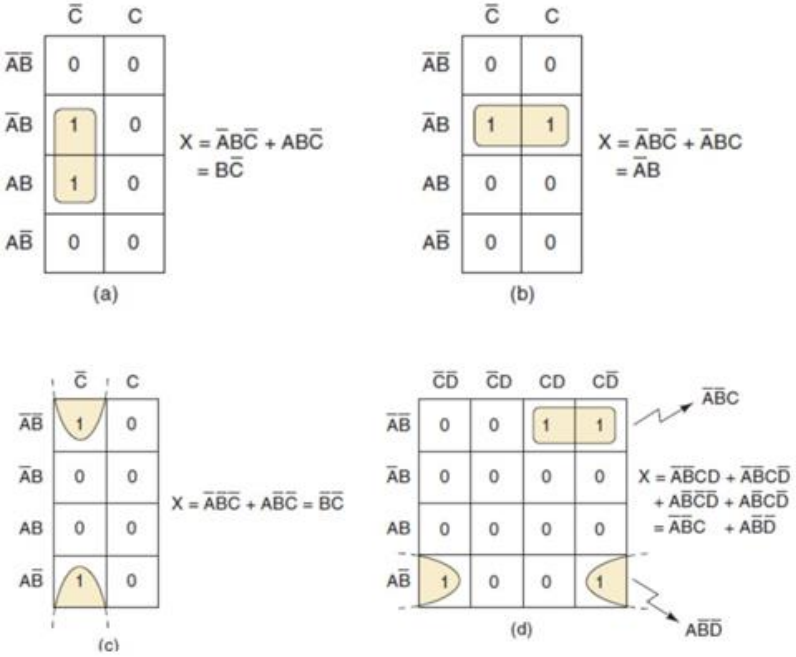
11 yang sama adalah angka 1 yang pertama dan kedua adalah A dan B (A dan B diambil dari tabel K-Map Diatas) jika yang sama angka 0 pada urutan kedua adalah A' dst diatas sudah disebutkan bahwa angka 0 = Bar/'

Jadi kesimpulan dari contoh diatas adalah dari rumus :

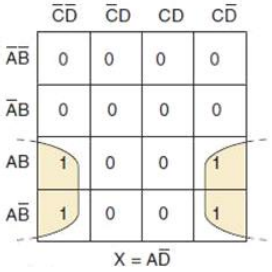
H = ABCD + ABCD'+AB'CD+ABC'D' dapat disederhanakan menggunakan K-Map menjadi H = ACD + ABC + ABC' (boleh dibalik menurut abjad tetapi harus 1 teman atau tidak dapat dibalik dengan huruf yang dipisahkan dengan penjumlahan atau pengurangan)

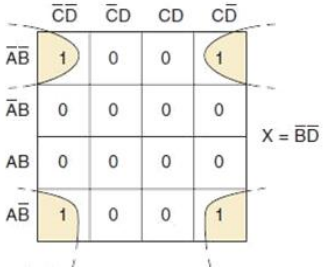
Persamaan output X dapat disederhanakan dengan cara membuat kotak pada Kmap untuk yang berisi 1. Proses ini disebut dengan **looping**

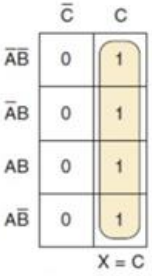
Looping → 2

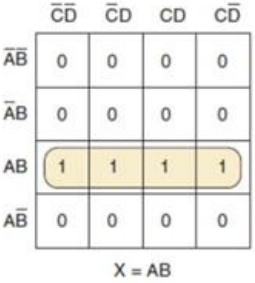


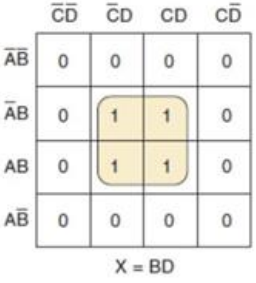
Looping → 4



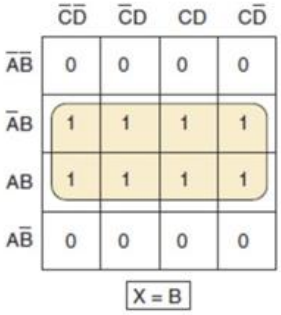


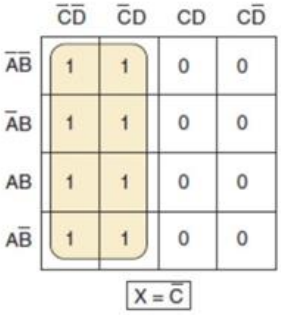


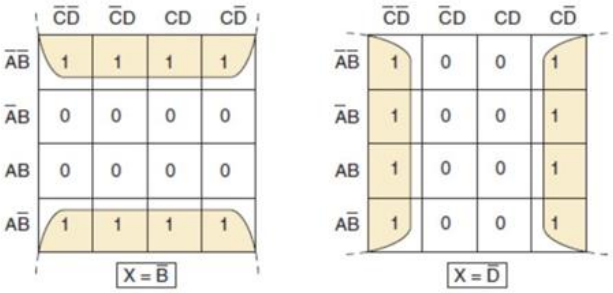




Looping → 8

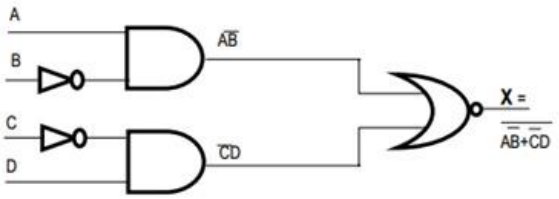






4.3 Sum of Product (SOP)

Persamaan bentuk SOP terbentuk dari dua atau lebih gerbang logika AND yang kemudian di OR kan di dalam tanda kurung, kemudian di dalam tanda kurung tersebut bisa jadi terdiri dari dua buah variabel maupun lebih. Secara sederhana dapat dijelaskan bahwa SOP merupakan bentuk persamaan yang menjalankan operasi OR terhadap AND. Berikut adalah contoh rangkaian persamaan SOP:

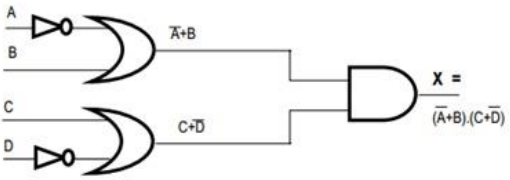


Gambar 4.1 rangkaian persamaan SOP

4.4 Product of Sum (POS)

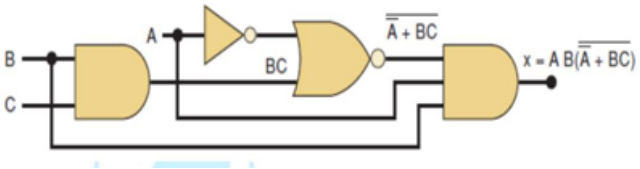
Persamaan bentuk Product of Sum (POS) terbentuk dari dua atau lebih gerbang logika OR yang kemudian diAND-kan, dalam

persamaan ini dapat berisi dua atau lebih variabel.Secara sederhana dapat dijelaskan bahwa POS merupakan bentuk persamaan yang menjalankan operasi AND terhadap keluaran-keluaran OR. Berikut adalah contoh Rangkaian Product of Sum (POS).

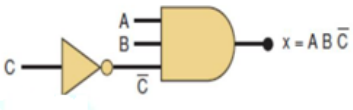


Gambar 4.2 rangkaian persamaan POS

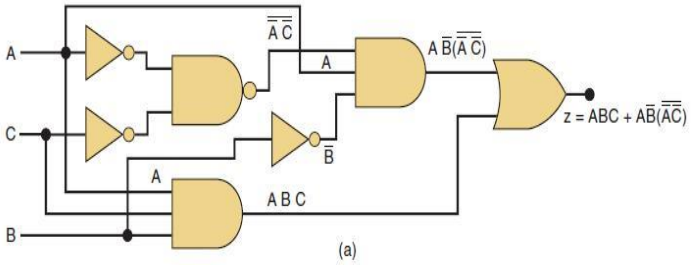
Contoh :



Hasil penyederhanaan :



Contoh:



Persamaan : $z = ABC + A\overline{B} \cdot \overline{(A\overline{C})}$

Langkah langkah :

Menggunakan teori demorgan → menghasilkan bentuk SOP

$$
\begin{aligned}
z &= ABC + A\overline{B}(\overline{\overline{A}} + \overline{\overline{C}}) && \text{[theorem (17)]}\\
&= ABC + A\overline{B}(A + C) && \text{[cancel double inversions]}\\
&= ABC + A\overline{B}A + A\overline{B}C && \text{[multiply out]}\\
&= ABC + A\overline{B} + A\overline{B}C && [A \cdot A = A]
\end{aligned}
$$

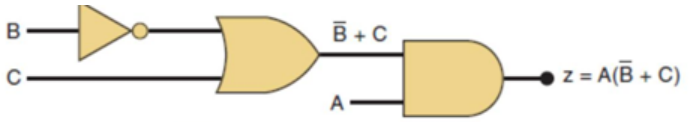
$$z = AC(B + \overline{B}) + A\overline{B}$$

Karena $B + B = 1$ maka

$$
\begin{aligned}
z &= AC(1) + A\overline{B}\\
&= AC + A\overline{B}
\end{aligned}
$$

$$z = A(C + \overline{B})$$

Rangkaian Sederhana menjadi :



Complete Design Procedure

Setiap masalah logika dapat diselesaikan dengan langkah langkah sebagai berikut:

1. Interpret the problem and set up a truth table to describe its operation.
2. Write the AND (product) term for each case where the output is 1.
3. Write the sum-of-products (SOP) expression for the output.
4. Simplify the output expression if possible.
5. Implement the circuit for the final, simplified expression.

Contoh :

Rancanglah sebuah rangkaian logika yang memiliki 3 input A, B dan C, dimana output akan menjadi HIGH hanya pada saat semua input mayoritas adalah HIGH.

Jawab :

1. Menterjemahkan masalah dan membuat table kebenaran.
   Output = 1 apabila mayoritas input adalah 1 (2/lebih input), dan yang lain output = 0

A	B	C	X
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

2. Membuat persamaan AND untuk output = 1

A	B	C	X	
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{A}BC$
1	0	0	0	
1	0	1	1	$A\overline{B}C$
1	1	0	1	$AB\overline{C}$
1	1	1	1	ABC

3. Menulis persamaan sum of products untuk output

$$x = \overline{A}BC + A\overline{B}C + AB\overline{C} + ABC$$

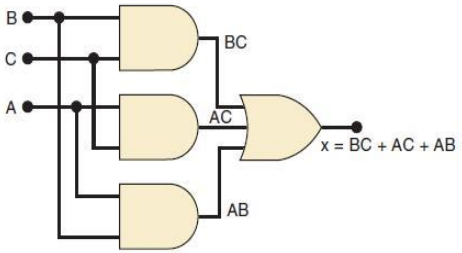
4. Menyederhanakan persamaan

$$x = \overline{A}BC + ABC + A\overline{B}C + ABC + AB\overline{C} + ABC$$

$$x = BC(\overline{A} + A) + AC(\overline{B} + B) + AB(\overline{C} + C)$$

$$x = BC + AC + AB$$

5. Implementasi rangkaian untuk persamaan terakhir

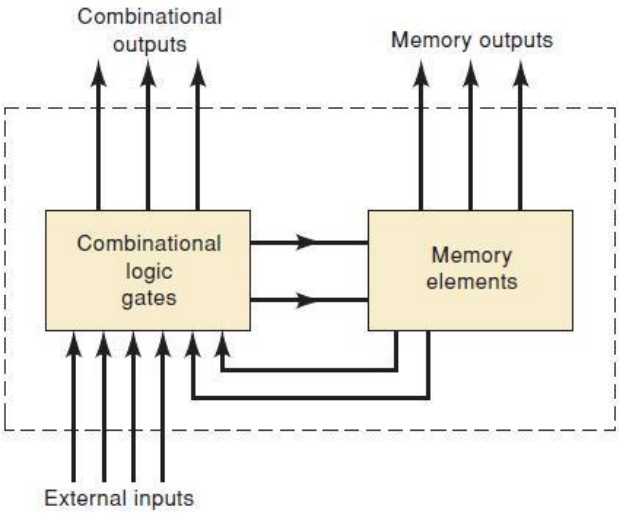


Soal Latihan :

Rancanglah sebuah rangkaian logika yang memiliki 3 input A, B dan C, dimana output akan menjadi HIGH hanya pada saat kedua input adalah HIGH.

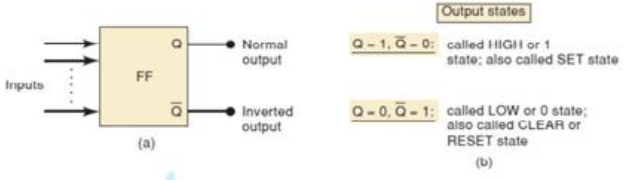
BAB 5

RANGKAIAN MULTIVIBRATOR



Gambar 5.1 Diagram sistem Digital

Elemen memory yang paling penting adalah flip flop, yang dibuat dari rangkaian gerbang gerbang gerbang logika. Walaupun sebuah gerbang logika tidak memiliki kemampuan penyimpanan, beberapa dapat dihubungkan sebagai cara untuk menyimpan informasi. Beberapa gerbang yang berbeda disusun dan digunakan untuk menghasilkan flip flop (FF).



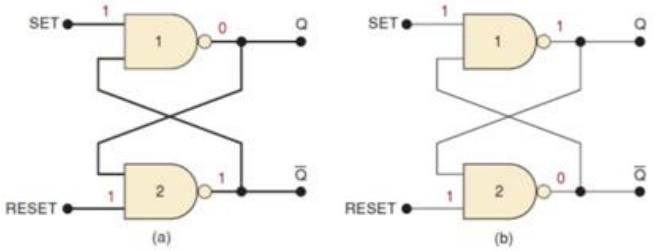
Gambar 5.2 General Flip Flop

Gambar 5.2 a, merupakan symbol yang digunakan oleh flip flop. Menunjukkan 2 input yang diberi label $\overline{Q}$ dan Q . Q/Q merupakan output FF. Q output disebut sebagai normal output FF dan Q merupakan invers output FF. Contoh Sebuah FF berada pada state HIGH (1), maka Q =1, jika FF pada state LOW (0) maka Q=0. Dan Q memiliki nilai sebaliknya. Gambar 6.2b merupakan 2 kemungkinan operasi pada FF. Untuk HIGH atau state 1(Q=1/$Q = 0$) disebut sebagai SET state. Pada saat input ke FF menyebabkan Q=1, maka disebut setting FF, sehingga FF di set. Dengan cara yang sama LOW atau 0 state (Q= 0/$Q = 1$ disebut sebagai CLEAR atau RESET state. Pada saat input ke FF menyebabkan Q=0, maka disebut clearing atau resetting FF, sehingga FF di reset. FF memiliki SET input dan/atau CLEAR (RESET) input yang digunakan untuk drive FF menjadi state output.

Nand Gate Latch

Rangkaian dasar FF dapat dibangun dari 2 gerbang NAND atau 2 gerbang NOR. Versi gerbang NAND disebut sebagai NAND gate latch atau secara sederhana latch (Gambar 5.3a). 2 gerbang NAND di cross-coupled sehingga output NAND-1 dihubungkan ke salah satu input dari NAND-2 dan sebaliknya. Output gerbang diberi label dan disebut dengan latch out put. Pada kondisi normal, output akan selalu di invers dengan yang lain. Ada 2 latch input : SET input yaitu input set Q sama dengan state 1; RESET input akan resets Q ke 0 state.

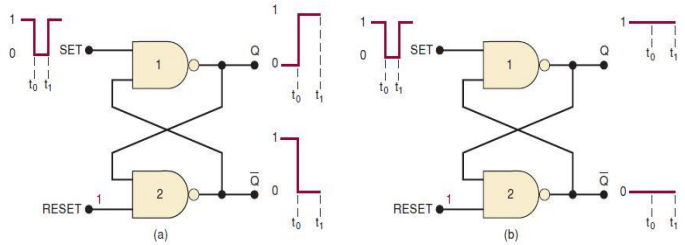
Versi kedua gerbang NAND (Gambar 5.3b). , dimana Q = 1 dan $\overline{Q} = 0$. HIGH dari gerbang NAND-1 menghasilkan LOW pada output NAND-2, dan output NAND-1 tetap HIGH. Ada 2 kemungkinan output yaitu SET = RESET = 1.



Gambar 5.3 NAND Latch

Setting the Latch (FF)

Berdasarkan Gambar 5.4 dibawah, memperlihatkan proses SET dan RESET dari Latch



Gambar 5.4 SET dan RESET latch

Keterangan gambar 5.4a.

SET terjadi pada saat :

Pulsa LOW pada waktu t0, Q menjadi HIGH, dan menjadi LOW dan NAND-1 memiliki 2 input LOW.

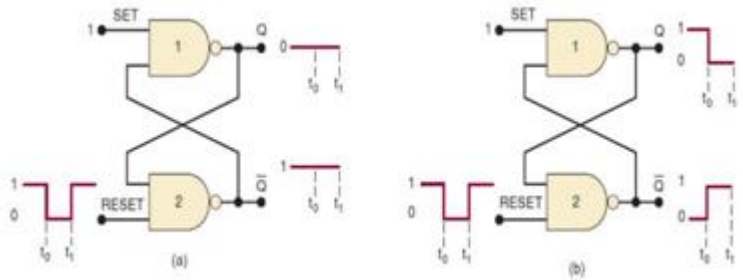
Pada saat SET kembali ke state 1 pada waktu t1, output NAND-1 menjadi HIGH, dan output NAND-2 tetap LOW.

Keterangan gambar 5.4b.

Jika$Q = \overline{0}$ maka output NAND-1 HIGH karena pulsa LOW pada SET tidak akan merubah apapun. Sehingga pada saat SET = HIGH, latch output tetap berada pada$Q$ = 1 dan$Q$ = 0

ReSetting the Latch (FF)

Jika input RESET = LOW dan SET = HIGH. Pada saat



Gambar 5.5 RESETTING LATCH

Tabel 5.1 Tabel Kebenaran

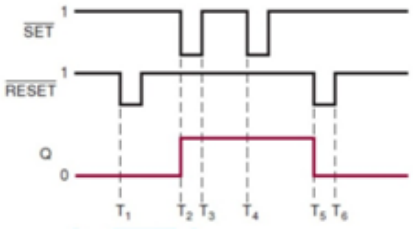
Set	Reset		Output
1	1		No Change
0	1		Q = 1
1	0		Q = 0
0	0		Invalid*

*Produces Q = Q' =1

1. SET = RESET = 1
   Kondisi ini normal state, dan tidak membuat perubahan apapun terhadap state output. akan tetap sama state nya dengan sebelumnya.
2. SET = 0, RESET = 1
   Menyebabkan output berubah ke state Q=1, disebut sebagai setting latch
3. SET = 1, RESET = 0
   Kondisi ini menghasilkan state0, dan disebut sebagai clearing/resetting latch
4. SET=RESET=0
   Kondisi ini mencoba untuk SET dan CLEAR latch pada waktu yang bersamaan, dan

menghasilkan =1. Jika input dikembalikan ke 1 secara bersamaan, maka hasilnya adalah tidak bisa diprediksi. Kondisi input ini tidak dipergunakan

Contoh :

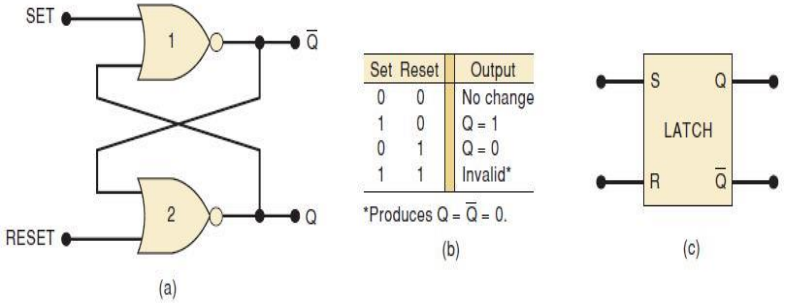


Gambar 5.6 Kondisi untuk SET dan CLEAR latch pada waktu yang bersamaan

NOR GATE LATCH

2 gerbang NOR yang di cross-coupled dapat digunakan sebagai NOR gate latch. Gambar

5.7(a) mirip dengan NAND latch kecuali output Q dan dengan posisi terbalik.



Gambar 5.7 NOR GATE LATCH

1. SET = RESET = 0

   Kondisi ini normal state, dan tidak membuat perubahan apapun terhadap state output.akan tetap sama state nya dengan sebelumnya.

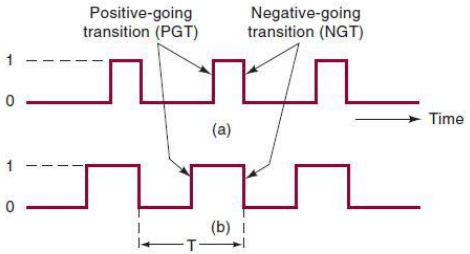
2. SET = 1, RESET = 0
   Menyebabkan output berubah ke state Q=1, disebut sebagai setting latch
3. SET = 0, RESET = 1
   Kondisi ini menghasilkan state0, dan disebut sebagai clearing/resetting latch
4. SET=RESET=1
   Kondisi ini mencoba untuk SET dan CLEAR latch pada waktu yang bersamaan, dan menghasilkan =0. Jika input dikembalikan ke 0 secara bersamaan, maka hasilnya adalah tidak bisa diprediksi. Kondisi input ini tidak dipergunakan.

Contoh :

Gambar 5.8 Kondisi untuk SET dan CLEAR latch pada waktu yang bersamaan

CLOCK SIGNALS AND CLOCKED FLIP-FLOPS

System digital dapat beroperasi secara asinkron atau sinkron. Pada system asinkron, output rangkaian logika dapat merubah state setiap saat satu atau lebih input berubah. Pada system sinkron, waktu setiap output dapat merubah state ditentukan oleh sebuah sinyal yang disebut dengan **clock.** Sinyal clock merupakan sebuah pulsa segiempat seperti pada Gambar 5.9 berikut.



Gambar 5.9 Sinyal Clock

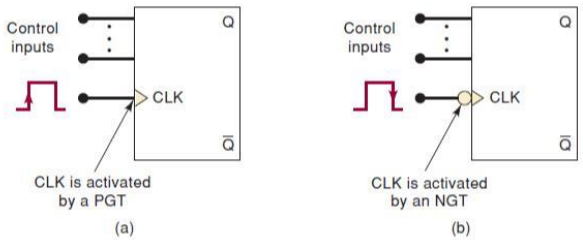
Sinyal clock didistribusikan ke semua bagian system, dan sebagian (if not all) output system dapat merubah state hanya pada saat clock membuat transisi. Transisi disebut sebagai *edges*.

Pada saat clock berubah dari 0 ke 1, maka disebut sebagai **positive-going transition(PGT)**; pada saat clock berubah dari 1 ke 0, maka disebut sebagai **negative-going transition (NGT)**.

Sebagiian system digital merupakan sinkron karena rangkaian sinkron lebih mudah untuk melakukan perancangan dan troubleshoot. Hal ini disebabkan rangkaian output dapat berubah hanya pada waktu spesifik atau disinkronkan dengan transisi clock-signal.

Sinkronisasi dari sinyal clock juga digunakan pada **clocked flip-flops** yang digunakan untuk merubah states pada satu atau lebih clock's transitions.

Clocked Flip-Flops



Gambar 5.10 Clock FF

Karakteristik Clocked FF

1. Clock FF memiliki sebuah clock input yang diberi label CLK, CK atau CP. Normalnya menggunakan CLK (Gambar 6.8a). Pada sebagian Clock FF, CLK input merupakan edge triggered, yang akan diaktifkan oleh sebuah sinyal transisi. Hal ini diindikasikan oleh adanya segitiga kecil pada CLK input.

   Sebuah FF dengan segitiga kecil pada CLK Input mengindikasikan bahwa input ini hanya diaktifkan pada saat positive going transition (PGT) terjadi, dan tidak ada bagian input pulse yang akan mempengerahu CLK Input. FF symbol memiliki sebuah bubble (Gambar 6.8b). seperti segitiga pada CLK Input. Yang berarti CLK input hanya diaktifkan pada saat Negative going Transition terjadi, dan juga tidak ada bagian input lainnya yang akan mempengaruhi CLK Input
2. Clock FFjuga memiliki satu atau lebih input control yang dapat memiliki beberapa nama, tergantung kepada operasinya. Control input tidak memiliki efek pada Q sampai aktif transisi clock terjadi.

   Contoh :

   Control input pada FF (Gambar 6.8a) tidak memiliki efek pada Q sampai PGT clock signal terjadi, sedangkan untuk Gambar 6.8b tidak memiliki efek sampai NGT clock signal terjadi.

5.1 FLIP - FLOP

Flip-flop adalah suatu rangkaian elektronika yang memiliki dua kondisi stabil dan dapat digunakan untuk menyimpan informasi. Flip Flop merupakan pengaplikasian gerbang logika yang bersifat Multivibrator Bistabil. Dikatakan Multibrator Bistabil karena kedua tingkat tegangan keluaran pada Multivibrator tersebut adalah stabil dan hanya akan mengubah situasi tingkat tegangan keluarannya saat

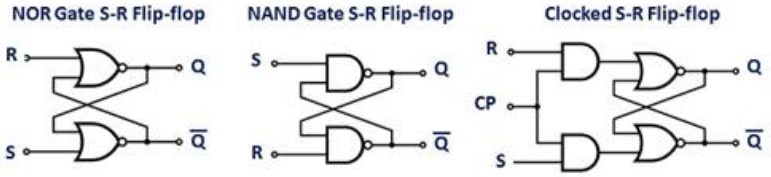
dipicu (trigger). Flip-flop mempunyai dua Output (Keluaran) yang salah satu outputnya merupakan komplemen Output yang lain.
Flip-flop Elektronik yang pertama kali ditemukan oleh dua orang ahli fisika Inggris William Eccles and F. W. Jordan pada tahun 1918 ini merupakan dasar dari penyimpan data memory pada komputer maupun Smartphone. Flip-flop juga dapat digunakan sebagai penghitung detak dan sebagai penyinkronsasian input sinyal waktu variabel untuk beberapa sinyal waktu referensi.

5.2 Jenis-jenis Flip-flop

Rangkaian Flip-flop pada umumnya dapat dibagi menjadi beberapa jenis, yaitu S-R Flip-flop, D Flip-flop, T Flip-flop dan JK Flip-flop.
Berikut dibawah ini adalah penjelasan singkatnya.

1. Set Reset (S-R) Flip-flop

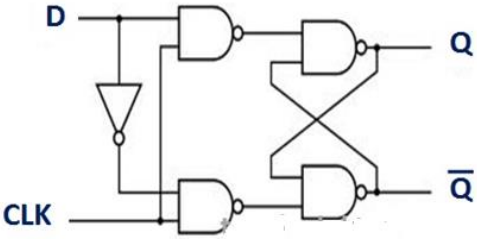
S-R adalah singkatan dari "Set" dan "Reset". Sesuai dengan namanya, S-R Flip-flop ini terdiri dari dua masukan (INPUT) yaitu S dan R. S-R Flip-flop ini juga terdapat dua Keluaran (OUTPUT) yaitu Q dan Q'. Rangkaian S-R Flip-flop ini umumnya terbuat dari 2 gerbang logika NOR ataupun 2 gerbang logika NAND. Ada juga S-R Flip-flop yang terbuat dari gabungan 2 gerbang Logika NOR dan NAND.
Berikut ini adalah diagram logika NOR Gate S-R Flip-flop, NAND Gate S-R Flip-Flop dan Clocked S-R Flip-flop (gabungan gerbang logika NOR dan NAND).



Gambar 5.11 Set Reset (SR) Flip-Flop

2. D Flip-flop

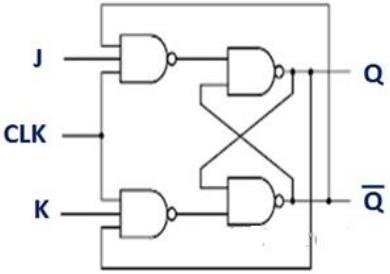
D Flip-flop pada dasarnya merupakan modifikasi dari S-R Flip-flip yaitu dengan menambahkan gerbang logika NOT (Inverter) dari Input S ke Input R. Berbeda dengan S-R Flip-flop, D Flip-flop hanya mempunyai satu Input yaitu Input atau Masukan D. Berikut ini diagram logika D Flip-flop.



Gambar 5.12 D Flip-Flop

3. J-K Flip Flop

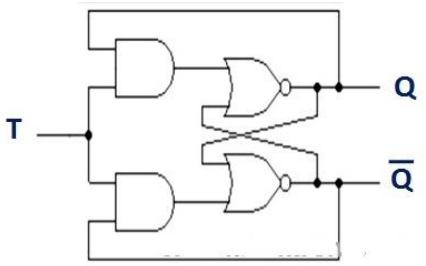
J-K Flip-flop juga merupakan pengembangan dari S-R Flip-flop dan paling banyak digunakan. J-K Flip-flop memiliki 3 terminal Input J, K dan CL (Clock). Berikut ini adalah diagram logika J-K Flip-flop.



Gambar 5.13 J-K Flip-Flop

4. T Flip-flip

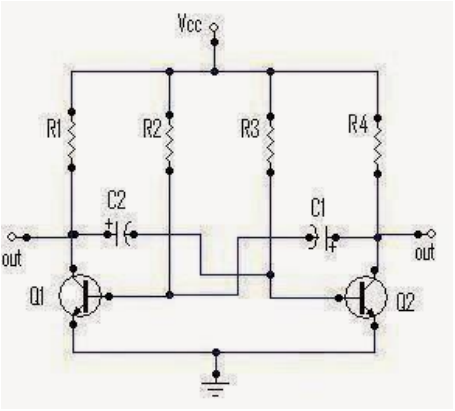
T Flip-flop merupakan bentuk sederhana dari J-K Flip-flop. Kedua Input J dan K dihubungkan sehingga sering disebut juga dengan Single J-K Flip-Flop. Berikut ini adalah diagram logika T flip-flop.



Gambar 5.14 T Flip-Flop

5.3 Rangkaian Multivibrator

Astable muiltivibrator adalah suatu bagian rangkaian yang bagian output nya pada satu keadaan atau level, akan tetapi berubah-ubah secara terus-menerus dari keadaan 0 ke keadaan 1 berulang-ulang. keadaan ini sering disebut keadaan semi stabil.

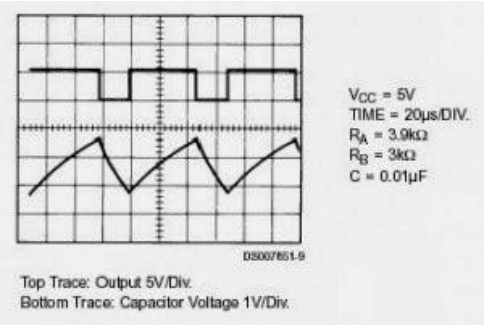


Gambar 5.15 Blok Diagram Dan Skema Rangkaian Multibrator

Cara Kerja Rangkaian Multibrator :

Cara kerja umum multivibrator adalah penguat transistor dua tingkat yang dihubungkan dengan kondensator dimana output dari tingkat yang terakhir dihubungkan dengan penguat pertama, sehingga kedua transistor itu akan saling umpan balik. Pulsa tegangan itu terjadi selama 1 periode yang ditentukan oleh komponen-komponen

penyusun rangkaian multivibrator tersebut. Rangkaian tersebuthanya mengubah keadaan tingkat tegangan keluarannya diantara 2 keadaan, masing-masing memiliki periode yang tetap.apabila pin 6 dan pin 2 dihubungkan (lihat blok diagram) maka akan memicu dirinya sendiri dan bergerak bebas sebagai multivibrator , rangkaian multivibrator tersebut akan bekerja secara bebas dan tidak lagi memerlukan pemicu.



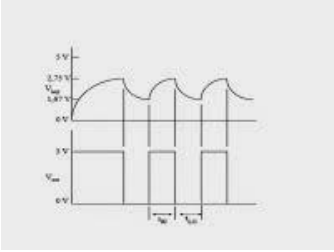
Gambar 5.16 kurva respon

Contoh :

Buatlah bentuk gelombang dari rangkaian multivibrator astabil Schmitt trigger berdasarkan rangkaian Scmitt Trigger yang mempunyai spesifikasi CMOS 74HC14 (VCC = 5 V).VOH = 5 V,VOL = 0 V VT+ = 2,75 V,VT-= 1,67 V

Jawab:

a. Bentuk gelombang darirangkaian Schmitt Trigger Multivibrator Astabil adalah:



2. Hitunglah waktu yang dibutuhkan saat pengisian tegangan kapasitor(tHI), pengosongan tegangan kapasitor(tLO), duty cycle dan rekuensi jika R = 10 KΩ dan C = 0,022 µF.

   Jawab :

   Untuk mencari tHI adalah:

   ΔV = VT+ −VT-

   ΔV = 2,75 V −1,67 V = 1,08 V

   E = 5 V −1,67 V = 3,33 V tHI= RC ln= (10 KΩ).(0,022 µF) ln= 86,2 µs

   Untuk mencari tLO adalah:

   ΔV = 2,75 V −1,67 V = 1,08 V

   E = 2,75 V −0 V = 2,75 V

   tLO= RCln= (10 KΩ).(0,022 µF) ln=110 µs

   Untuk mencari duty cycl (perbandingan antara lebar waktu saat kondisi high/tinggi dengan total perioda suatu gelombang) adalah:

   D = 0,439 = 43,9 %

   Untuk mencari frekuensi adalah:

   f = 5,10 KHz

BAB 6
RANGKAIAN ARITMETHIC

6.1 Arithmatic Logical Unit (ALU)

Arithmatic Logical Unit (ALU), adalah salah satu bagian/komponen dalam sistem di dalam sistem komputer yang berfungsi melakukan operasi/perhitungan aritmatika dan logika (Contoh operasi aritmatika adalah operasi penjumlahan dan pengurangan, sedangkan contoh operasi logika adalah logika AND dan OR. ALU bekerja besama-sama memori, di mana hasil dari perhitungan di dalam ALU di simpan ke dalam memori. Perhitungan dalam ALU menggunakan kode biner, yang merepresentasikan instruksi yang akan dieksekusi (opcode) dan data yang diolah (operand). ALU biasanya menggunakan sistem bilangan biner *two's complement*. ALU mendapat data dari register. Kemudian data tersebut diproses dan hasilnya akan disimpan dalam register tersendiri yaitu ALU output register, sebelum disimpan dalam memori. Pada saat sekarang ini sebuah chip/IC dapat mempunyai beberapa ALU sekaligus yang memungkinkan untuk melakukan kalkulasi secara paralel. Salah satu chip ALU yang sederhana (terdiri dari 1 buah ALU) adalah IC 74LS382/HC382ALU (TTL). IC ini terdiri dari 20 kaki dan beroperasi dengan 4×2 pin data input (pinA dan pinB) dengan 4 pin keluaran (pinF). Arithmatic Logical Unit (ALU), fungsi unit ini adalah untuk melakukan suatu proses data yang berbentuk angka dan logika, seperti data matematika dan statistika. ALU terdiri dari register-register untuk menyimpan informasi.Tugas utama dari ALU adalah melakukan perhitungan aritmatika (matematika) yang terjadi sesuai dengan instruksi program. Rangkaian pada ALU (Arithmetic and Logic Unit) yang digunakan untuk menjumlahkan bilangan dinamakan dengan Adder. Adder digunakan untuk memproses operasi aritmetika, Adder juga disebut rangkaian kombinasional aritmatika.

Ada 3 jenis adder:

1) Rangkaian Adder dengan menjumlahkan dua bit disebut Half Adder.
2) Rangkaian Adder dengan menjumlahkan tiga bit disebut Full Adder.
3) Rangkain Adder dengan menjumlahkan banyak bit disebut Paralel Adder

1. Half Adder

Rangkaian Half Adder merupakan dasar penjumlahan bilangan Biner yang terdiri dari satu bit, oleh karena itu dinamai Penjumlah Tak Lengkap.

1. jika A = 0 dan B = 0 dijumlahkan, hasilnya S (Sum) = 0.
2. jika A = 0 dan B = 1 dijumlahkan, hasilnya S (Sum) = 1.
3. jika A = 1 dan B = 1 dijumlahkan, hasilnya S (Sum) = 0
4. jika A = 1 dan B =1 dijumlahkan, hasilnya S (Sum) = 0. dengan nilai pindahan cy(Carry Out) = 1

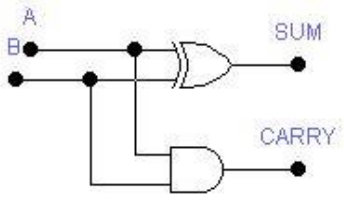
Dengan demikian, half adder memiliki 2 masukan (A dan B) dan dua keluaran (S dan Cy).

Tabel 6.1 Tabel Kebenaran Half Adder

A	B	S	Cy
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Dari tabel diatas, terlihat bahwa nilai logika dari Sum sama dengan nilai logika dari gerbang XOR, sedangkan nilai logika Cy sama dengan

gerbang logika AND. Dari tabel diatas, dapat dibuat rangkaian half adder seperti dibawah ini.



Gambar 6.1 Rangkaian Half Adder.

2. Full Adder

Full adder adalag mengolah data penjumlahan 3 bit bilangan atau lebih (bit tidak terbatas), oleh karena itu dinamakan rangkaian penjumlah lengkap. Perhatikan tabel dibawah ini.

Tabel 6.2 Tabel Kebenaran Full Adder

A	B	C	S	Cy
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

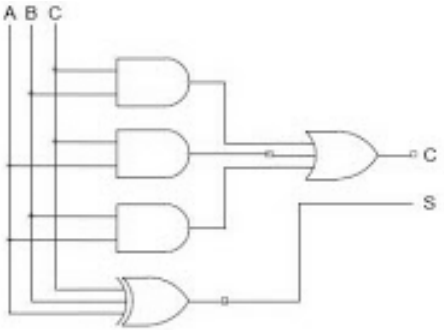
S = A’.B’.C + A’.B.C’ + A.B’.C’ + A.B.C

S = A⊕B⊕C

Cy = A’.B.C + A.B’.C + A.B.C’ + A.B.C

Dengan menggunakan peta Karnaugh, Cy dapat disederhanakan menjadi :

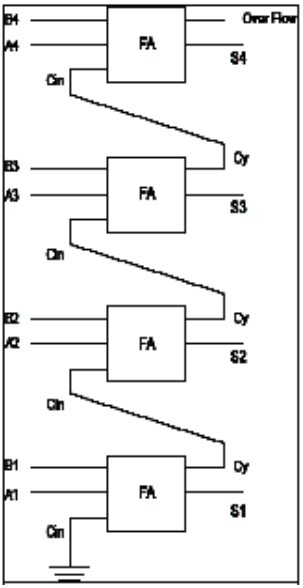
Cy = A.B + A.C + B.C



Gambar 6.2 rangkaian Full Adder

3. Paralel Adder

Paralel Adder adalah rangkaian Full Adder yang disusun secara paralel dan berfungsi untuk menjumlahkan bilangan biner berapa pun bitnya, tergantung jumlah Full Adder yang diparalelkan. Gambar dibawah ini menunjukan Paralel Adder yang terdiri dari 4 buah Full Adder yang disusun paralel sehingga membentuk sebuah penjumlahan 4 bit.



Gambar 6.3 rangkaian Paralel Adder

4. **Paralel Subtractor**

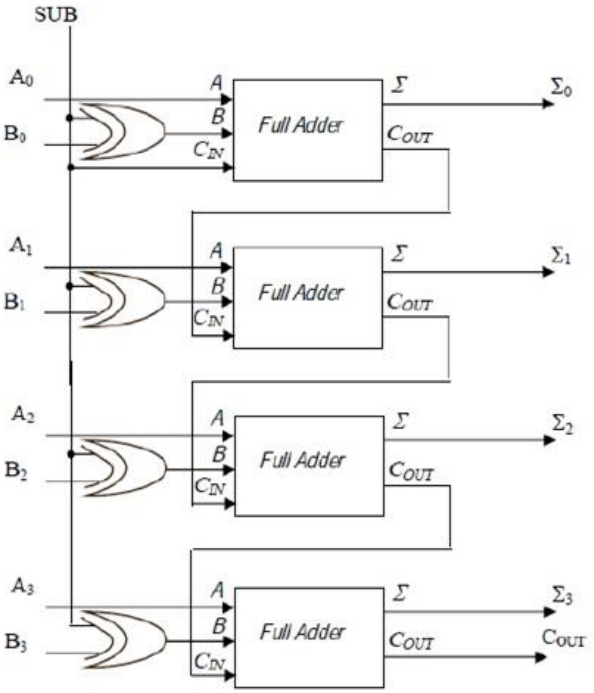
Merupakan rangkaian Parallel Adder yang dimodifikasi. Prinsip parallel subtractor adalah mengimplementasikan 2's complement yang cara kerjanya sama dengan parallel adder yaitu dengan menjumlahkan 2 bilangan, tetapi bedanya dari pengurangan bilangan dijadikan cara penjumlahan. Contoh : pengurangan 10 dengan 5 dirubah menjadi penjumlahan menjadi 10 dengan (-5). Prosesnya seperti dibawah ini.

3 A2 A1 A0

\- B3 B2 B1 B0 +

C out∑3∑2∑1∑0

Dimana : -B3 B2 B1 B0 artinya bilangan negatif dari B3 B2 B1 B0 yang di 2'S complement. Jadi kesimpulannya rangkaian parallel subtractor adalah rangkaian adder yang salah satuinputnya diubah menjadi negatif.



Gambar 6.4 Rangkaian Paralel Subtractor

BAB 7
CODER DAN MULTIPLEXER

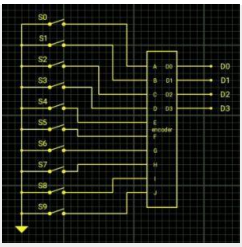
7.1 Encoder

Pengertian encoder dan cara kerja encoder. Encoder adalah suatu peralatan yang digunakan untuk mempersingkat jalur input yang awalnya berjumlah banyak menjadi output dengan jumlah yang lebih sedikit. Prinsip kerja encoder merupakan kebalikan dari decoder. Encoder dapat diartikan sebagai suatu peralatan yang digunakan untuk merubah kondisi input menjadi kondisi tertentu dan kondisi tersebut dapat dikembalikan lagi seperti semula dengan menggunakan rangkaian decoder. Encoder digunakan karna dengan encoder kita dapat menghemat jalur ataupun untuk menyesuaikan input supaya dapat diproses oleh rangkaian selanjutnya. Selengkapnya tentang pengertian dan cara kerja encoder adalah sebagai berikut :

Cara kerja encoder :

Ada banyak jenis encoder diantara adalah encoder 16 ke 4, encoder 8 ke 3, encoder keypad atau tombol dan lain sebagainya. Kebalikan dari rangkaian encoder adalah rangkaian decoder.

Gambar berikut menunjukkan skema rangkaian encoder keypad atau encoder tombolSebenarnya ada banyak jenis encoder tombol namun yang paling simpel dan mudah dipelajari adalah sebagai berikut



Gambar 7.1 Rangkaian Encoder tombol

Prinsip kerja encoder tombol :

Contoh sederhana dari penggunaan encoder tombol adalah pada kalkulator Rangkaian encoder tombol terdiri dari bermacam-macam namun encoder yang paling sederhana adalah seperti diatas. Rangkaian encoder tombol pada gambar diatas akan mengkodekan tombol 10 jalur menjadi 4 jalur biner. Dengan demikian rangkaian diatas dapat menghemat jalur yang digunakan Berikut tabel kebenaran encoder diatas.

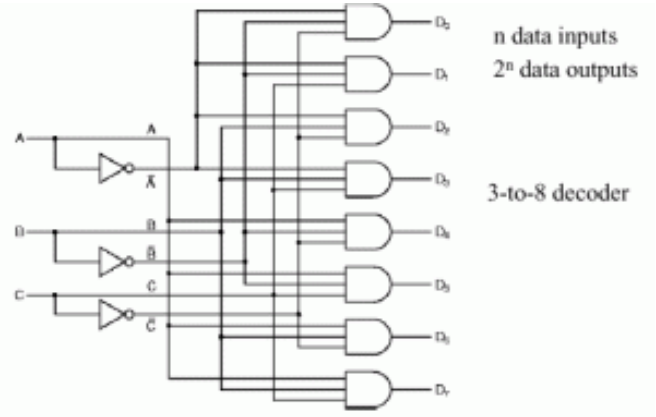
Tabel 7.1 Tabel Kebenaran Encoder

S0	S1	S2	S3	S4	S5	S6	S7	S8	S9	D0	D1	D2	D3
0	1	1	1	1	1	1	1	1	1	0	0	0	0
1	0	1	1	1	1	1	1	1	1	1	0	0	0
1	1	0	1	1	1	1	1	1	1	0	1	0	0
1	1	1	0	1	1	1	1	1	1	1	1	0	0
1	1	1	1	0	1	1	1	1	1	0	0	1	0
1	1	1	1	1	0	1	1	1	1	1	0	1	0
1	1	1	1	1	1	0	1	1	1	0	1	1	0
1	1	1	1	1	1	1	0	1	1	1	1	1	0
1	1	1	1	1	1	1	1	0	1	0	0	0	1
1	1	1	1	1	1	1	1	1	0	1	0	0	1

rangkaian encoder tombol seperti diatas sering ditemukan pada rangkaian mikrokontroler. Untuk mendecodekan kode diatas menjadi seperti semula maka dibutuhkan rangkaian decoder. Rangkaian encoder dapat dibuat dengan menggunakan kombinasigerbang logika. Serangkaian gerbang logika diatur sedemikian rupa sehingga dapat mengkodekan kode input. Pada umumnya rangkaian encoder mempunyai jumlah input banyak dan mempunyai jumlah output yang lebih sedikit.

7.2 Decoder

Pengertian Decoder adalah alat yang di gunakan untuk dapat mengembalikan proses encoding sehingga kita dapat melihat atau menerima informasi aslinya. Pengertian Decoder juga dapat di artikan sebagai rangkaian logika yang di tugaskan untuk menerima input input biner dan mengaktifkan salah satu outputnya sesuai dengan urutan biner tersebut. Kebalikan dari decoder adalah encoder.



Gambar 7.2 Rangkaian Decoder

Fungsi Decoder adalah untuk memudahkan kita dalam menyalakan seven segmen. Itu lah sebabnya kita menggunakan decoder agar dapat dengan cepat menyalakan seven segmen. Output dari decoder maksimum adalah 2n. Jadi dapat kita bentuk n-to-2n decoder. Jika kita ingin merangkaian decoder dapat kita buat dengan 3-to-8 decoder menggunakan 2-to-4 decoder. Sehingga kita dapat membuat 4-to-16 decoder dengan menggunakan dua buah 3-to-8 decoder.

Beberapa rangkaian decoder yang sering kita jumpai saat ini adalah decoder jenis 3 x 8 (3 bit input dan 8 output line), decoder jenis 4 x 16, decoder jenis BCD to Decimal (4 bit input dan 10 output line) dan decoder jenis BCD to 7 segmen (4 bit input dan 8 output line). Khusus untuk *pengertian decoder* jenis BCD to 7 segmen mempunyai prinsip kerja yang berbeda dengan decoder decoder lainnya, di mana kombinasi setiap inputnya dapat mengaktifkan beberapa output linenya.

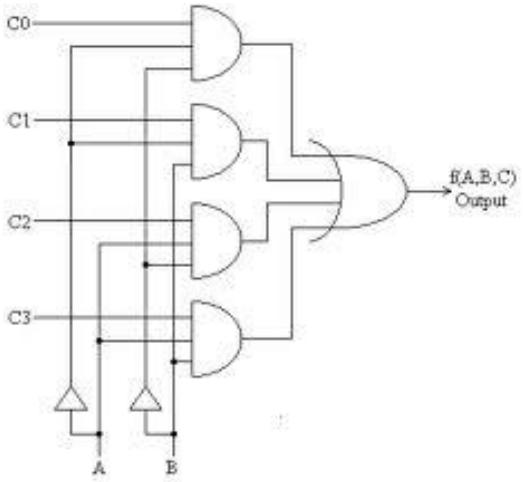
Salah satu jenis IC decoder yang umum di pakai adalah 74138, karena IC ini mempunyai 3 input biner dan 8 output line, di mana nilai output adalah 1 untuk salah satu dari ke 8 jenis kombinasi inputnya. Jika kita perhatikan, pengertian decoder sangat mirip dengan demultiplexer dengan pengecualian yaitu decoder yang satu ini tidak mempunyai data input. Sehingga input hanya di gunakan sebagai data control.

Pengertian decoder dapat di bentuk dari susunan gerbang logika dasar atau menggunakan IC yang banyak jual di pasaran, seperti decoder 74LS48, 74LS154, 74LS138, 74LS155 dan sebagainya. Dengan menggunakan IC, kita dapat merancang sebuah decoder dengan jumlah bit dan keluaran yang di inginkan. Contohnya adalah dengan merancang sebuah decoder 32 saluran keluar dengan IC decoder 8 saluran keluaran.

7.3 Multiplexer

Multiplexer adalah rangkaian logika yang menerima beberapa input data digital dan menyeleksi salah satu dari input tersebut pada saat tertentu, untuk dikeluarkan pada sisi output.

Seleksi data-data input dilakukan oleh selector line, yang juga merupakan input dari multiplexer tersebut. Blok diagram sebuah multiplexer ditunjukkan pada gambar dibawah ini :



Gambar 7.3 Rangkaian Multiplexer

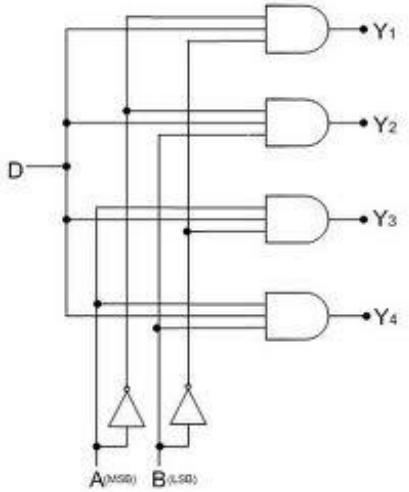
Data select control input		Data output
A	B	Selected
0	0	c0
0	1	c1
1	0	c2
1	1	c3

Penjelasan pada gambar diatas sebagai berikut :
Diagram logika untuk 4 jalur multiplexer dengan A = 0, B = 1 (Data C1 yang dipilih)

7.4 Demultiplexer

Demultiplexer adalah sebuah rangkain logika yang menerima satu input data dan mendistribusikan input tersebut ke beberapa output yang tersedia, dan juga merupakan kebalikkan dari multiplexer.

Selekai data - data input dilakukan oleh selektor line, yang juga merupakan input dari demultiplexer tersebut. Blok diagram sebuah dimultiplexer ditunjukkan pada gambar tersebut.



Gambar 7.4 rangkaian demultiplexer

Data select

control input			Data output			
A	B	Ipn	y1	y2	y3	y4
0	0	0	0	x	x	x
0	0	1	1	x	x	x
0	1	0	x	0	x	x
0	1	1	x	1	x	x
1	0	0	x	x	0	x
1	0	1	x	x	1	x
1	1	0	x	x	x	0
1	1	1	x	x	x	1

Penjelasan pada gambar diatas sebagai berikut :
Diagram logika untuk 4 jalur multiplexer dengan A = 0, B = 1 (Data y2 yang dipilih)

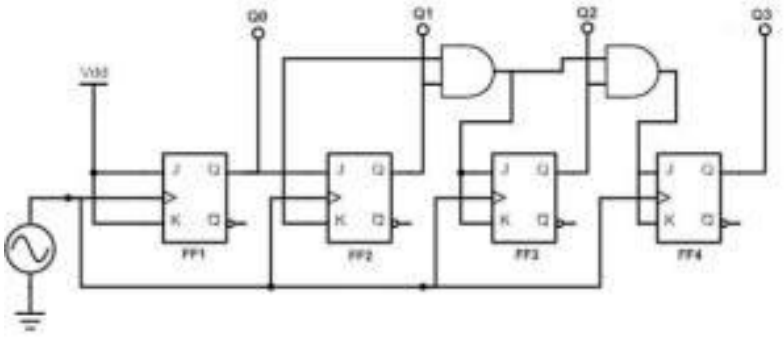
BAB 8
COUNTER

Counter

Counter atau pencacah adalah suatu peranti elektronik yang digunakan atau dapat digunakan untuk menghitung jumlah pulsa yang masuk melalui inputnya. Jenis-jenis counter menurut hitungan Dilihat dari cara counter menghitung maka counter dibagi menjadi beberapa, jenis-jenis counter yang diantaranya adalah Up counter & Down counter

1. Up counter

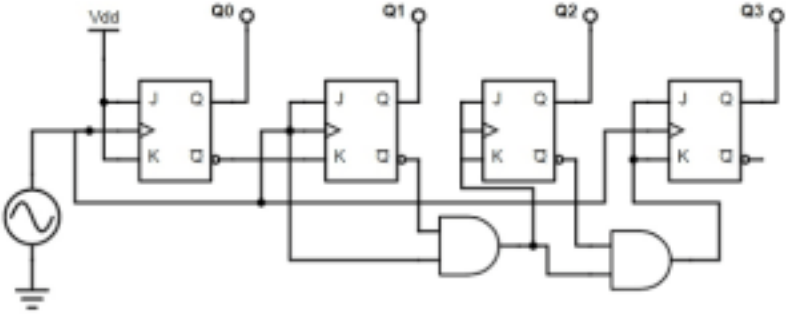
Up counter adalah rangkaian counter yang berfungsi menghitung naik. Rangkaian up counter dapat di buat dengan menggunakan D Flip-flop maupun JK Flip-flop. Berikut merupakan skema rangkaian up counter menggunakan JK flip-flop.



Gambar 8.1 Rangkaian up counter

2. Down counter

Down counter adalah rangkaian yang berfungsi menghitung turun. Counter jenis ini dapat sobat temui pada lampu lalu lintas dimana bilangan akan menghitung mundur sampai angka 0. Contoh skemanya adalah sebagai berikut.



Gambar 8.2 rangkaian down counter

BAB 9
ADC dan DAC

ADC (ANALOG TO DIGITAL CONVERTER)

ADC (Analog To Digital Converter) adalah perangkat elektronika yang berfungsi untuk mengubah sinyal analog (sinyal kontinyu) menjadi sinyal digital. Perangkat ADC (Analog To Digital Convertion) dapat berbentuk suatu modul atau rangkaian elektronika maupun suatu chip IC. ADC (Analog To Digital Converter) berfungsi untuk menjembatani pemrosesan sinyal analog oleh sistem digital.

Converter

Alat bantu digital yang paling penting untuk teknologi kontrol proses adalah yang menerjemahkan informasi digital ke bentuk analog dan juga sebaliknya. Sebagian besar pengukuran variabel-variabel dinamik dilakukan oleh piranti ini yang menerjemahkan informasi mengenai vaiabel ke bentuk sinyal listrik analog. Untuk menghubungkan sinyal ini dengan sebuah komputer atau rangkaian logika digital, sangat perlu untuk terlebih dahulu melakukan konversi analog ke digital (A/D). Hal-hal mengenai konversi ini harus diketahui sehingga ada keunikan, hubungan khusus antara sinyal analog dan digital.

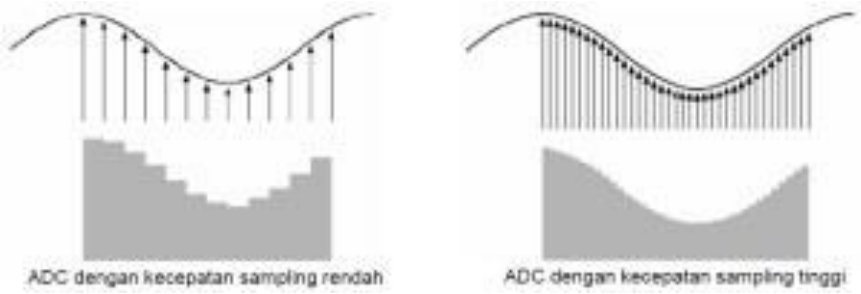
9.1 ADC (Analog to Digital Convertion)

Analog To Digital Converter (ADC) adalah pengubah input analog menjadi kode – kode digital. ADC banyak digunakan sebagai Pengatur proses industri, komunikasi digital dan rangkaian pengukuran/ pengujian. Umumnya ADC digunakan sebagai perantara antara sensor yang kebanyakan analog dengan sistim komputer seperti sensor suhu, cahaya, tekanan/ berat, aliran dan

sebagainya kemudian diukur dengan menggunakan sistim digital (komputer). ADC (Analog to Digital Converter) memiliki 2 karakter prinsip, yaitu kecepatan sampling dan resolusi.

Kecepatan Sampling ADC

Kecepatan sampling suatu ADC menyatakan "seberapa sering sinyal analog dikonversikan ke bentuk sinyal digital pada selang waktu tertentu". Kecepatan sampling biasanya dinyatakan dalam sample per second (SPS).



Gambar 9.1 Ilustrasi Kecepatan Sampling ADC

Resolusi ADC

Resolusi ADC menentukan "ketelitian nilai hasil konversi ADC". Sebagai contoh: ADC 8 bit akan memiliki output 8 bit data digital, ini berarti sinyal input dapat dinyatakan dalam 255 ($2^n - 1$) nilai diskrit. ADC 12 bit memiliki 12 bit output data digital, ini berarti sinyal input dapat dinyatakan dalam 4096 nilai diskrit. Dari contoh diatas ADC 12 bit akan memberikan ketelitian nilai hasil konversi yang jauh lebih baik daripada ADC 8 bit.

Prinsip Kerja ADC

Prinsip kerja ADC adalah mengkonversi sinyal analog ke dalam bentuk besaran yang merupakan rasio perbandingan sinyal input dan tegangan referensi. Sebagai contoh, bila tegangan referensi 5 volt, tegangan input 3 volt, rasio input terhadap referensi adalah 60%. Jadi, jika menggunakan ADC 8 bit dengan skala maksimum 255, akan didapatkan sinyal digital sebesar 60% x 255 = 153 (bentuk decimal) atau 10011001 (bentuk biner).

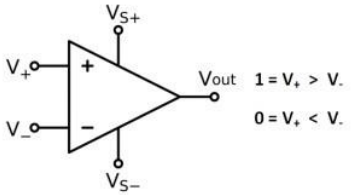
signal = (sample/max_value) * reference_voltage

= (153/255) * 5

= 3 Volts

Komparator ADC

Bentuk komunikasi yang paling mendasar antara wujud digital dan analog adalah piranti (biasanya berupa IC) disebut komparator. Piranti ini, yang diperlihatkan secara skematik pada gambar dibawah, secara sederhana membandingkan dua tegangan pada kedua terminal inputnya. Bergantung pada tegangan mana yang lebih besar, outputnya akan berupa sinyal digital 1 (high) atau 0 (low). Komparator ini digunakan secara luas untuk sinyal alarm ke komputer atau sistem pemroses digital. Elemen ini juga merupakan satu bagian dengan konverter analog ke digital dan digital ke analog yang akan didiskusikan nanti.



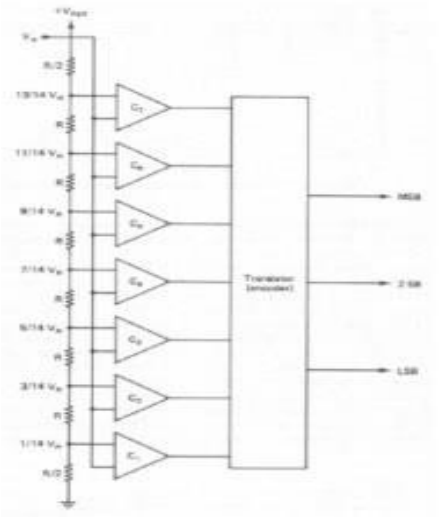
Gambar 9.2 Konsep Kompataror Pada ADC (Analog to Digital Converter)

Gambar diatas memperlihatkan sebuah komparator merubah keadaan logika output sesuai fungsi tegangan input analog. Sebuah komparator dapat tersusun dari sebuah opamp yang memberikan output terpotong untuk menghasilkan level yang diinginkan untuk kondisi logika (+5 dan 0 untuk TTL 1 dan 0). Komparator komersil didesain untuk memiliki level logika yang dperlukan pada bagian outputnya.

Jenis-Jenis ADC (Analog to Digital Converter)

a. ADC Simultan

ADC Simultan atau biasa disebut flash converter atau parallel converter. Input analog Vi yang akan diubah ke bentuk digital diberikan secara simultan pada sisi + pada komparator tersebut, dan input pada sisi – tergantung pada ukuran bit converter. Ketika Vi melebihi tegangan input – dari suatu komparator, maka output komparator adalah high, sebaliknya akan memberikan output low.



Gambar Rangkaian 9.3 ADC Simultan

Bila Vref diset pada nilai 5 Volt, maka dari gambar 3 dapat didapatkan :

V(-) untuk C7 = Vref * (13/14) = 4,64

V(-) untuk C6 = Vref * (11/14) = 3,93

V(-) untuk C5 = Vref * (9/14) = 3,21

V(-) untuk C4 = Vref * (7/14) = 2,5

V(-) untuk C3 = Vref * (5/14) = 1,78

V(-) untuk C2 = Vref * (3/14) = 1,07

V(-) untuk C1 = Vref * (1/14) = 0,36

Misal :

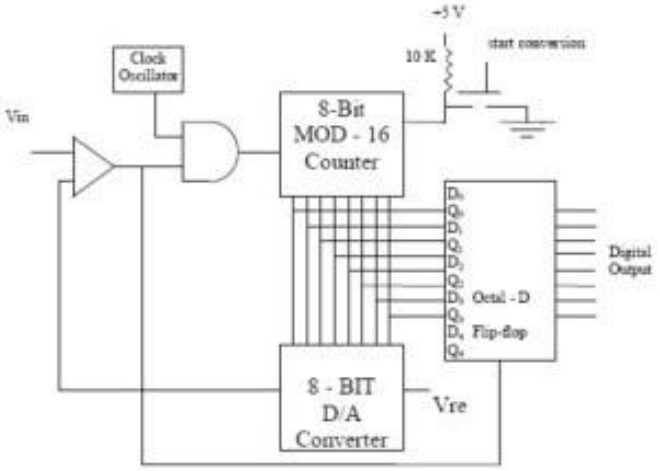
Vin diberi sinyal analog 3 Volt, maka output dari C7=0, C6=0, C5=0, C4=1, C3=1, C2=1, C1=1, sehingga didapatkan output ADC yaitu 100 biner.

Tabel 9.1 Tabel Output ADC Simultan

Output Comparator							Output Translator		
C7	C6	C5	C4	C3	C2	C1	2^2	2^1	2^0
0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	1
0	0	0	0	0	1	1	0	1	0
0	0	0	0	1	1	1	0	1	1
0	0	0	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1

Ada beberapa konsep dasar dari ADC adalah dengan cara Counter Ramp ADC, Successive Aproximation ADC dan lain sebagainya.

b. Counter Ramp ADC



Gambar 9.4 rangkaian Blok Diagram Counter Ramp ADC

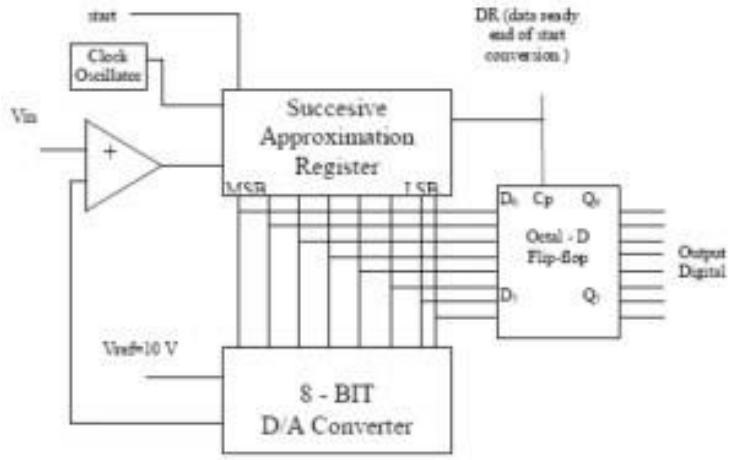
Pada gambar diatas, ditunjukkan blok diagram Counter Ramp ADC didalamnya tedapat DAC yang diberi masukan dari counter, masukan counter dari sumber Clock dimana sumber Clock dikontrol dengan cara meng AND kan dengan keluaran Comparator. Comparator membandingkan antara tegangan masukan analog dengan tegangan keluaran DAC, apabila tegangan masukan yang akan dikonversi belum sama dengan tegangan keluaran dari DAC maka keluaran comparator = 1 sehingga Clock dapat memberi masukan counter dan hitungan counter naik.

Misal akan dikonversi tegangan analog 2 volt, dengan mengasumsikan counter reset, sehingga keluaran pada DAC juga 0 volt. Apabila konversi dimulai maka counter akan naik dari 0000 ke 0001 karena mendapatkan pulsa masuk dari Clock oscillator dimana saat itu keluaran Comparator = 1, karena mendapatkan kombinasi biner dari counter 0001 maka tegangan keluaran DAC naik dan dibandingkan lagi dengan tegangan masukan demikian seterusnya

nilai counter naik dan keluaran tegangan DAC juga naik hingga suatu saat tegangan masukan dan tegangan keluaran DAC sama yang mengakibatkan keluaran komparator = 0 dan Clock tidak dapat masuk. Nilai counter saat itulah yang merupakan hasil konversi dari analog yang dimasukkan.

Kelemahan dari counter tersebut adalah lama, karena harus melakukan trace mulai dari 0000 hingga mencapai tegangan yang sama sehingga butuh waktu.

c. SAR (Successive Aproximation Register) ADC



Gambar 9.5 rangkaian Blok Diagram SAR ADC

Pada gambar diatas ditunjukkan diagram ADC jenis SAR, Yaitu dengan memakai konvigurasi yang hampir sama dengan counter ramp tetapi dalam melakukan trace dengan cara tracking dengan mengeluarkan kombinasi bit MSB = 1 ====> 1000 0000. Apabila belum sama (kurang dari tegangan analog input maka bit MSB berikutnya = 1 ===>1100 0000) dan apabila tegangan analog input

ternyata lebih kecil dari tegangan yang dihasilkan DAC maka langkah berikutnya menurunkan kombinasi bit ====> 10100000.

Untuk mempermudah pengertian dari metode ini diberikan contoh seperti pada timing diagram gambar 6 Misal diberi tegangan analog input sebesar 6,84 volt dan tegangan referensi ADC 10 volt sehingga apabila keluaran tegangan sbb :

Jika D7 = 1 Vout=5 volt

Jika D6 = 1 Vout=2,5 volt

Jika D5 = 1 Vout=1,25 volt

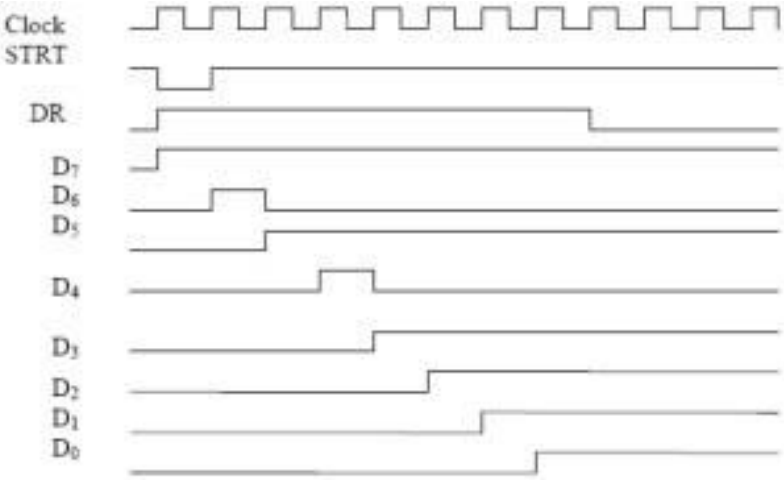
Jika D4 = 1 Vout=0,625 volt

Jika D3 = 1 Vout=0,3125 volt

Jika D2 = 1 Vout=0,1625 volt

Jika D1 = 1 Vout=0,078125 volt

Jika D0 = 1 Vout=0,0390625 volt



Gambar 9.6 rangkaian Timing diagram urutan Trace SAR ADC

Setelah diberikan sinyal start maka konversi dimulai dengan memberikan kombinasi 1000 0000 ternyata menghasilakan tegangan 5 volt dimana masih kurang dari tegangan input 6,84 volt, kombinasi

berubah menjadi 1100 0000 sehingga Vout = 7,5 volt dan ternyata lebih besar dari 6,84 sehingga kombinasi menjadi 1010 0000 tegangan Vout = 6,25 volt kombinasi naik lagi 1011 0000 demikian seterusnya hingga mencapai tegangan 6,8359 volt dan membutuhkan hanya 8 clock.

Uraian diatas merupakan konsep dasar dari ADC (Analog to Digital Converter), untuk pengembangan atau aplikasi ADC dan ADC dalam bentuk lain akan ditulis dalam artikel berbeda dengan tujuan dapat memberikan penjelasan yang lebih lengkap dari ADC (Analog to Digital Converter)

9.2 DAC (Digital To Analog Converter)

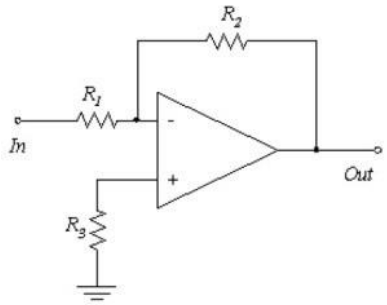
DAC (Digital To Analog Converter) adalah perangkat elektronika yang berfungsi untuk mengubah sinyal digital (diskrit) menjadi sinyal analog (kontinyu). Aplikasi DAC (Digital To Analog Converter) adalah sebagai antarmuka (interface) antara perangkat yang bekerja dengan sistem digital dan perangkat pemroses sinyal analog. Perangkat DAC (Digital To Analog Converter) dapat berupa rangkaian elektronika dan chip IC DAC.

Konsep Dasar DAC (Digital To Analog Converter)

Pada dasarnya rangkaian penjumlah op-amp (summing amplifier) dapat digunakan untuk menyusun suatu konverter D/A (DAC “Digital To Analog Converter)” dengan memakai sejumlah hambatan masukan yang diberi bobot dalam deret biner.

Penguat Inverting

Rangkaian untuk penguat inverting adalah seperti yang ditunjukkan gambar dibawah. Penguat ini memiliki ciri khusus yaitu sinyal keluaran memiliki beda fasa sebesar 180°.



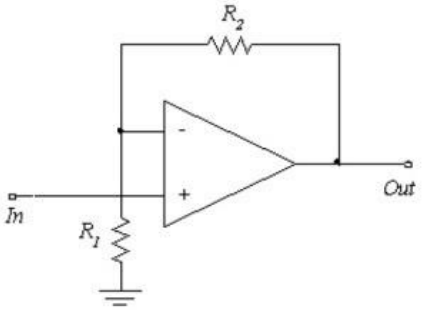
Gambar 9.7 Rangkaian penguat Inverting

Penguatan rangkaian penguat inverting adalah berdasar pada persamaan berikut :

$V_{out} = -V_{in}(R2/R1)$

Penguat Non-Inverting

Penguat non-inverting memiliki ciri khusus yaitu sinyal output adalah sefasa dengan sinyal masukan. Rangkaian ini ditunjukkan oleh gambar berikut.



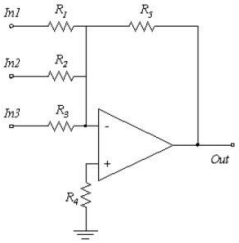
Gambar 9.8 rangkaian penguat Non-Inverting

Penguatan dari rangkaian penguat jenis ini adalah berdasar pada persamaan berikut :

$V_{out} = V_{in}((R1+R2)/R1)$

Penguat Penjumlah (Dasar DAC)

Penguat penjumlah memiliki ciri khusus yaitu sinyal keluaran merupakan hasil penguatan dari penjumlahan sinyal masukannya. Pada bagian ini dicontohkan penguat penjumlah berdasarkan rangkaian penguat inverting. Sehingga sinyal keluaran adalah berbeda fasa sebesar 180^o. Rangkaian penguat penjumlah merupakan konsep dasar dari rangkaian DAC (Digital To Analog Converter).



Gambar 9.9 Penguat Penjumlah (Dasar DAC)

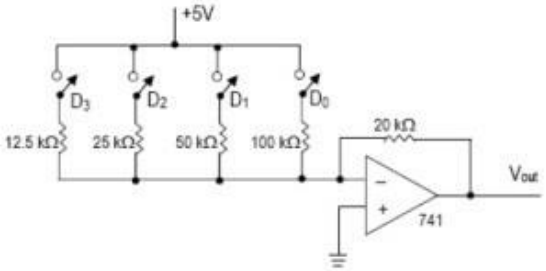
Penguatan dari rangkaian ini dihitung menggunakan persamaan berikut :

$V_{out} = (-V_{in1}(R5/R1))+(-V_{in2}(R5/R2))+(-V_{in3}(R5/R3))$

Jenis-Jenis DAC (Digital To Analog Converter)

a. Binary-Weighted DAC (Digital To Analog Converter)

Suatu rangkaian Binary-weighted DAC dapat disusun dari beberapa Resistor dan Operational Amplifier (Op-Amp) seperti gambar berikut.



Gambar 9.10 Rangkaian Binary Weighted DAC

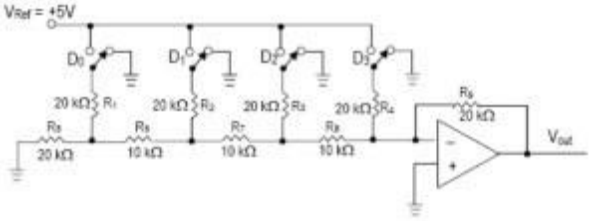
Secara prinsip rangkaian DAC diatas dapat dijelaskan sebagai berikut. Resistor 20 kΩ menjumlahkan arus yang dihasilkan dari penutupan switch-switch D_0 sampai D_3. Resistor-resistor ini diberi skala nilai sedemikian rupa sehingga memenuhi bobot biner (binary-weighted) dari arus yang selanjutnya akan dijumlahkan oleh resistor 20 kΩ. Dengan menutup D_0 menyebabkan arus 50 µA mengalir melalui resistor 20 kΩ, menghasilkan tegangan -1 V pada V_{out}. Penutupan masing-masing switch menyebabkan penggandaan nilai arus yang dihasilkan dari switch sebelumnya. Nilai konversi dari kombinasi penutupan switch ditunjukkan pada tabel berikut.

Tabel 9.2 R/2R Ladder DAC (Digital To Analog Converter)

D3	D2	D1	D0	V_{out} (V)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

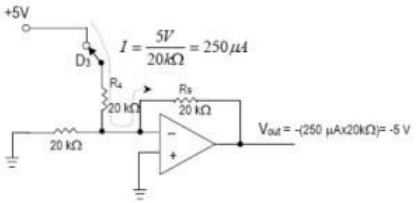
b. R/2R Ladder DAC (Digital To Analog Converter)

Metode lain dari konversi Digital to Analog adalah R/2R Ladder. Metode ini banyak digunakan dalam IC-IC DAC. Pada rangkaian R/2R Ladder, hanya dua nilai resistor yang diperlukan, yang dapat diaplikasikan untuk IC DAC dengan resolusi 8,10 atau 12 bit. Rangkaian R/2R Ladder ditunjukkan pada gambar berikut.



Gambar 9.11 Rangkaian R/2R Ladder DAC

Prinsip kerja dari rangkaian R/2R Ladder DAC adalah sebagai berikut : informasi digital 4 bit masuk ke switch D_0 sampai D_3. Switch ini mempunyai kondisi "1" (sekitar 5 V) atau "0" (sekitar 0 V). Dengan pengaturan switch akan menyebabkan perubahan arus yang mengalir melalui R_9 sesuai dengan nilai ekivalen biner-nya Sebagai contoh, jika $D_0 = 0$, $D_1 = 0$, $D_2 = 0$ dan $D_3 = 1$, maka R_1 akan paralel dengan R5menghasilkan 10 k . Selanjutnya 10 k ini seri dengan R6 = 10 k menghasilkan 20 k . 20 k ini paralel dengan R2 menghasilkan 10 k , dan seterusnya sampai R7, R3 dan R8. Rangkaian ekivalennya ditunjukkan pada gambar 6. Vout yang dihasilkan dari kombinasi switch ini adalah -5V.



Gambar 9.12 Rangkaian Ekivalen R/2R Ladder DAC

Untuk mendapatkan V_{out} analog dari rangkaian R/2R Ladder DAC diatas dapat dihitung dengan menggunakan persamaan :

$V_{out} = (-V_{ref}(R9/R))*((D0/16)+(D1/8)+(D2/4)+(D1/2))$

Tabel Output Rangkaian R/2R Ladder DAC

Nilai kombinasi dan hasil konversi rangkaian R/2R Ladder DAC ditunjukkan pada tabel dibawah.

Tabel 9.3 Tabel Output Rangkaian R/2R Ladder DAC

D3	D2	D1	D0	V_{out} (V)
0	0	0	0	0.000
0	0	0	1	0.625
0	0	1	0	1.250
0	0	1	1	1.875
0	1	0	0	2.500
0	1	0	1	3.125
0	1	1	0	3.750
0	1	1	1	4.375
1	0	0	0	5.000
1	0	0	1	5.625
1	0	1	0	6.125
1	0	1	1	6.875
1	1	0	0	7.500
1	1	0	1	8.125
1	1	1	0	8.750
1	1	1	1	9.375

Tabel diatas merupakan hasil konversi dari nilai digital ke nilai analog berdasarkan rangkaian R/2R Ladder DAC (Digital To Analog Converter).

DAFTAR PUSTAKA

1. Malvino and Leach, ***Digital principles and Applications***, ed 5, McGraw Hill, 1995
2. Tocci, Ronald J, ***Digital Systems Principles and Applications***, ed 6, Prentice Hall,
3. Elektronika Digiltal konsep dasar dana plikasinya, Sumarna,GRAHA ILMU
4. Malvino, ***Elektronika Komputer Digital***, terj. Dali S Naga, Gunadarma
5. Suryadi, Agus S, ***Dasar Rangkaian Logika***, jilid I, Gunadarma
6. Bartee, Thomas C, ***Dasar Komputer Digital***, terj. The How Liong, ed. 6, Penerbit Erlangga, 1994
7. Wakerle, John F, ***Digital Principles and Practices***, Prentice Hall, 1994
8. Lee, Samuel C, ***Rangkaian Digital dan Rancangan Logika***, terj. Sutisno, Erlangga, 1991
9. Mano M, Morris and Kime R, Charles, ***Logic and Computer Design Fundamentals***, Prentice Hall, 1997
10. Ronald J. Tocci, Neal S.Widmer, Gregory L. Moss, Digital Systems Principles and Applications TENTH EDITION, 2007, Pearson Education International.
11. Harper C.A., 1996. *Active Electronic Component Handbook*. 2nd McGraw-Hill. Inc
12. Kuphaldt, Tony. 2007. *Electric Circuit, Volume IV – Digital,* Design Science Licences
13. Kuphaldt, Tony. 2008. *Electric Circuit, Volume 1 – Direct Current,* Design Science Licences
14. Kurniawan, Freddy. 2005. *Sistem Digital Konsep dan Aplikasi*. Penerbit Gavamedia, Yogyakarta.

15. Leach, Malvino.1994. *Digital Principles And Applications Third Edition*, McGraw-Hill,Inc
16. Soedarto, Gatot. 1987. *Teknik Digital KomputerDasar – dasar Sistem Digital*. Penerbit Usaha Nasional, Surabaya
17. Sunarto, 1998. *Dasar-dasar Teknologi Digital*. Jakarta
18. Varjaman. J 1993. *Surface Mount Technology*. Recent Japanese Development. IEEE Press
19. Widjanarka, Wijaya. 2006. *Teknik Digital*. Penerbit Erlangga, Jakarta.

BIODATA PENULIS

Dr.Hindarto, S.Kom, MT. dilahirkan di Surabaya, 30 Juli 1973. Pada tahun 1995, penulis mendapatkan gelar Diploma dari Politeknik Elektronika Negeri Surabaya dan melanjutkan jenjang Sarjana di prodi Informatika Fakultas Teknik Umsida. Penulis melanjutkan magister Teknik Elektro ITS dengan program beasiswa dari DIKTI. Tahun 2007, penulis secara resmi mendakatkan gelar M.T. Penulis melanjutkan doctoral di Jaringan Cerdas Multimedia Teknik ElektroITS dengan program beasiswa dari DIKTI. Tahun 2016, penulis secara resmi mendakatkan gelar Dr. Penulis mengawali karirnya sebagai Dosen di prodi Informatika Universitas Muhaammdiyah Sidoarjo pada Tahun 2004. Beberapa penelitian yang pernah dilakukan oleh penulis adalah tentang Deteksi Sinyal Jantung dan Deteksi Sinyal EEG. Penulis menjadi dosen di Umsida dengan mengampuh mata kuliah Sistem Digital, Kecerdasan Buatan dan Jaringan Syaraf Tiruan.