

PERWAKILAN NOMBOR

Sistem Nombor

- Kebanyakan sistem komputer (sistem Digital) melakukan operasi pengiraan nombor dalam kuantiti yang banyak.
- Maka, Sistem Pernomboran yang digunakan oleh Sistem Digital perlu diketahui dari segi:-

Kenapa Perlu Sistem Nombor

- Kenapa perlu ada berbagai sistem nombor?
 - Peralatan/peranti terdahulu melakukan pengiraan komputer menggunakan suis dan geganti.
 - Suis/geganti hanya ada dua keadaan sahaja iaitu 'tutup' dan 'buka'.
 - Maka operasi komputer tersebut dapat dinyatakan dalam digit nombor binari, yang terdiri daripada dua keadaan sahaja.
- Jika sistem Desimal diguna, maka ia memerlukan suatu peralatan/peranti yang menghasilkan 10 keadaan! Hasilnya suatu peranti yang kompleks, tak 'reliable', tidak cekap dan mahal

Jenis-jenis Sistem Nombor

- Decimal (asas 10)
- Binary (asas 2)
- Octal (asas 8)
- Hexadecimal (asas 16)

Nombor Decimal

- Terdiri daripada 10 angka iaitu 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Ia merupakan nombor 'Asas 10'.
- Salah satu contoh dalam sistem nombor Decimal adalah **1428.79** atau **1428.7910**. Kedudukan setiap digit menunjukkan magnitud bagi setiap digit tersebut iaitu:-

Pemberat	10^3	10^2	10^1	10^0		10^{-1}	10^{-2}
Nilai	1	4	2	8	.	7	9

- Secara pernyataan matematik:-

$$142810 = 1 \times 10^3 + 4 \times 10^2 + 2 \times 10^1 + 8 \times 10^0$$

Nombor Binari

- Terdiri daripada 2 angka iaitu 0,1. Ia merupakan nombor 'Asas 2'.
- Salah satu contoh dalam sistem nombor Binari adalah **1001.01** atau **1001.012**. Kedudukan setiap digit menunjukkan magnitud bagi setiap digit tersebut iaitu:-

Pemberat	2^3	2^2	2^1	2^0		2^{-1}	2^{-2}
Nilai	1	0	0	1	.	0	1

- Secara pernyataan matematik:-

$$10012 = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

- Dalam no. Binari, bilangan digit dipanggil *bit*.

Nombor Oktal

- Terdiri daripada 8 angka iaitu 0, 1, 2, 3, 4, 5, 6, 7. Ia merupakan nombor 'Asas 8'.
- Salah satu contoh dalam sistem nombor Oktal adalah **5641.27** atau **5641.278**. Kedudukan setiap digit menunjukkan magnitud bagi setiap digit tersebut iaitu:-

Pemberat	8^3	8^2	8^1	8^0		8^{-1}	8^{-2}
Nilai	5	6	4	1	.	2	7

- Secara pernyataan matematik:-

$$56418 = 5 \times 8^3 + 6 \times 8^2 + 4 \times 8^1 + 1 \times 8^0$$

Nombor Heksadesimal

- Terdiri daripada 8 angka iaitu 0, 1, 2, 3, 4, 5, 6, 7. Ia merupakan nombor 'Asas 8'.
- Salah satu contoh dalam sistem nombor Octal adalah **5641.27** atau **5641.278**. Kedudukan setiap digit menunjukkan magnitud bagi setiap digit tersebut iaitu:-

Pemberat	16^3	16^2	16^1	16^0		16^{-1}	16^{-2}
Nilai	5	B	8	F	.	2	1

- Secara pernyataan matematik:-

$$5B8F_{16} = 5 \times 16^3 + B \times 16^2 + 8 \times 16^1 + F \times 16^0$$

Decimal (Perpuluhan)	Binary (Perduaan)	Octal (Perlapanan)	Hexadecimal (Perenambelasan)
0	0000	00	0
1	0001	01	1
2	0010	02	2
3	0011	03	3
4	0100	04	4
5	0101	05	5
6	0110	06	6
7	0111	07	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Penukaran Nombor

- Penukaran Binari ke Desimal

$$1001_2 = 1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$
$$= 8 + 0 + 0 + 1$$

- Penukaran Desimal ke Binari

$$18_{10} = ?_2$$

Berhenti apabila $N = 0$

$$18_{10} = 10010_2$$

<i>N</i>	<i>X</i>	<i>Baki</i> (<i>N-X</i>)
18	=18	0
9	=8	1
4	=4	0
2	=2	0
1	=0	1
0		

- Penukaran Oktal ke Desimal

$$1271_8 = 1 \times 8^3 + 2 \times 8^2 + 7 \times 8^1 + 1 \times 8^0$$
$$= 512 + 128 + 56 + 1$$

- Penukaran Oktal ke Desimal

$$6971_8 = ?_{10}$$

<i>N</i>	<i>X</i>	<i>Baki</i> (<i>N-X</i>)
697	=696	1
87	=80	7
10	=8	2
1	=0	1
0		

Contoh 1:

Tukarkan nilai 10110101_2 kepada nilai perpuluhan:

Penyelesaian:

$$1\ 0\ 1\ 1\ 0\ 1\ 0\ 1_2 = 2^7 + 2^5 + 2^4 + 2^2 + 2^0 = 181_{10}$$

Contoh 2:

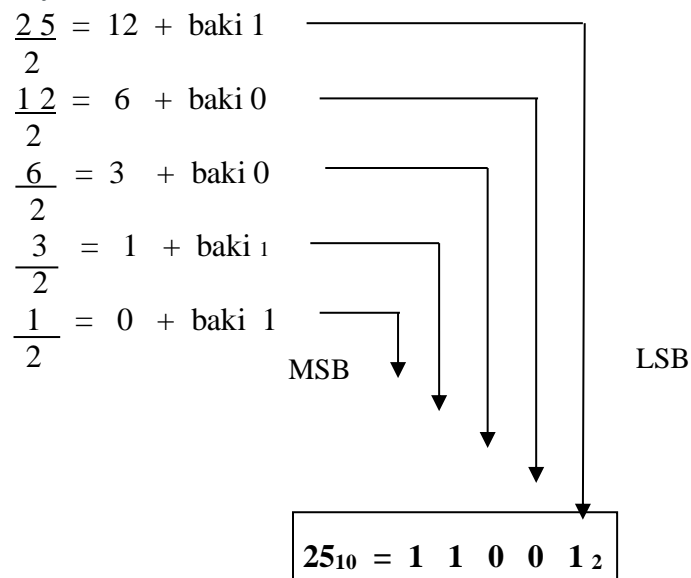
Tukarkan nilai 101.1_2 kepada nilai perpuluhan:

Penyelesaian:

$$1\ 0\ 1.1_2 = (1 \times 2^2) + (0 \times 2^1) + (1 \times 2^0) + (1 \times 2^{-1}) = 5.5_{10}$$

Contoh 3:

Tukarkan nilai 25_{10} kepada nilai perdua:

Penyelesaian:**Contoh 4:**

Tukarkan nilai 37_{10} kepada nilai perdua:

Penyelesaian:

$$\begin{array}{rcl} 37 & = & 18.5 \longrightarrow \text{baki } 1 \text{ (LSB)} \\ \hline 18 & = & 9.0 \longrightarrow \text{baki } 0 \\ \hline 9 & = & 4.5 \longrightarrow \text{baki } 1 \\ \hline 4 & = & 2.0 \longrightarrow \text{baki } 0 \\ \hline 2 & = & 1.0 \longrightarrow \text{baki } 0 \\ \hline 1 & = & 0.5 \longrightarrow \text{baki } 1 \text{ (MSB)} \end{array}$$

$37_{10} = 1\ 0\ 0\ 1\ 0\ 1_2$

Contoh 5:

Tukarkan nilai 372_8 kepada nilai perpuluhan:

Penyelesaian:

$$\begin{aligned}
 372_8 &= (3 \times 8^2) + (7 \times 8^1) + (2 \times 8^0) \\
 &= 250_{10}
 \end{aligned}$$

Contoh 6:

Tukarkan nilai 24.6_8 kepada nilai perpuluhan:

Penyelesaian:

$$\begin{aligned}
 24.6_8 &= (2 \times 8^1) + (4 \times 8^0) + (6 \times 8^{-1}) \\
 &= 20.75_{10}
 \end{aligned}$$

Contoh 7:

Tukarkan nilai 266_{10} kepada nilai perlapanan:

Penyelesaian:

$$\begin{aligned}
 \frac{266}{8} &= 33 \longrightarrow \text{baki } 2 \text{ (LSB)} \\
 \frac{33}{8} &= 4 \longrightarrow \text{baki } 1 \\
 \frac{4}{8} &= 0 \longrightarrow \text{baki } 4 \text{ (MSB)}
 \end{aligned}$$

$266_{10} = 412_8$

Contoh 8:

Tukarkan nilai 156.78_{10} kepada nilai perlapanan:

Penyelesaian:

$$\begin{aligned}
 \frac{156}{8} &= 19 \longrightarrow \text{baki } 4 \text{ (LSB)} \\
 \frac{19}{8} &= 2 \longrightarrow \text{baki } 3 \\
 \frac{2}{8} &= 0 \longrightarrow \text{baki } 2 \text{ (MSB)}
 \end{aligned}$$

$$\begin{aligned}
 0.78 \times 8 &= 6.24 \\
 0.24 \times 8 &= 1.92 \\
 0.92 \times 8 &= 7.36 \\
 0.36 \times 8 &= 2.88
 \end{aligned}$$

$156.78_{10} = 234.6172_8$

Contoh 9:

Tukarkan nilai 478_8 kepada nilai perdua:

Penyelesaian:

$$478_8 = \underbrace{100}_4 \underbrace{111}_7 \underbrace{010}_2$$

Contoh 10:

Tukarkan nilai 543_8 kepada nilai perduaan:

Penyelesaian:

$$5\ 4\ 3\ 1_8 = \underbrace{1\ 0\ 1}_5 \underbrace{1\ 0\ 0}_4 \underbrace{0\ 1\ 1}_3 \underbrace{0\ 0\ 1}_1_2$$

Contoh 11:

Tukarkan nilai 100111010_2 kepada nilai perlapanan:

Penyelesaian:

$$1\ 0\ 0\ 1\ 1\ 1\ 0\ 1\ 0_2 = \underbrace{1\ 0\ 0}_4 \underbrace{1\ 1\ 1}_7 \underbrace{0\ 1\ 0}_{2_8}$$

Contoh 12:

Tukarkan nilai 100111010_2 kepada nilai perlapanan:

Penyelesaian:

$$\begin{aligned} A\ B\ C\ D_{16} &= (10 \times 16^3) + (11 \times 16^2) + (12 \times 16^1) + (13 \times 16^0) \\ &= 4\ 3\ 9\ 8_{10} \end{aligned}$$

Contoh 13:

Tukarkan nilai $9F2_{16}$ kepada nilai perduaan:

Penyelesaian:

$$9\ F\ 2_{16} = \underbrace{1\ 0\ 0\ 1}_9 \underbrace{1\ 1\ 1\ 1}_F \underbrace{0\ 0\ 1\ 0}_2_2$$

Contoh 14:

Tukarkan nilai $ABF.2_{16}$ kepada nilai perduaan:

Penyelesaian:

$$A\ B\ F . 2_{16} = \underbrace{1\ 0\ 1\ 0}_A \underbrace{1\ 0\ 1\ 1}_B \underbrace{1\ 1\ 1\ 1}_F . \underbrace{0\ 0\ 1\ 0}_2_2$$

Contoh 15:

Tukarkan nilai 101110100110_2 kepada nilai perenambelasan:

Penyelesaian:

$$1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0_2 = \underbrace{1\ 0\ 1\ 1}_B \underbrace{1\ 0\ 1\ 0}_A \underbrace{0\ 1\ 1\ 0}_6$$

Contoh 16:

Tukarkan nilai 11011101011.11_2 kepada nilai perenambelasan:

Penyelesaian:

$$1\ 1\ 0\ 1\ 1\ 1\ 0\ 1\ 0\ 1\ 1 . 1\ 1_2 = \underbrace{1\ 1\ 0\ 1}_D \underbrace{1\ 1\ 1\ 0}_E \underbrace{1\ 0\ 1\ 1}_B . \underbrace{1\ 1\ 0\ 0}_C$$

Contoh 17:*Tukarkan $3\ 5\ 6_{16}$ kepada nilai desimal***Penyelesaian:**

$$3\ 5\ 6_{16} = (3 \times 16^2) + (5 \times 16^1) + (6 \times 16^0) = \mathbf{8\ 5\ 4_{10}}$$

Contoh 18:*Tukarkan $3\ 4\ 5.2_{16}$ kepada nilai desimal***Penyelesaian:**

$$3\ 4\ 5.2_{16} = (3 \times 16^2) + (4 \times 16^1) + (5 \times 16^0) + (2 \times 16^{-1}) = \mathbf{8\ 3\ 7.1\ 2\ 5_{10}}$$

Contoh 19:*Tukarkan $4\ 2\ 3_{10}$ kepada nilai heksadesimal.***Penyelesaian:**

$$\begin{array}{rcl} 4\ 2\ 3 & = & 2\ 6 \longrightarrow \text{baki } 7 \text{ (LSB)} \\ \hline 1\ 6 & & \\ 2\ 6 & = & 1 \longrightarrow \text{baki } 10 \\ \hline 1\ 6 & & \\ 1 & = & 0 \longrightarrow \text{baki } 1 \text{ (MSB)} \\ \hline 1\ 6 & & \end{array}$$

$\mathbf{4\ 2\ 3_{10} = 1\ A\ 7_{16}}$
--

Contoh 20:*Tukarkan 312.78_{10} kepada nilai heksadesimal***Penyelesaian:**

$$\begin{array}{rcl} 3\ 1\ 2 & = & 19 \longrightarrow \text{baki } 8 \text{ (LSB)} \\ \hline 1\ 6 & & \\ 1\ 9 & = & 1 \longrightarrow \text{baki } 3 \\ \hline 1\ 6 & & \\ 1 & = & 0 \longrightarrow \text{baki } 1 \text{ (MSB)} \\ \hline 1\ 6 & & \end{array}$$

$$0.78 \times 16 = 12.48$$

$$0.48 \times 16 = 7.68$$

$$0.68 \times 16 = 10.88$$

$$0.88 \times 16 = 14.08$$

$\mathbf{312.78_{10} = 1\ 3\ 8.C\ 7A\ E_{16}}$
--

Contoh 21:*Selesaikan operasi tambah 564_8 dan 777_8* **Penyelesaian:**

$$\begin{array}{r} 5\ 6\ 4_8 \\ + 7\ 7\ 7_8 \\ \hline 1\ 5\ 6\ 3_8 \end{array}$$

Selesaikan operasi tolak 621_8 dan 267_8

Penyelesaian:

$$\begin{array}{r} 6 \ 2 \ 1_8 \\ - 2 \ 6 \ 7_8 \\ \hline 3 \ 3 \ 2_8 \end{array}$$

Contoh 23:

Selesaikan operasi tambah ABC_{16} dan 777_{16}

Penyelesaian:

$$\begin{array}{r} \text{A B C} \\ + \quad 7 \quad 7 \quad 7 \\ \hline \mathbf{1\ 2\ 3\ 3}_{16} \end{array}$$

Contoh 24:

Selesaikan operasi tolak 621_{16} dan 267_{16}

Penyelesaian:

$$\begin{array}{r} 6 \quad 2 \quad 1 \quad 16 \\ - 2 \quad 6 \quad 7 \quad 16 \\ \hline 3 \quad \mathbf{B} \quad \mathbf{A} \quad 16 \end{array}$$

Latihan 1

- Tukar no. Decimal kepada no. Binary, (a) 33_{10} (b) 78_{10} (c) 101_{10}
- (a) 33_{10} (b) 78_{10} (c) 101_{10}
- Tukar no. Binary kepada no. Decimal, (a) 111101_2 (b) 011010110_2 (c) 0101_2
- (a) 111101_2 (b) 011010110_2 (c) 0101_2
- Tukar no. pecahan kepada no. Binary, (a) 23.25_{10} (b) 68.75_{10}
- (a) 23.25_{10} (b) 68.75_{10}

Latihan 2

- Tukarkan 61_8 ke nilai perpuluhan.
- Tukarkan 83_{10} ke nilai perlapanan.
- Tukarkan 24.6_8 ke nilai perpuluhan.
- Tukarkan 250_{10} ke nilai perlapanan.
- Tukarkan nombor perduaan berikut ke nilai perpuluhan:
 - (a) 001100_2
 - (b) 000011_2
 - (c) 011100_2
 - (d) 111100_2
 - (e) 11100.011_2
- Tukarkan nombor perpuluhan berikut ke nilai perduaan :
 - (a) 64
 - (b) 500
 - (c) 34.75
 - (d) 25.25
 - (e) 27.1875

GET LOGIK

Pengenalan Get Logik

- Logik dalam sistem Binari digunakan untuk menyatakan proses dan operasi sesuatu maklumat Binari dari segi penyataan matematik. Logik Binari ialah pembolehubah Binaiy dan **operasi logik**.
- Dalam dunia sebenar, sistem logik digital diguna untuk memproses maklumat dan mengawal mesin.
- Kegunaan sistem digital terutamanya di dalam bidang pengkomputeran semakin maju tanpa menunjukkan tanda-tanda penurunan.
- Ini adalah kerana dua sebab utama iaitu pertamanya ialah kerana maklumat lebih mudah dipersembahkan dan dimanipulasikan dalam **bentuk perduaan**.
- Ia juga mudah disimpan menggunakan get-get logik dalam litar-litar elektronik yang mudah.
- Keduanya ialah kerana kebolehan memfabrikasikan litar terkamil di atas cip-cip silikon yang mengandungi ratusan malahan ribuan **get-get logik** mudah menjadikan kosnya jauh lebih rendah berbanding litar elektronik biasa.

Get Logik Asas

- Get-get logik adalah elemen asas untuk membentuk suatu litar sistem digital. Get logik mempunyai satu terminal keluaran dan satu atau lebih terminal masukan.
- Keluarannya ialah logik '1' atau logik '0' bergantung kepada keadaan masukan. Sebelum memulakan unit ini, beberapa istilah yang perlu kita tahu dan fahami ialah:-
 - i) **Aras Logik** – keadaan pembolehubah voltan. Julat voltan tersebut diungkapkan samada dengan logik '1' (Tinggi) atau logik '0' (Rendah).
 - ii) **Gelembung** – bermaksud bulatan kecil pada garis masukan atau keluaran pada simbol litar logik yang melambangkan penyongsangan isyarat tertentu. Jika gelombang itu hadir, masukan atau keluaran itu dikatakan menjadi aktif -rendah.
 - iii) **Aktif Tinggi** –masukan atau keluarannya aktif apabila masukan atau keluarannya Tinggi (logik 1)
 - iv) **Aktif Rendah** –masukan atau keluarannya aktif apabila masukan atau keluarannya Rendah (logik 0)

Get Logik dan Jadual Kebenaran

- Sistem penduaan menjadi sistem nombor pilihan dalam litar elektronik dan sistem perkomputeran. Ini adalah kerana hanya dua simbol atau digit digunakan, iaitu "1" dan "0".
- Dua bit ini dapat mewakili dua paras voltan yang terhasil dalam litar elektronik. Paras 5 V diwakili oleh bit "1" manakala paras 0 V diwakili oleh bit "0".
- Bit "1" dan "0" juga boleh mewakili mana- mana dua keadaan (state) yang dihasilkan oleh sesuatu sistem seperti yang ditunjukkan dalam jadual

Sistem	Perwakilan "0"	Perwakilan "0"
Pernyataan berlogik	Salah / Palsu	Betul / Benar
Pernyataan kepastian	Tidak	Ya
Pensuisan	Tutup	Buka
Aras Voltan	Tinggi / 5 V	Rendah / 0 V
Lampu	Tidak menyala	Menyala

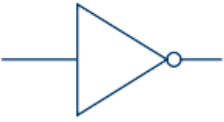
Perwakilan Nombor Penduaan

- Konsep perwakilan dalam jadual di atas disebut sebagai logik positif. Sekiranya perwakilan "1" dan "0" diterbalikkan, iaitu "1" mewakili 0 V dan "0" mewakili 5 V, maka perwakilan ini disebut logik negatif.
- Bagaimanakah litar elektronik beroperasi hanya dengan menggunakan logik "1" dan "0"?
- Beberapa teorem dan peraturan turut diperkenalkan yang dikenali sebagai Teorem Boole (Boolean Theorem) atau Peraturan Boole (Boolean Rules).

- Dalam litar elektronik, elemen yang dapat melaksanakan operasi-operasi logik ini dikenali sebagai get. Get memerlukan satu atau lebih input tetapi hanya menghasilkan satu output sahaja.
- Terdapat tiga get asas: get TAK (**NOT gate** / Inverter), get ATAU (**OR gate**) dan get DAN (**AND gate**). Daripada tiga get asas ini, beberapa get lain yang mempunyai operasi tertentu dapat dihasilkan.
- Antaranya get TAK-ATAU (**NOR gate**), get TAK-DAN (**NAND gate**), get eksklusif ATAU (**XOR gate**) dan get eksklusif TAK-ATAU (**XNOR gate**).
- Setiap get mempunyai jadual kebenaran (truth table). Jadual kebenaran ialah satu senarai semua input yang mungkin dan output yang sepadan. Rajah berikut menunjukkan simbol, jadual kebenaran, pernyataan Boolean dan nombor cip untuk setiap jenis get.

Get TAK

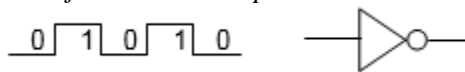
- Get TAK dikenali juga sebagai penyongsang atau 'INVERTER'. Get TAK mempunyai hanya satu masukan dan satu keluaran. A digunakan untuk komplimen atau alihan isyarat digital. Jika masukan '0' keluaran akan dialihkan ke '1' dan sebaliknya.

Simbol			
Fungsi		A	C
Jadual benar (truth table)		0	1
		1	0
$C = \bar{A}$ @ $C = \text{NOT } A$			

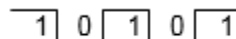
Simbol dan jadual kebenaran get TAK

Contoh 1:

Tentukan bentuk keluaran get TAK jika masukan seperti di bawah:

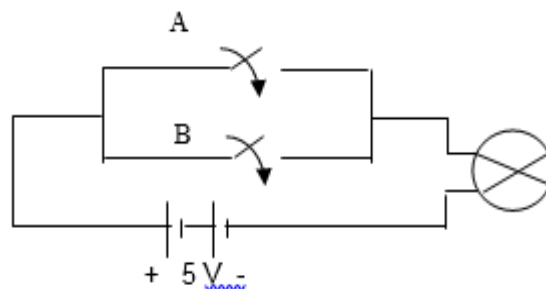


Penyelesaian:




Get ATAU

- Get ATAU boleh digambarkan dengan litar skematik seperti pada rajah berikut di mana suis A dan B disambungkan secara selari. Lampu Y akan menyala hanya apabila suis A atau suis B di'ON'kan atau kedua-dua suis di'ON'kan.



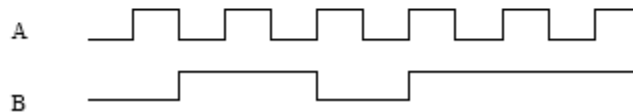
Litar skematik get logik ATAU

Simbol				
Fungsi		A	B	C
Jadual benar (truth table)		0	0	0
		0	1	1
		1	0	1
		1	1	1
$C = A+B @ C = A \text{ OR } B$				

Simbol dan jadual kebenaran get ATAU

Contoh 2:

Apakah jadual kebenaran untuk keluaran Y bagi get logik di bawah jika masukan A dan B seperti yang ditunjukkan?

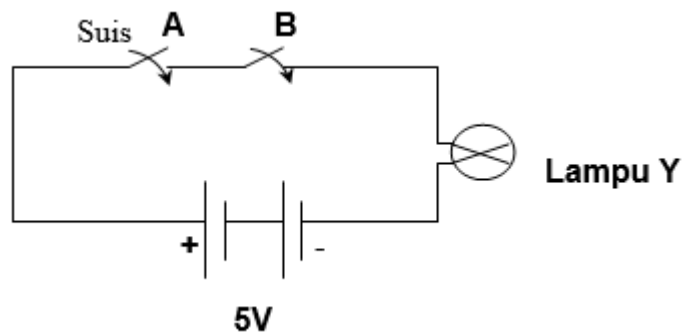


Penyelesaian:


A	B	Y
0	0	0
1	0	1
0	1	1
1	1	1
0	1	1
1	0	1
0	0	0
1	1	1

Get DAN

- Litar skematik berikut menunjukkan idea asas get DAN. Lampu Y akan menyala hanya apabila kedua-dua suis A dan B di'ON'kan.



Litar skematik get logik ATAU

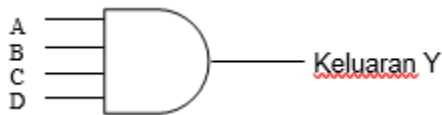
Simbol				
Fungsi		A	B	C
Jadual benar (truth table)		0	0	0
		0	1	0
		1	0	0
		1	1	1
$C = A \cdot B @ C = A \text{ AND } B$				

Simbol dan jadual kebenaran get DAN

Contoh 3:

Lukiskan simbol logik untuk get DAN 4 masukan beserta jadual kebenarannya.

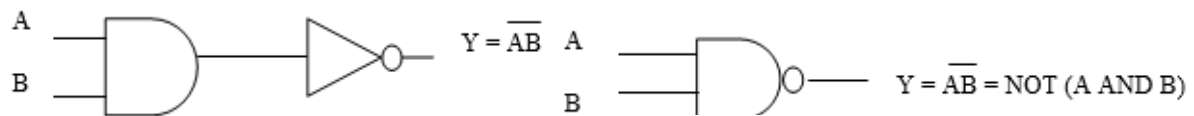
Penyelesaian:

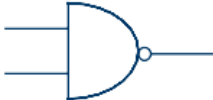


A	B	C	D	Y
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Get TAKDAN

- Get TAK DAN berfungsi sebagai get TAK dan get DAN atau dikenali sebagai get DAN diikuti oleh get TAK. Operasi get TAK DAN sama seperti get DAN kecuali keluarannya adalah terbalik

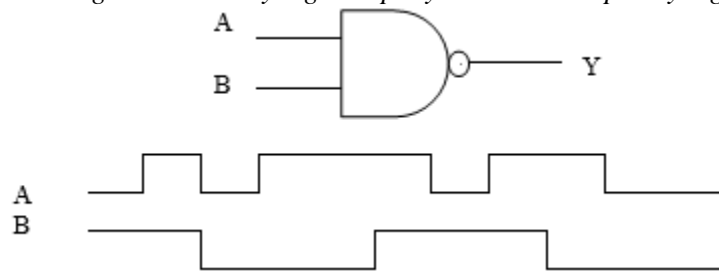


Simbol				
Fungsi		A	B	C
Jadual benar (truth table)		0	0	1
		0	1	1
		1	0	1
		1	1	0
$C = A \cdot B @ C = A \text{ NAND } B$				

Simbol dan jadual kebenaran get TAKDAN

Contoh 4:

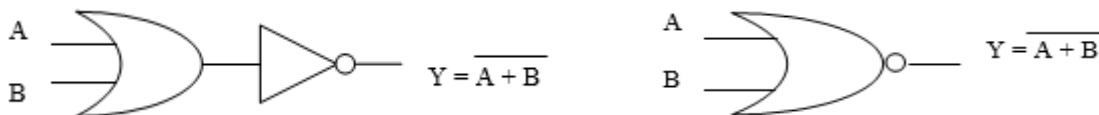
Tentukan bentuk keluaran get TAK DAN yang mempunyai masukan seperti yang ditunjukkan.


**Penyelesaian:**

A	B	Y
0	1	1
1	1	0
0	0	1
1	0	1
1	0	1
1	1	0
0	1	1
1	1	1
1	0	0
0	0	1

Get TAKATAU

- Get TAK ATAU adalah gabungan get ATAU dan get TAK. Operasi bagi get TAK ATAU adalah sama dengan get ATAU kecuali keluarannya adalah terbalik.

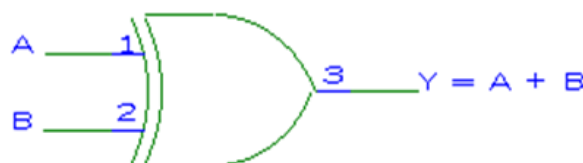



Simbol				
Fungsi		A	B	C
Jadual benar (truth table)		0	0	1
		0	1	0
		1	0	0
		1	1	0
$C = \overline{A+B}$ @ $C = A \text{ NOR } B$				

Simbol dan jadual kebenaran get TAKATAU

Get EKSLUSIF ATAU

- Get EX-ATAU menghasilkan keluaran tinggi jika hanya salah satu masukannya tinggi. Simbol serta jadual kebenarannya adalah seperti yang di bawah.




Simbol				
Fungsi		A	B	C
Jadual benar (truth table)		0	0	0
		0	1	1
		1	0	1
		1	1	0
$C = A \cdot \bar{B} + \bar{A} \cdot B = A \oplus B$ @ $C = A \text{ EX-OR } B$				

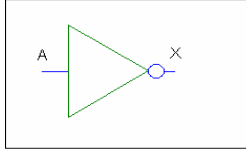
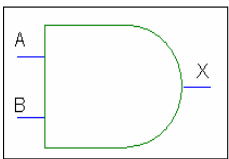
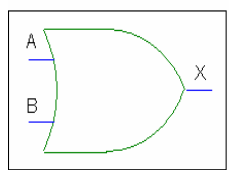
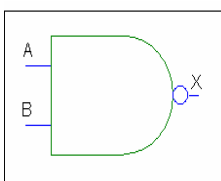
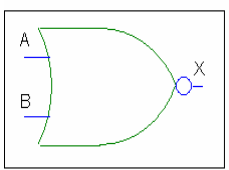
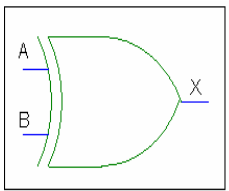
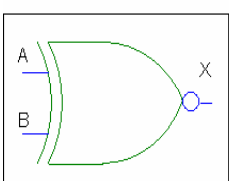
Simbol dan jadual kebenaran get EKSLUSIF ATAU

Get EKSLUSIF TAKATAU

- Get EX-ATAU menghasilkan keluaran bertentangan atau songsang daripada get EKSLUSIF ATAU. Simbol serta jadual kebenarannya adalah seperti yang di bawah.

Simbol				
Fungsi		A	B	C
Jadual benar (truth table)		0	0	1
		0	1	0
		1	0	0
		1	1	1
$C = \bar{A} \cdot \bar{B} + A \cdot B = A \oplus B$ @ $C = A \text{ EX-NOR } B$				

Simbol dan jadual kebenaran get EKSLUSIF TAKATAU

	Simbol logik	Pernyataan Boolean	Jadual Kebenaran															
TAK (Inverter) No. cip: 7404		$X = A$	<table><tr><th>A</th><th>X</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	A	X	0	1	1	0									
A	X																	
0	1																	
1	0																	
DAN (AND) No. cip: 7408		$X = A.B$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	0	0	1	0	1	0	0	1	1	1
A	B	X																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
ATAU (OR) No. cip: 7432		$X = A + B$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	1
A	B	X																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
TAK-DAN (NAND) No. cip: 7400		$X = A . B$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	X	0	0	1	0	1	1	1	0	1	1	1	0
A	B	X																
0	0	1																
0	1	1																
1	0	1																
1	1	0																
TAK-ATAU (NOR) No. cip: 7402		$X = \overline{A + B}$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	X	0	0	1	0	1	0	1	0	0	1	1	0
A	B	X																
0	0	1																
0	1	0																
1	0	0																
1	1	0																
X-ATAU (XOR) No. cip: 7486		$X = \overline{A}B + A\overline{B}$ $= A \oplus B$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	A	B	X	0	0	0	0	1	1	1	0	1	1	1	0
A	B	X																
0	0	0																
0	1	1																
1	0	1																
1	1	0																
X- TAK-ATAU (XNOR)		$X = \overline{A \oplus B}$ $= \overline{A \oplus B}$	<table><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	A	B	X	0	0	1	0	1	0	1	0	0	1	1	1
A	B	X																
0	0	1																
0	1	0																
1	0	0																
1	1	1																

Simbol, Pernyataan Boolean dan Jadual Kebenaran get logik

Litar Logik Gabungan

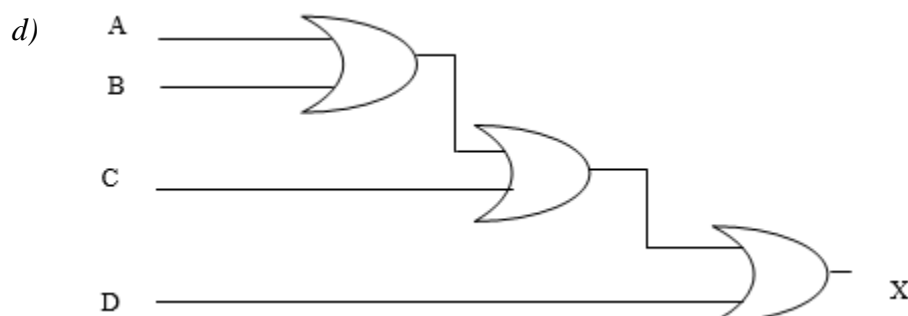
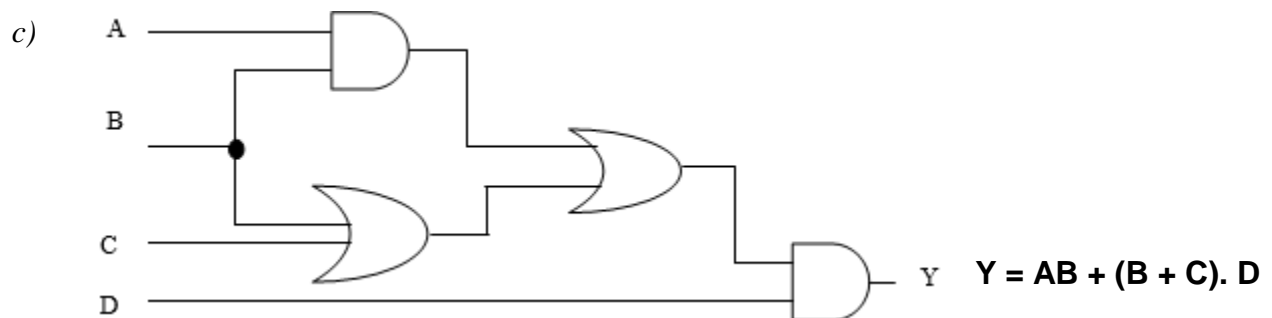
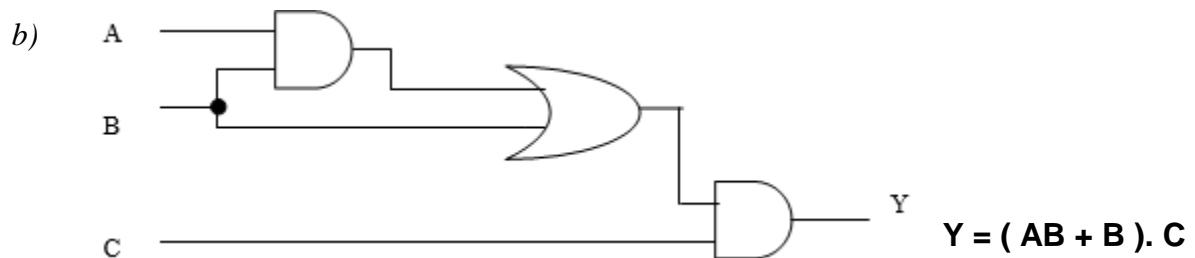
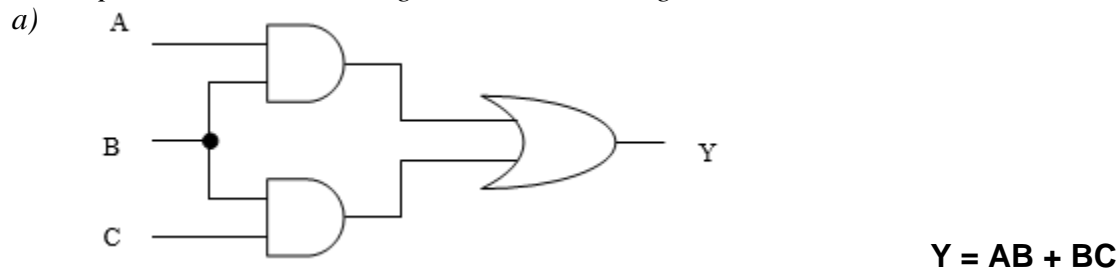
- Litar logik gabungan bermaksud gabungan beberapa get-get logik seperti get DAN, TAK DAN dan TAK di dalam satu litar.
- Secara umum kita boleh katakan litar logik gabungan ialah salah satu jenis atau cabang litar sistem digital di mana pada masa tertentu.
- Keadaan keluaran bergantung pada keadaan masukan hanya pada masa itu sahaja dan tidak pada masukan pada masa sebelumnya.

Rekabentuk Logik Gabungan

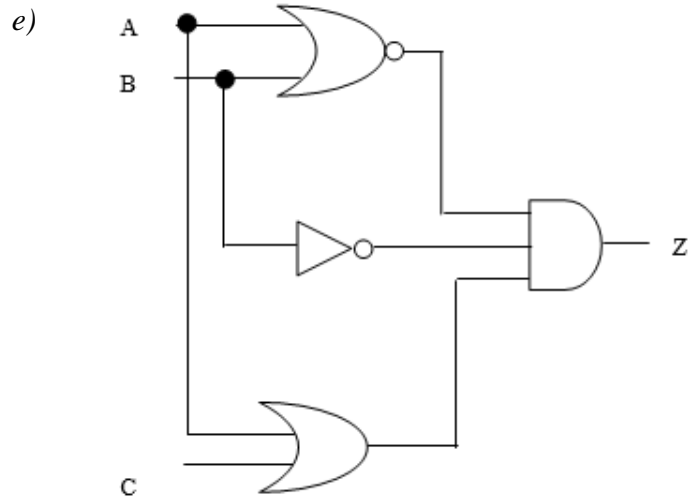
- Untuk memudahkan rekabentuk litar kombinasi logik, 2 perkara asas persamaan logik perlu diketahui iaitu:
 - a) Apabila ada tanda '+' di antara dua atau lebih pembolehubah, contohnya $A + B + C$, ini bermakna semua pembolehubah **diATAUkan**.
 - b) Apabila ada tanda '.' di antara dua atau lebih pembolehubah, contohnya $A . B . C$, ini bermakna semua pembolehubah **diDANkan**.

Contoh:

1. Tuliskan persamaan Boolean bagi kombinasi litar logik di bawah:



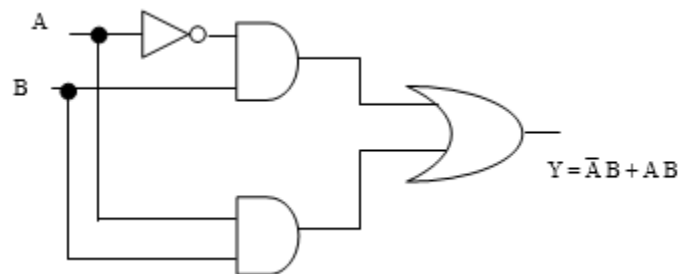
$$X = A + B + C + D$$



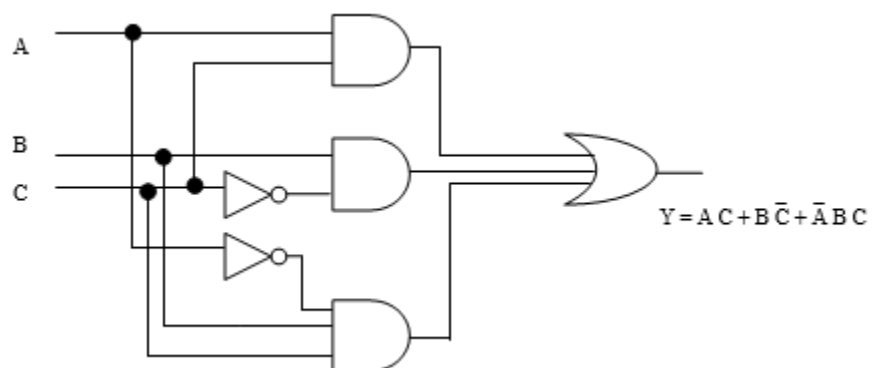
$$Z = A + B . B . (A + C)$$

2. Diberi fungsi kombinasi logik, lukiskan rajah skematik logik gabungan

a) $Y = A . B + \bar{A} . B$



b) $Y = A . \bar{C} + B . C + A . B . C$



OPERASI BOOLEAN ALGEBRA

Pengenalan Algebra Boolean

- Teorem Algebra Boolean dan Teorem De Morgan adalah salah satu cara untuk mempermudah rangkaian ekspresi dan logik.
- Terdapat beberapa teorem seperti yang dinyatakan di bawah. Perhatikan bahawa dalam setiap teorem, x buat pemboleh ubah logik yang mungkin logik 0 atau 1.
- Setiap teorem mengambil bahagian dengan litar logik angka menunjukkan kesahihannya.

1) Commutative law	$A + B = B + A$	$A.B = B.A$
2) Associative law	$A.(B.C) = (A.B).C$	$A + (B.C) = (A + B).C$
3) Distribution law	$A.(B.C) = (A.B).(A.C)$	$A.(B + C) = (A.B) + (A.C)$
4) OR – AND law	$A + 0 = A$ $A + A = A$ $A + 1 = 1$ $A + \bar{A} = 1$	$A.1 = A$ $A.A = A$ $A.0 = 0$ $A.\bar{A} = 0$
5) De Morgan Theorem	$\overline{A.B} = \bar{A} + \bar{B}$	$\overline{A + B} = \bar{A} . \bar{B}$

TEOREM BOOLEAN

- Seorang pakar matematik, George Boolean telah mencadangkan konsep algebra digunakan dalam penyelesaian masalah yang berkaitan dengan logik.
- Beberapa operasi logik telah diperkenalkan seperti operasi TAK,ATAU dan DAN.
- Beberapa teorem dan peraturan turut diperkenalkan yang dikenali sebagai Teorem Boolean (Boolean Theorem) atau Peraturan Boole (Boolean Rules).
- Sistem matematik dua logic ini dipanggil sebagai “ Aljabar Boolean” (Boolean Algebra).

$$X + XY = X$$

$$X (X + Y) = X$$

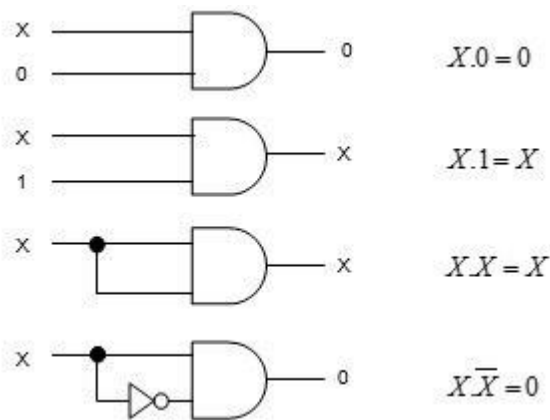
$$\overline{X} + XY = \overline{X} + Y$$

$$X + \overline{X}Y = X + Y$$

$$X (\overline{X} + Y) = XY$$

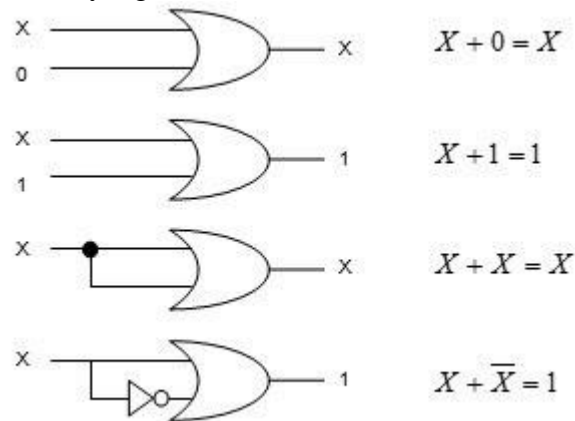
Hukum DAN

- Hukum ini menyatakan jika pembolehubah adalah DAN dengan 0, hasilnya akan menjadi 0.
- Ini dapat diingat dengan mudah kerana operan DAN sebagai pendaraban biasa, di mana bilangan apa pun apabila didarab dengan 0, hasilnya adalah 0.



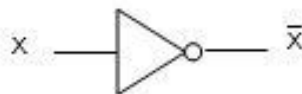
Hukum ATAU

- Hukum ATAU adalah aditif di mana output ATAU gerbang menjadi 1 apabila salah satu dari inputnya adalah 1 tanpa mengira nilai masuk yang lain.



Hukum TAK

- Hukum TAK atau SONGSANG menunjukkan bahawa outputnya terbalik daripada input. Sekiranya input X adalah 1 maka outputnya akan menjadi 0 dan itu 'sebaliknya.



Hukum KOMUTATIF

- Teorem ini melibatkan lebih daripada satu pemboleh ubah. Hukum komutatif menunjukkan bahawa pemboleh ubah ATAU atau DAN keduanya tidak penting, hasilnya sama.

$$X + Y = Y + X$$

$$X \cdot Y = Y \cdot X$$

Hukum SEKUTUAN

- Dengan menggunakan hukum ini kita dapat mengelompokkan pemboleh ubah dalam ungkapan DAN atau ATAU.

$$X (YZ) = (XY)Z = XYZ$$

$$X + (Y + Z) = (X + Y) + Z = X + Y + Z$$

Hukum SEBARAN

- Hukum ini menyatakan bahawa sesuatu ekspresi dapat dikembangkan dengan mengalikan sebutan untuk sebutan. Teorem ini juga menunjukkan kepada kita bagaimana memfaktorkan sesuatu ekspresi.

$$X(Y + Z) = XY + XZ$$

$$(W + X)(Y + Z) = WY + XY + WZ + XZ$$

TEOREM DE MORGAN

- Hukum **De Morgan** menyatakan bahawa komplement dari hasil penjumlahan akan sama dengan hasil perkalian dari masing masing komplement.

$$\overline{X + Y} = \overline{X} \cdot \overline{Y}$$

$$\overline{X \cdot Y} = \overline{X} + \overline{Y}$$

- Teorem pertama De Morgan menyatakan bahawa dua (atau lebih) pembolehubah TAKATAU bersama adalah sama dengan dua pemboleh ubah terbalik (Pelengkap) dan di'DAN'kan.
- Teorem kedua pula menyatakan bahawa dua (atau lebih) pembolehubah bersama TAKDAN adalah sama sebagai dua istilah terbalik (Pelengkap) dan di'ATAU'KAN.

Persamaan SOP dan POS

- Pernyataan atau formula Boolean boleh dihasilkan dengan menggabungkan pembolehubah (*variable*) dan pemalar (*constant*) melalui operasi logik.
- Pernyataan ini kemudiannya digunakan untuk menerangkan sesuatu fungsi. Fungsi ini disebut sebagai **fungsi Boolean** (*Boolean function*).
- Sebagai contoh, pernyataan $x(y + wz)$ digunakan untuk menerangkan fungsi f . Maka fungsi ini boleh ditulis sebagai:

$$f(w, x, y, z) = x(y + wz) \quad \text{atau} \quad f = x(y + wz)$$
- Formula Boolean yang diperolehi daripada jadual kebenaran mempunyai dua struktur atau bentuk: SOP (Sum-Of-Products) dan POS (Product-Of-Sums).

Formula SOP

- Beberapa istilah penting mesti diketahui dalam formula SOP. Lihat persamaan di bawah sebagai contoh.

$$F(w, x, y, z) = w + x y z + w \overline{x} y z \quad \text{(formula SOP biasa)}$$
- Persamaan tersebut mempunyai empat pembolehubah (w, x, y, z) dan enam literal ($w, \overline{w}, x, y, z, \overline{z}$).
- Literal-literal ini membentuk tiga sebutan ($w, xyz, wxyz$). Perhatikan bahawa sebutan-sebutan ini adalah **hasil darab** (*product term*) antara beberapa literal.
- Ketiga-tiga sebutan ini kemudian ditambahkan menghasilkan persamaan SOP. Perhatikan juga bahawa sebutan ketiga mempunyai kesemua pembolehubah fungsi F .
- Sebutan ini dipanggil sebagai **sebutan minimum** (*minterm*). Sekiranya terdapat persamaan yang kesemua sebutannya ialah sebutan minimum, maka persamaan tersebut dikenali sebagai **persamaan SOP piawai** (*canonical/ standard SOP expression*).
- Contoh persamaan piawai SOP ialah seperti di bawah.

$$h(a, b, c) = a b c + a b \overline{c} + a \overline{b} c \quad \text{(formula SOP piawai)}$$

Ingat ! Persamaan SOP akan menghasilkan "1" apabila salah satu sebutannya bernilai "1".

Latihan:

Lengkapkan jadual di bawah dengan mencari sebutan minimum untuk setiap kombinasi input.

Perpuluhan	a	b	c	Sebutan minimum (m)
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

Sebutan minimum untuk tiga pembolehubah

Untuk memudahkan penulisan, fungsi h di atas boleh ditulis dalam bentuk notasi menggunakan simbol atau digit perpuluhan (**decimal notation**) seperti yang ditunjukkan di bawah. Simbol “ $\square m$ ” menunjukkan fungsi tersebut adalah hasil tambah sebutan minimum yang pertama, ketiga dan keenam.

$$h(a, b, c) = a b c + a b c + a b c = \square m(1, 3, 6) \quad (\text{notasi-m})$$

Formula POS

- Persamaan POS di bawah mempunyai empat pembolehubah dan tujuh literal. Penambahan beberapa literal menghasilkan sebutan tambah (**sum term**).
- Apabila sebutan-sebutan ini didarabkan, ia menghasilkan persamaan POS. Perhatikan bahawa sebutan ketiga mempunyai kesemua pembolehubah fungsi f , oleh itu sebutan ini dikenali sebagai sebutan maksimum (**maxterm**).

$$F(w, x, y, z) = (x + y)(x + y + z)(w + x + y + z) \quad (\text{formula POS biasa})$$

Ingat! Persamaan POS akan menghasilkan “0” apabila salah satu sebutannya bernilai “0”.

- Persamaan POS di bawah mempunyai empat pembolehubah dan tujuh literal. Penambahan beberapa literal menghasilkan sebutan tambah (**sum term**).

Apabila sebutan-sebutan ini didarabkan, ia menghasilkan persamaan POS. Perhatikan bahawa sebutan ketiga

Persamaan yang kesemua sebutannya adalah sebutan maksimum dikenali sebagai **persamaan piawai POS** (**canonical/standard POS expression**), seperti yang ditunjukkan di bawah. Persamaan POS juga boleh ditulis dalam bentuk notasi perpuluhan. Simbol “ ΠM ” menunjukkan fungsi tersebut adalah hasil darab sebutan maksimum yang ketiga, kelima dan ketujuh.

$$G(a, b, c, d) = (a + b + c + d)(a + b + c + d)(a + b + c + d) \quad (\text{formula POS piawai})$$

$$= \Pi M(3, 5, 12) \quad (\text{notasi-M})$$

Latihan:

Lengkapkan jadual di bawah dengan mencari sebutan maksimum untuk setiap kombinasi input.

Perpuluhan	a	b	c	Sebutan maksimum (M)
0	0	0	0	
1	0	0	1	
2	0	1	0	
3	0	1	1	
4	1	0	0	
5	1	0	1	
6	1	1	0	
7	1	1	1	

Sebutan maksimum untuk tiga pembolehubah

Penentuan dan Pertukaran Persamaan

- Beberapa istilah penting mesti diketahui dalam formula SOP. Lihat persamaan di bawah sebagai contoh.
$$F(w, x, y, z) = w + x y z + w x y z$$

(formula SOP biasa)
- Persamaan tersebut mempunyai empat pembolehubah (w, x, y, z) dan enam literal ($w, \bar{w}, x, \bar{x}, y, \bar{y}, z, \bar{z}$).
- Literal-literal ini membentuk tiga sebutan ($w, xyz, wxyz$). Perhatikan bahawa sebutan-sebutan ini adalah **hasil darab (product term)** antara beberapa literal.
- Sebelum ini, persamaan untuk sesuatu fungsi didapati daripada rajah logik gabungan. Tahukah pula cara untuk mendapatkan persamaan daripada sesuatu jadual kebenaran?
- Untuk menulis persamaan piawai SOP, semua kombinasi input yang menghasilkan output bernilai “1” diambil dan ditulis dalam bentuk sebutan minimum.
- Untuk menulis persamaan piawai POS, semua kombinasi input yang menghasilkan output “0” diambil dan ditulis dalam bentuk sebutan maksimum.

Contoh: Dapatkan persamaan SOP dan POS piawai untuk fungsi Y dalam bentuk senarai perpuluhan dan pembolehubah bagi setiap jadual kebenaran berikut.

a)

X2	X1	X0	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

b)

p	q	r	Y
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	1

Anda juga boleh tukar persamaan SOP ke bentuk POS dan begitu juga sebaliknya kerana kedua-duanya adalah pelengkap di antara satu sama lain.

Rajah Logik Persamaan SOP dan POS

- Untuk menulis persamaan piawai POS, semua kombinasi input yang menghasilkan output “0” diambil dan ditulis
- Rajah logik persamaan SOP adalah berstruktur **DAN-ATAU** manakala rajah logik persamaan POS berstruktur **ATAU-DAN**.
- Rajah-rajah ini biasanya mempunyai dua tingkat/aras tetapi sekiranya get TAK dilukis sekali dalam rajah, maka bilangan tingkat menjadi tiga.

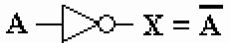
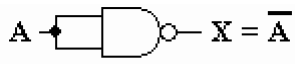
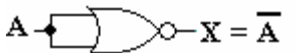
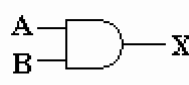
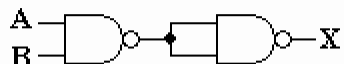
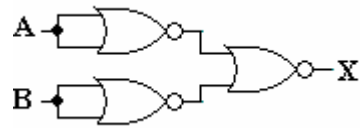

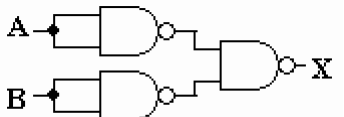
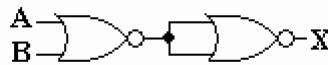

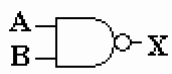
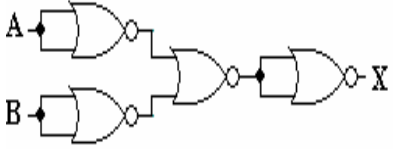
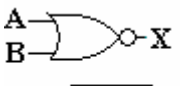
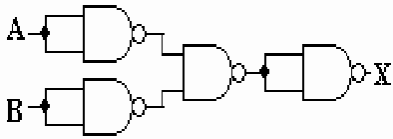

Contoh: Lukis rajah logik dua tingkat untuk persamaan-persamaan di bawah.

a) $H(p, q, r) = pr + pq + pqr$

b) $F(a, b, c) = (a + b)(b + c)(a + b + c)$

Pelaksanaan Rajah Logik Satu Get – NAND dan NOR

- Untuk menulis persamaan piawai POS, semua kombinasi input yang menghasilkan output “0” diambil dan ditulis
- Rajah logik biasanya mempunyai lebih daripada satu get. Untuk mengurangkan penggunaan cip dan seterusnya mengurangkan kos, adalah lebih baik untuk menghasilkan rajah logik yang mempunyai satu get sahaja.
- Get yang dapat dimanipulasi untuk menghasilkan operasi TAK, DAN, ATAU, TAK-ATAU dan TAK-DAN dikenali sebagai get universal.
- Terdapat dua get universal iaitu get TAK-DAN (NAND) dan TAK-ATAU (NOR). Jadual menunjukkan bagaimana dua jenis get ini digunakan untuk menghasilkan operasi yang lain.

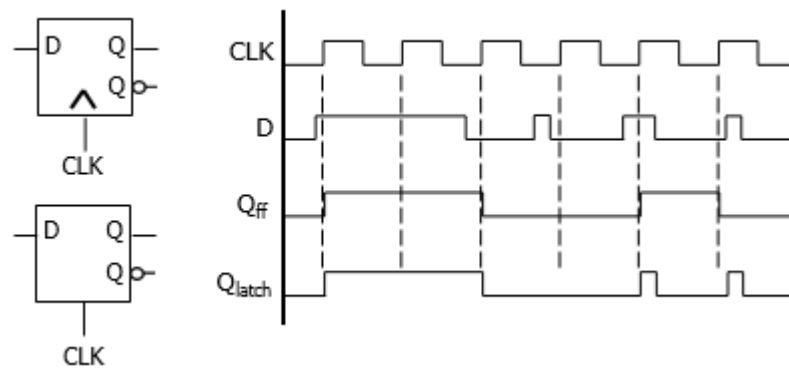
Operasi logik	Pelaksanaan NAND	Pelaksanaan NOR
 $A \rightarrow \neg \rightarrow X = \overline{A}$	 $A \rightarrow \text{NAND} \rightarrow X = \overline{A}$	 $A \rightarrow \text{NOR} \rightarrow X = \overline{A}$
 $X = A \cdot B$		
 $X = A + B$		
 $X = \overline{A \cdot B}$		
 $X = \overline{A + B}$		

Pertukaran get lain ke get TAKDAN dan TAKATAU

LITAR BERJUJUKAN

Pengenalan Selak dan Flip-flop

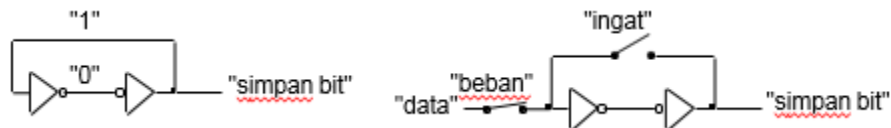
- Selak dan flip-flop adalah elemen memori bit tunggal yang digunakan untuk membina litar berurutan dengan satu atau dua masukan atau keluaran, yang dirangka menggunakan get logik individu dan gelung suapan.
- Output selak bergantung pada masukan semasa dan keluaran sebelumnya dan perubahan keadaan boleh terjadi pada bila-bila masa apabila masukan berubah.
- Flip-flop pula adalah litar logik berjujukan di mana ia mempunyai ciri-ciri ingatan. Dengan itu ia berupaya menyimpan data binari yang telah dimasukkan.
- Litar ingatan ini juga dikenali sebagai flip-flop kerana ia boleh 'flipped' kepada keadaan set iaitu menyimpan binari 1 atau ia 'flopped' kepada keadaan reset iaitu menyimpan binari 0.
- Keluaran flip-flop bergantung pada masukan semasa dan keluaran sebelumnya, tetapi perubahan keadaan berlaku pada masa-masa tertentu mengikut masukan yang ditentukan oleh jam
- Litar kecil inilah yang menjadi asas kepada sistem ingatan di dalam komputer peribadi anda. Tingkah laku selak dan flip-flop adalah hampir sama namun berbeza bagi flip-flop pada masukan jam tinggi (HIGH).



Perbandingan isyarat bagi selak dan flip-flop

Selak

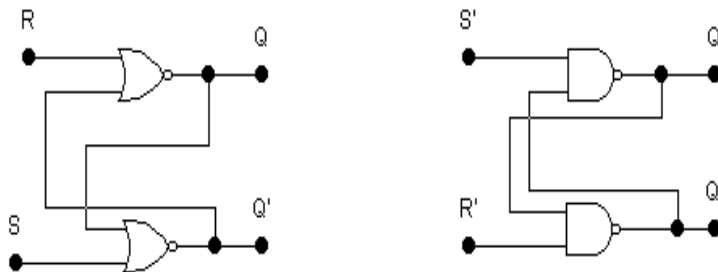
- Keluaran mana-mana litar logik gabungan bergantung secara langsung kepada masukan
- Secara amnya, dalam litar logik berurutan, keluaran bergantung bukan hanya pada masukan tetapi juga pada keadaan yang tersimpan
- Selak digunakan untuk penyimpanan sementara data bit flip-flop menjadi asas bagi kebanyakan jenis logik berurutan seperti daftar dan pembilang.
- Jenis litar pemasaan: pemasa 555
- Selak adalah jenis peranti storan sementara yang mempunyai dua keadaan stabil.
- Sama seperti flip-flop, keluaran disambungkan kembali ke masukan bertentangan.
- Perbezaan utama dengan flip-flop adalah kaedah yang diguna untuk mengubah keadaan.
- Jenis selak: Selak S-R dan selak D.
- Bagaimana menghasilkan selak:
 - (i) Dua penyongsang memegang bit – selagi kuasa disalurkan
 - (ii) Menyimpan memori baru – hentikan laluan suapan sementara waktu



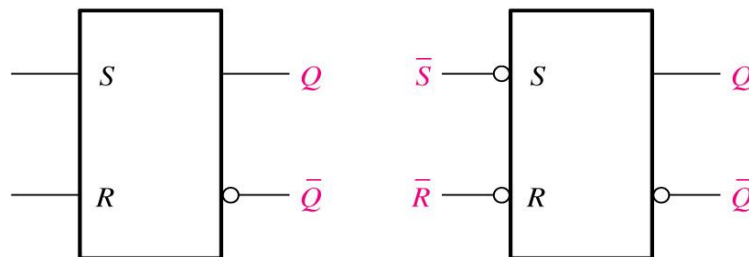
Kaedah menghasilkan selak

Selak S-R (Set-Reset)

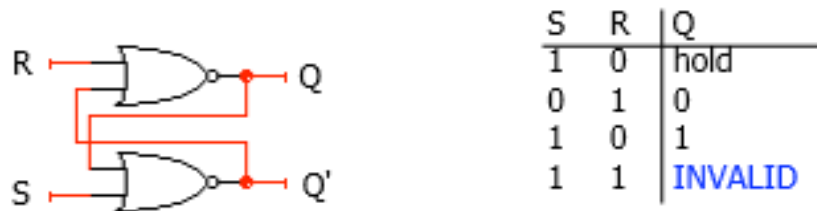
- Selak S-R terbina dari dua jenis iaitu S-R get TAKATAU dan S-R get TAKDAN di mana keluaran get adalah disuapbalik kepada masukan yang bertentangan.
- S-R get TAKATAU boleh set ($S=1, R=0$) atau reset ($R=1, S=0$) bagi keluaran, Selak jenis ini adalah aktif tinggi (Active High)
- S-R get TAKATAU boleh set ($S=1, R=0$) atau reset ($R=1, S=0$) bagi keluaran. Selak jenis ini adalah aktif rendah (Active Low)



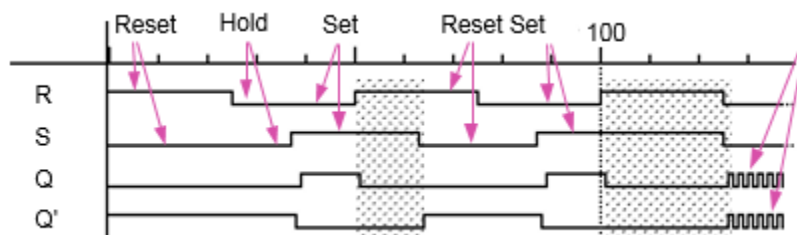
Litar skematik bagi S-R iaitu selak S-R TAKATAU dan S-R TAKDAN

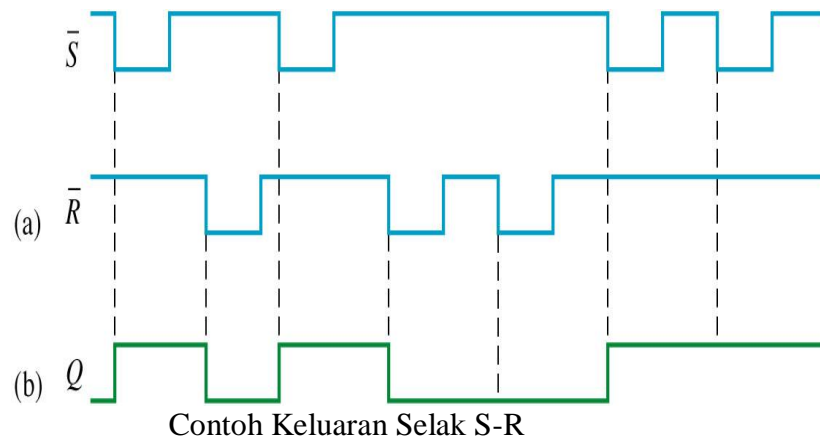


Simbol logik bagi S-R iaitu selak S-R TAKATAU dan S-R TAKDAN



Jadual Kebenaran Selak S-R





- Selak S-R adalah sensitif terhadap kesalahan. Kesalahan pada masukan S akan menetapkan (SET) selak manakala kesalahan pada masukan R akan menetapkan semula (RESET) selak.

