BAB III

ANALISA RANGKAIAN LOGIKA MENGGUNAKAN ALJABAR BOOLE

3.1 MENGURAIKAN RANGKAIAN LOGIKA SECARA ALJABAR

Setiap rangkaian logika, bagaimanapun kompleknya dapat diuraikan secara lengkap dengan menggunakan operasi Boole yang telah dikemukakan pada BAB II. Hal ini dikarenakan ketiga operasi itu, yaitu rangkaian pintu OR, pintu AND dan pintu NOT sesungguhnya merupakan blok bangun dasar sistem digital. Misalnya, perhatikan rangkaian Gambar 3.1, rangkaian ini mempunyai tiga masukan A, B dan C dan sebuah keluaran tunggal, X. Dengan menerapkan operasi Boole yang sesuai untuk tiap pintu dengan mudah dapat ditentukan persamaan keluarannya.

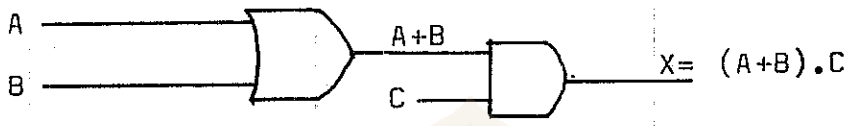
Gambar 3.1 Rangkaian logika dengan persamaan Boolenya.

Persamaan keluaran pintu AND ditulis A.B . Keluaran pintu AND ini dihubungkan ke pintu OR sebagai masukan bersama-sama masukan lainnya, C. Pintu OR mengoperasikan masukannya sehingga keluarannya merupakan operasi OR atas masukannya. Jadi keluaran pintu-OR dapat dinyatakan sebagai X= A.B+C (persamaan ini dapat juga ditulis sebagai X= C+A.B)

Kadang-kadang dapat terjadi kebingungan untuk menetukan operasi mana dalam suatu persamaan yang pertama-tama harus dikerjakan. Pernyataan A.B+C dapat ditafsirkan menurut dua cara yang berbeda : (1) A.B di OR kan dengan C atau (2) A di AND kan dengan (B+C). Untuk menghindari hal ini, maka apabila suatu pernyataan mengandung operasi AND maupun OR, maka operasi AND dikerjakan lebih dahulu, kecuali bila ada tanda kurung didalam pernyataan tersebut, dalam hal ini operasi yang berada didalam tanda kurung ha-

rus dikerjakan lebih dahulu. Aturan seperti ini sama dengan apa - yang diterapkan pada aljabar biasa.

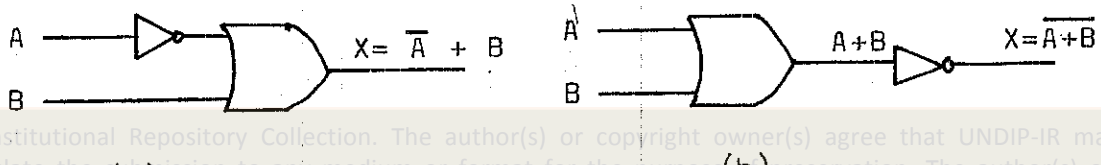
Untuk menjelaskan lebih jauh, perhatikanlah rangkaian Gambar 3.2. Pernyataan untuk keluaran pintu OR adalah A+B. Keluaran- ini berfungsi sebagai masukan untuk pintu AND bersama-sama dengan masukan yang lain, C.



Gambar 3.2 Rangkaian logika yang persamaan keluarannya memerlu - kan tanda kurung.

Jadi keluaran pintu AND dapat dinyatakan sebagai X= (A+B).C . Perhatikanlah penggunaan tanda kurung disini untuk menunjukkan bahwa A dan B di OR kan lebih dahulu, setelah itu hasilnya baru di AND- kan dengan C. Tanpa tanda kurung ini dapat terjadi penafsiran - yang tidak benar karena A+B.C berarti A di OR kan dengan hasil - B.C .

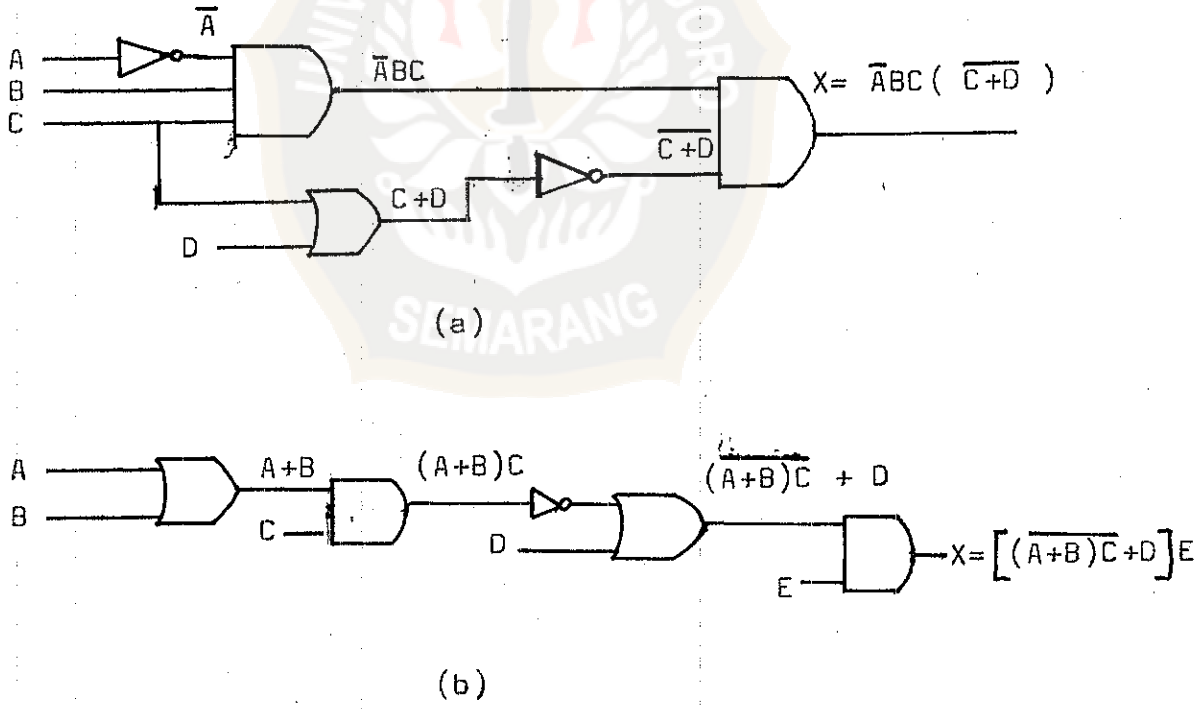
Apabila inverter (rangkaian NOT) terdapat didalam suatu - diagram rangkaian logika, maka pernyataan keluarannya sama dengan pernyataan masukannya ditambah garis atas. Gambar 3.3 menunjukkan dua contoh rangkaian yang menggunakan inverter, oleh karena itu - keluarannya sama dengan A. Keluaran inverter tersebut diberikan - kepada pintu OR bersama-sama dengan B, sehingga keluaran pintu OR sama dengan $\overline{A} + B$.



Gambar 3.3 Rangkaian yang menggunakan inverter.

Perhatikanlah bahwa garis hanya ada diatas A, menunjukkan bahwa A- pertama-tama diinversikan dan kemudian di OR kan dengan B.

Pada Gambar 3.3a keluaran pintu OR sama dengan A+B yang ke mudian diberikan pada sebuah inverter. Oleh karena itu keluaran in verter sama dengan $\overline{(A + B)}$, karena inverter menginversikan keselu ruhan pernyataan masukan. Perhatikanlah bahwa garis menutup selu - ruh pernyataan (A + B). Hal ini penting karena, seperti apa yang - akan ditunjukkan kemudian, pernyataan $\overline{(A + B)}$ dan $(\overline{A} + \overline{B})$ tidak - ekivalen. Bentuk $\overline{(A + B)}$ berarti bahwa A di OR kan dengan B dan - kemudian hasil operasi OR tersebut diinversikan. Sedangkan pernya taan $(\overline{A} + \overline{B})$ menunjukkan bahwa A diinversikan dan B diinversikan- dan hasilnya kemudian di OR kan menjadi satu.



Gambar 3.4 Contoh-contoh lain lagi.

Gambar 3.4 menunjukkan dua contoh lagi yang harus dipelaja ri dengan hati-hati. Khususnya perhatikanlah penggunaan dua set - tanda kurung terpisah dalam gambar 3.4b. Juga perhatikanlah dalam- gambar 3.4a bahwa variabel masukan C dihubungkan sebagai masukan - kepada dua pintu yang berbeda.

3.2 MENGEVALUASI KELUARAN RANGKAIAN LOGIKA

Sekali persamaan keluaran Boole untuk suatu rangkaian tertentu diperoleh, maka tingkat logika keluarannya dapat ditentukan untuk setiap harga pada masukan rengkaian tersebut. Sebagai contoh misalnya diinginkan untuk mengetahui tingkat logika keluaran X rangkaian Gambar 3.4a untuk kasus masukan A=0, B=1, C=1 dan D=0 . Seperti halnya pada aljabar harga X dapat diketemukan dengan memasukkan harga-harga keempat variabel tersebut kedalam persamaan dan mengerjakan operasi yang diminta sebagai berikut :

$$
\begin{aligned}
X &= \bar{A}BC\,(\overline{C + D}) \\
&= \bar{0}.1.1.(\overline{1+0}) \\
&= 1.1.1.(\overline{1+0}) \\
&= 1.1.1.(\bar{1}) \\
&= 1.1.1.0 \\
&= 0
\end{aligned}
$$

Sebagai contoh lain, dapat dicoba menghitung keluaran rangkaian Gambar 3.4b untuk A=0, B=0, C=1, D=1 dan E=1

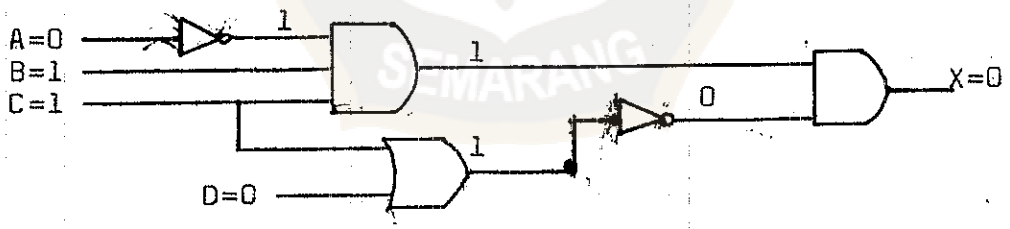
$$
\begin{aligned}
X &= \left[D + \overline{(A+B)C}\right].E \\
&= \left[1 + \overline{(0+0).1}\right].1 \\
&= \left[1 + \overline{0.1}\right].1 \\
&= \left[1 + \bar{0}\right].1 \\
&= \left[1 + 1\right].1 \\
&= 1.1 \\
&= 1
\end{aligned}
$$

Sebagai petunjuk umum, maka aturan main berikut ini harus selalu diikuti apabila hendak menyelesaikan persamaan seperti ditunjukkan pada contoh diatas :

1. Pertama kerjakan semua inversi atas faktor tunggal; yaitu $\bar{0}=1$ atau $\bar{1}=0$.
2. Kemudian kerjakan semua operasi didalam tanda kurung.

3. Kerjakanlah operasi AND sebelum operasi OR kecuali tanda ku - rung menunjukkan lain.
4. Apabila suatu pernyataan mempunyai garis diatasnya, pertama-ta ma kerjakanlah operasi atas pernyataan tersebut dan kemudian - inversikan hasilnya.

Tingkat logika keluaran untuk tingkat masukan tertentu da- pat juga ditentukan secara langsung dari diagram rangkaian tanpa- menggunakan persamaan Boole. Teknik ini sering digunakan oleh pa- ra teknisi selama mencari kerusakan atau mengetest suatu sistem - logika, karena teknik ini juga dapat memberikan gambaran tentang- baik tidaknya setiap keluaran pintu maupun keluaran akhir. Untuk- jelasnya rangkaian gambar 3.4a digambar kembali pada Gambar 3.5 - dengan tingkat masukan A=0, B=1, C=1, dan D=0. Prosedurnya harus dimulai dari masukan dan berlanjut ketiap inverter dan pintu, me- nuliskan tiap-tiap keluarannya sampai keluaran akhir tercapai.



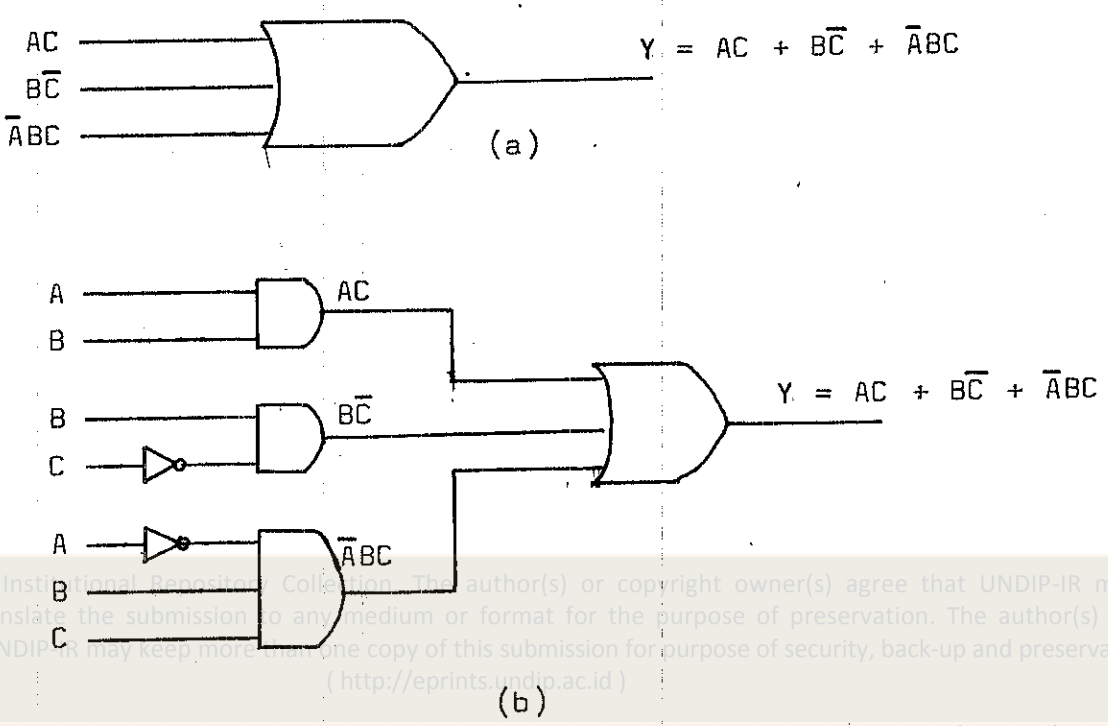
Gambar 3.5 Menentukan tingkat keluaran langsung dari diagram - rangkaian.

Pada Gambar 3.5 pintu AND pertama mempunyai tiga masukan - semuanya pada logika 1, karena inverter mengubah A=0 menjadi 1. Pintu OR mempunyai masukan 1 dan 0 yang menghasilkan keluaran 1 - karena 1+0=1. 1 ini diinversikan menjadi 0 dan diberikan ke pintu AND bersama-sama dengan 1 dari keluaran AND pertama. Masukan 0 - dan 1 ke pintu AND akhir menghasilkan keluaran 0 karena 0.1=0 .

3.3 MENYUSUN RANGKAIAN LOGIKA BERDASARKAN PERSAMAAN BOOLE

Apabila operasi suatu rangkaian didefinisikan oleh persamaan Boole, maka diagram rangkaian logikanya dapat langsung dibuat berdasarkan persamaan tersebut. Misalnya apabila diperlukan suatu rangkaian yang didefinisikan oleh X=A.B.C maka dari persamaan tersebut segera diketahui bahwa yang dibutuhkan adalah sebuah pintu AND tiga masukan. Apabila diperlukan rangkaian yang didefinisikan oleh $X= A + \overline{B}$, maka akan dibutuhkan sebuah pintu OR dua masukan dengan sebuah inverter pada salah satu masukan pintu OR tersebut. Penalaran yang sama yang dikenakan pada kasus yang sederhana ini dapat diperluas untuk rangkaian yang lebih kompleks.

Misalkan diinginkan untuk menyusun rangkaian yang keluarannya adalah $y= AC + B\overline{C} + \overline{A}BC$. Persamaan Boole ini mengandung tiga suku (AC, $B\overline{C}$, $\overline{A}BC$), yang di OR kan menjadi satu. Hal ini berarti bahwa dibutuhkan sebuah pintu OR tiga masukan dengan masing-masing masukan sama dengan AC, BC, dan ABC. Rangkaian pelaksanaannya ditunjukkan pada Gambar 3.6a, pada gambar ini ditunjukkan sebuah pintu OR tiga masukan dengan masukannya yang ditandai dengan AC, $B\overline{C}$, dan $\overline{A}BC$.



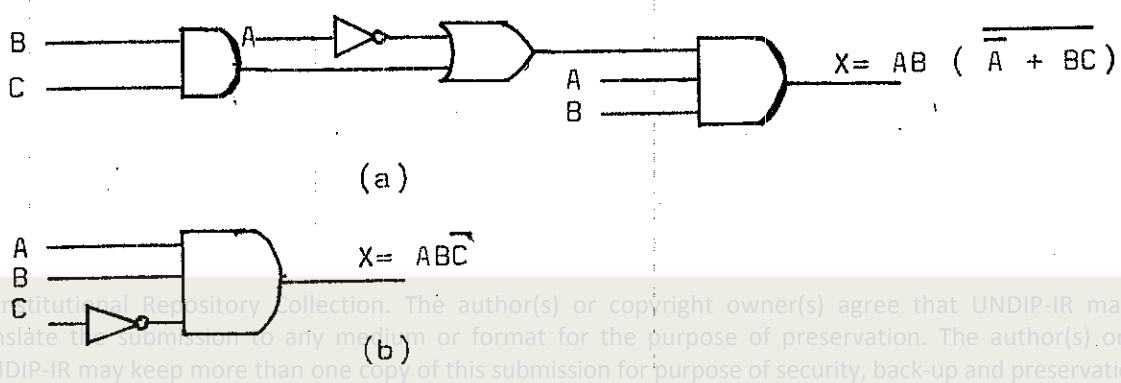
Gambar 3.6 Mengkonstruksikan suatu rangkaian logika dari suatu persamaan Boole.

Tiap-tiap masukan pintu OR merupakan bentuk pernyataan hasil operasi AND, yang berarti bahwa pintu AND dengan masukan yang sesuai dapat digunakan untuk menghasilkan tiap bentuk ini. Hal - ini ditunjukkan pada Gambar 3.6b, yang merupakan diagram rangkaian akhir. Perhatikanlah penggunaan inverter untuk menghasilkan - faktor A dan C yang diperlukan dalam pernyataan.

3.4 MENYEDERHANAKAN RANGKAIAN LOGIKA

Telah ditunjukkan bahwa aljabar Boole dapat digunakan untuk menganalisa suatu rangkaian logika dan menyatakan operasinya secara matematik. Hal ini memungkinkan pemakaian aljabar Boole - yang paling penting, yaitu penyerdehanaan rangkaian logika. Sekali persamaan Boole untuk suatu rangkaian logika telah diperoleh, maka persamaan tersebut pada umumnya dapat diubah menjadi bentuk-yang lebih sederhana dengan menggunakan teorema Boole tertentu. Persamaan Boole yang lebih sederhana, yang ekivalen dengan persamaan aslinya dapat digunakan untuk menggantikan persamaan aslinya tersebut. Penyederhanaan ini sering dapat menghasilkan penyusutan yang besar dalam jumlah pintu dan hubungan rangkaian logikanya.

Misalkan Gambar 3.7 menunjukkan suatu rangkaian logika dengan bentuk keluarannya $X = AB(\overline{\overline{A} + BC})$ yang diperoleh dengan metode yang telah dijelaskan terdahulu. Dengan menggunakan teorema-Boole persamaan ini dapat disederhanakan menjadi ekivalennya, - $X = AB\overline{C}$, yang dilaksanakan seperti pada Gambar 3.7b . Jelas bahwa

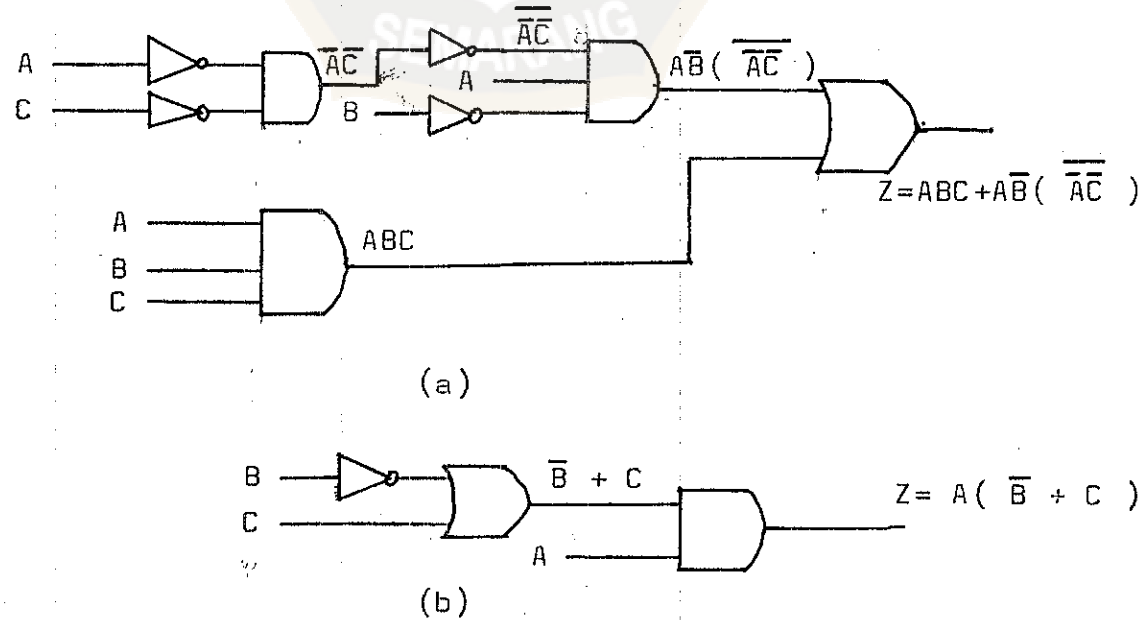


Gambar 3.7

kedua rangkaian itu ekivalen dalam hal operasi logika yang dilaku kannya. Dipandang dari biaya danukuran rangkaiannya, maka jelas - yang lebih sederhana itulah yang dipilih.

Teorema yang telah dikemukakan terdahulu dapat digunakan - untuk menyederhanakan setiap persamaan Boole. Biasanya, apabila di lakukan dengan benar, setiap jalur akan mengantarkan kepada hasil- yang paling sederhana. Kadang-kadang dua jalan yang berbeda dapat- memberikan hasil berbeda yang sama sederhananya. Contoh-contoh be- rikut hendaknya dipelajari dengan seksama untuk memahami sejumlah- penalaran yang digunakan dalam menyederhanakan rangkaian logika. Meskipun tidak ada prosedur tertentu dalam menerapkan teorema ini- , dengan mempelajari contoh-contoh ini dan contoh-contoh terdahulu dan dengan mengerjakan soal latihan kita akan menguasai suatu cara yang cepat untuk sampai pada suatu pemecahan yang baik.

Contoh 3.1 Sederhanakanlah rangkaian logika yang ditunjukkan pada Gambar 3.8



Gambar 3.8 Contoh 3.1

Jawaban : Langkah pertama adalah menentukan persamaan keluarannya. Hasilnya adalah :

$$Z = ABC + A\overline{B}.\ (\overline{\overline{AC}})$$

Sekali persamaannya telah ditentukan, biasanya merupakan langkah-yang baik untuk memecah semua tanda inverter besar dengan menggunakan teorema De Morgan's dan kemudian mengerjakan perkaliannya.

$Z = ABC + A\overline{B}(\overline{\overline{A}} + \overline{\overline{C}})$ Teorema 2.7b

$= ABC + A\overline{B}(A + C)$ Teorema 2.6

$= ABC + A\overline{B}A + A\overline{B}C$ Aksioma B_2

$= ABC + A\overline{B} + A\overline{B}C$ Teorema 2.1b

Apabila persamaan berbentuk seperti ini, maka variabel yang sama-harus dicari diantara berbagai macam suku untuk dikeluarkan sebagai faktor. Suku pertama dan ketiga diatas mempunyai kesamaan dalam variabel AC yang dapat dikeluarkan.

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

karena $B + \overline{B} = 1$ Aksioma B_4a maka

$Z = AC(1) + A\overline{B}$

$= AC + A\overline{B}$

Sekarang A dapat dikeluarkan yang menghasilkan

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

Hasil ini tidak dapat disederhanakan lagi, pelaksanaan - rangkaiannya ditunjukkan pada Gambar 3.8b.

Jelas bahwa rangkaian (b) jauh lebih sederhana daripada rangkaian aslinya (a).

Contoh 3.2 Sederhanakan persamaan $Z = ABC + A\overline{B\overline{C}} + A\overline{B}C$

Jawab : Terlihat ada dua cara yang berbeda untuk sampai pada hasil yang sama.

Metode I : Dua suku pertama dalam persamaan tersebut mempunyai - variabel AB yang sama, jadi :

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

$= AB(1) + A\overline{B}C$

$= AB + A\overline{B}C$

Variabel A dapat dikeluarkan dari kedua suku

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

Dengan menggunakan Teorema 2.8 maka

$Z = A(B + C)$.

Metode II :

Persamaan asalnya adalah $Z = ABC + AB\overline{C} + A\overline{B}C$. Dua suku pertama mempunyai variabel yang sama, suku pertama dan terakhir mempunyai variabel AC yang sama. Bagaimanakah cara mengetahui apa kah mengeluarkan AB dari dua suku pertama atau AC dari dua suku terakhir ? Sesungguhnya dapat dilakukan dua-duanya dengan menggu nakan suku ABC dua kali. Dengan kata lain persamaan tersebut dapat ditulis sebagai :

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

pada persamaan diatas telah ditambahkan suku ekstra ABC. Penamba han ini benar dan tidak akan merubah harga persamaan karena ABC + ABC = ABC (Teorema 2.1a). Sekarang AB dapat dikeluarkan dari dua suku pertama dan AC dari dua suku terakhir.

$$\begin{aligned} Z &= AB(C + \overline{C}) + C(\overline{B} + B) \\ &= AB.1 + AC.1 \\ &= AB + AC \\ &= A(B + C) \end{aligned}$$

Ternyata diperoleh hasil yang sama seperti metode I.

Teknik penggunaan suku yang sama sampai dua kali selalu dapat di-gunakan. Sebenarnya suku yang sama dapat digunakan lebih dari dua kali apabila diperlukan.

Contoh 3.3 Sederhanakan $Z = \overline{A}C(\overline{\overline{A}BD}) + \overline{A}B\overline{C}\overline{D} + A\overline{B}C$

Jawab :

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

menyelesaikan perkalian

$$Z = \bar{A}CA + \bar{A}C\bar{B} + \bar{A}C\bar{D} + \bar{A}B\bar{C}\bar{D} + A\bar{B}C$$

Karena $\bar{A}.A = 0$ Aksioma B_4b suku pertama dihilangkan

$$Z = \bar{A}\bar{B}C + \bar{A}C\bar{D} + \bar{A}B\bar{C}\bar{D} + A\bar{B}C$$

keluarkan $\bar{B}C$ dari suku pertama dan terakhir, juga faktor $\bar{A}\bar{D}$ dari suku kedua dan ketiga.

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

$$= \bar{B}C + \bar{A}\bar{D}(C + B) \quad \text{Teorema 2.8 dan Aksioma } B_4a .$$

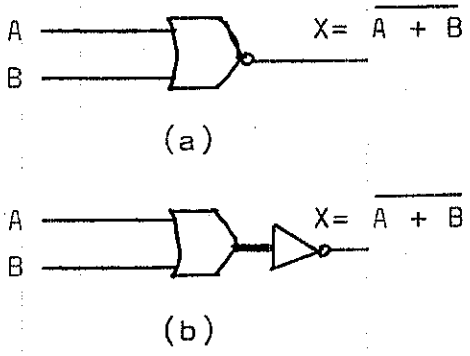
Persamaan ini tidak dapat disederhanakan lagi.

3.5 PINTU NOR DAN PINTU NAND

Dua jenis logika ini, pintu NOR dan pintu NAND digunakan-secara luas dalam rangkaian digital. Kedua pintu ini sesungguhnya merupakan kombinasi operasi dasar AND, OR dan NOT sehingga menjadi relatif lebih mudah untuk membahas kedua fase ini dengan menggunakan operasi aljabar Boole yang dibahas terdahulu.

PINTU NOR

Gambar 3.9a menunjukkan simbol pintu NOR dua masukan. Operasi pintu NOR ekivalen dengan pintu OR yang diikuti dengan inverter seperti ditunjukkan pada Gambar 3.9b



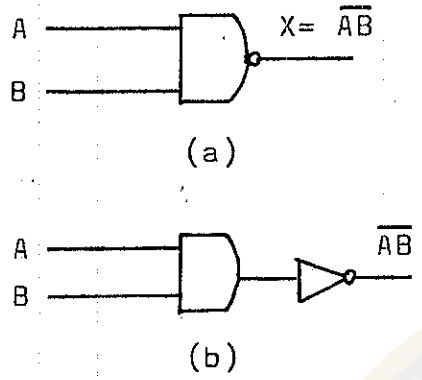
A	B	OR: A + B	NOR: $\overline{A + B}$
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

(c)

Gambar 3.9 (a) Simbol NOR (b) Rangkaian ekivalen (c) Tabel kebenaran.

PINTU NAND

Gambar 3.10a menunjukkan simbol pintu NAND dua masukan. Operasi pintu NAND ekivalen dengan pintu AND yang diikuti oleh se buah inverter, seperti ditunjukkan pada Gambar 3.10b



A	B	AND $A.B$	NAND $\overline{A.B}$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

(c)

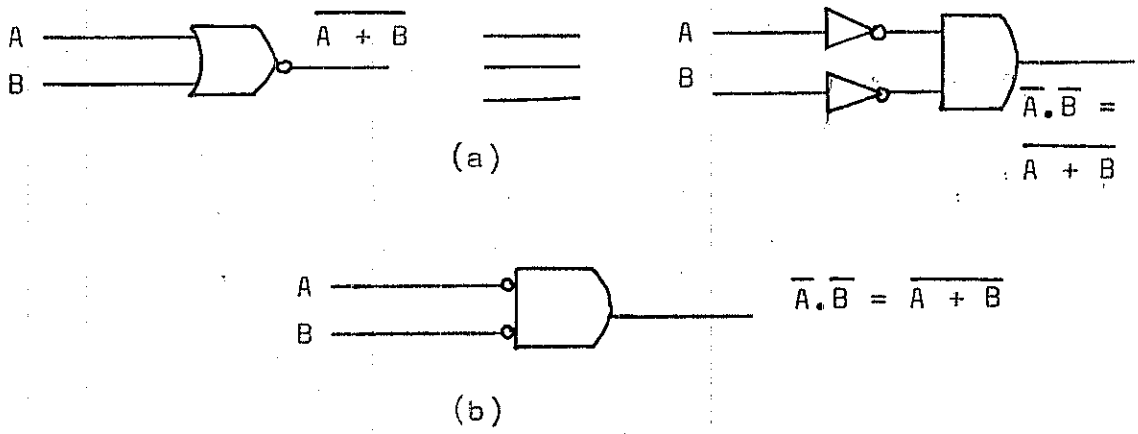
Gambar 3.10 (a) Simbol NAND (b) Rangkaian ekivalen (c) Tabel kebenaran.

IMPLIKASI TEOREMA DE MORGAN'S

Berikut ini akan dikaji lebih lanjut Teorema 2.7 dari su dut rangkaian logika. Pertama-tama perhatikanlah Teorema 2.7a

$$\overline{A + B} = \overline{A}.\overline{B}$$

Ruas kiri persamaan dapat dipandang sebagai keluaran pintu NOR - yang masukannya A dan B yang diikuti dengan pintu AND. Dua representasi ini ekivalen seperti ditunjukkan pada Gambar - 3.11a . Ini berarti bahwa pintu AND dengan inverter pada tiap-tiap masukannya ekivalen dengan pintu NOR. Dalam kenyataannya kedua ba ngun tersebut digunakan untuk menyatakan fungsi NOR. Apabila pin- tu AND dengan masukan yang diinversikan digunakan untuk menyata - kan fungsi NOR, biasanya digambar seperti ditunjukkan pada Gambar 3.11b . Pada Gambar ini lingkaran kecil pada masukan menyatakan - inverter.



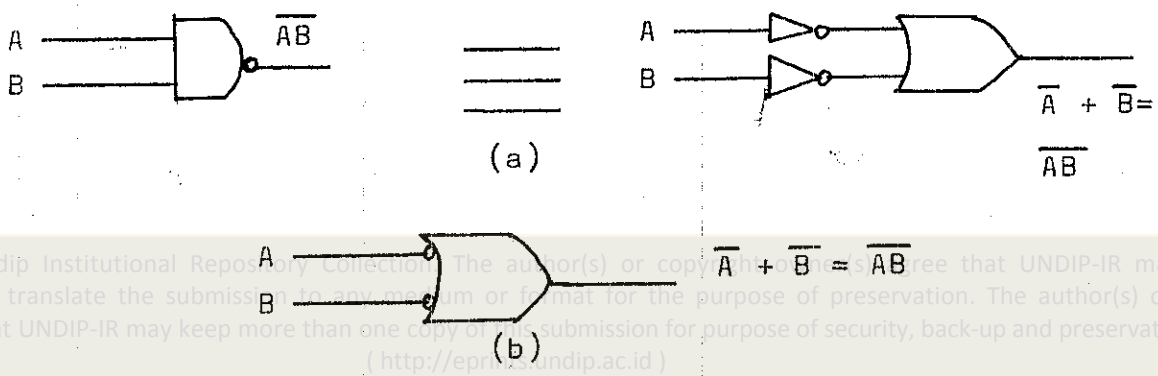
Gambar 3.11 (a) Rangkaian ekivalen yang diturunkan dari Teorema - 2.7a

(b) Simbol lain untuk fungsi NOR

Sekarang perhatikan Teorema 2.7b

$$\overline{A.B} = \overline{A} + \overline{B}$$

Ruas kiri persamaan dapat dilaksanakan oleh pintu NAND dengan masukan A dan B. Ruas kanan dapat dilaksanakan dengan pertama-tama mengiversikan masukan A dan B kemudian memberikannya kepada-pintu OR. Dua bangun ekivalen ini ditunjukkan pada Gambar 3.12a . Pintu OR dengan inverter pada tiap-tiap masukannya ekivalen dengan pintu NAND. Dalam kenyataannya kedua bangun tersebut digunakan untuk menyatakan fungsi NAND. Apabila pintu OR dengan kedua masukkan nya diinversikan digunakan untuk menyatakan fungsi NAND, biasanya-dinyatakan seperti Gambar 3.12b . Pada Gambar ini lingkaran kecil-juga menyatakan inversi.



Gambar 3.12
(a) Rangkaian ekivalen yang diturunkan dari Teorema 2.7b
(b) Simbol lain untuk fungsi NAND

Contoh 3.4 Sederhanakanlah persamaan $Z = \overline{(\overline{A} + C).(B + \overline{D})}$

Jawab :

Dengan menggunakan Teorema 2.7b persamaan ini dapat ditulis kembali sebagai :

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

Persamaan ini dapat dianggap sebagai memotong ditengah tanda inverter besar tersebut dan mengubah tanda AND (.) menjadi OR-(+) . Sekarang suku $(\overline{\overline{A} + C})$ dapat disederhanakan dengan menggunakan Teorema 2.7a, demikian juga $(\overline{B + \overline{D}})$ dapat disederhanakan :

$$Z = (\overline{\overline{A} + C}) + (\overline{B + \overline{D}})$$
$$= (\overline{\overline{A}}.\overline{C}) + (\overline{B}.\overline{\overline{D}})$$

Disini inverter besar telah dipotong ditengah dan (+) diganti dengan (.), dengan menghapuskan inversi ganda akhirnya didapatkan :

$$Z = A\overline{C} + \overline{B}D$$

Contoh 3.4 menunjukkan bahwa pada saat menggunakan Teorema De Morgan's untuk menyederhanakan suatu persamaan, tanda inverter boleh dipotong pada setiap titik pada persamaan tersebut dan operator - pada titik itu diubah menjadi lawannya (+ diubah menjadi . dan sebaliknya). Prosedur ini dilanjutkan sampai persamaan tersebut-disederhanakan menjadi suatu persamaan yang hanya mengandung variabel tunggal yang diiversikan. Dua contoh lagi ditunjukkan berikut ini :

1. $Z = \overline{A + \overline{B}.C}$
$= \overline{A}.(\overline{\overline{B}.C})$
$= \overline{A}.(\overline{\overline{B}} + \overline{C})$
$= \overline{A}.(B + \overline{C})$

2. $Z = \overline{(A + BC).(D + EF)}$
$= (\overline{A + BC}) + (\overline{D + EF})$
$= (\overline{A}.\overline{BC}) + (\overline{D}.\overline{EF})$
$= [\overline{A}.(\overline{B} + \overline{C})] + [\overline{D}.(\overline{E} + \overline{F})]$
$= \overline{AB} + \overline{AC} + \overline{DE} + \overline{DF}$

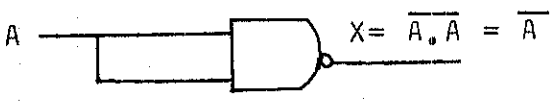
3.6 KESEBANGUNAN PINTU NAND DAN PINTU NOR

Semua persamaan Boole terdiri dari berbagai macam kombinasi operasi dasar OR, AND dan INVERSI. Oleh karena itu setiap persamaan dapat dilaksanakan dengan menggunakan pintu OR, pintu AND dan INVERTER. Tetapi bagaimanapun juga ada kemungkinan untuk melaksanakan setiap persamaan logika hanya dengan menggunakan pintu AND tanpa jenis pintu lainnya. Hal ini dikarenakan pintu NAND dalam kombinasi yang benar dapat digunakan untuk melaksanakan setiap operasi Boole OR, AND dan INVERTER. Pelaksanaannya ditunjukkan pada Gambar 3.13.

Pertama pada Gambar 3.13a terdapat sebuah pintu AND dua masukan yang kedua masukannya dihubungkan menjadi satu sehingga variabel A diberikan pada kedua masukan tersebut. Dalam bangun ini pintu NAND bekerja sebagai sebuah inverter, karena keluarannya adalah $X = \overline{A.A} = \overline{A}$.

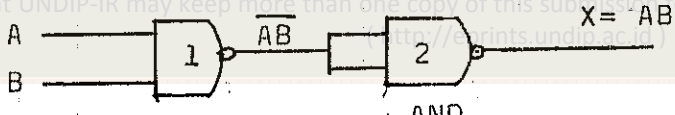
Pada Gambar 3.13b terdapat dua pintu NAND yang dihubungkan sehingga diperoleh operasi AND. Pintu NAND 2 digunakan sebagai inverter untuk mengubah $\overline{AB}$ menjadi $\overline{\overline{AB}} = AB$, yang memenuhi persyaratan fungsi AND.

Operasi OR dapat dilaksanakan dengan menggunakan pintu NAND yang dihubungkan seperti ditunjukkan pada Gambar 3.13c . Disini pintu NAND 1 dan 2 digunakan sebagai inverter untuk mengin - versikan masukannya, sehingga keluaran akhirnya adalah $X = \overline{\overline{A}.\overline{B}}$, yang dapat disederhanakan menjadi X= A + B dengan menggunakan Teorema De Morgan's.



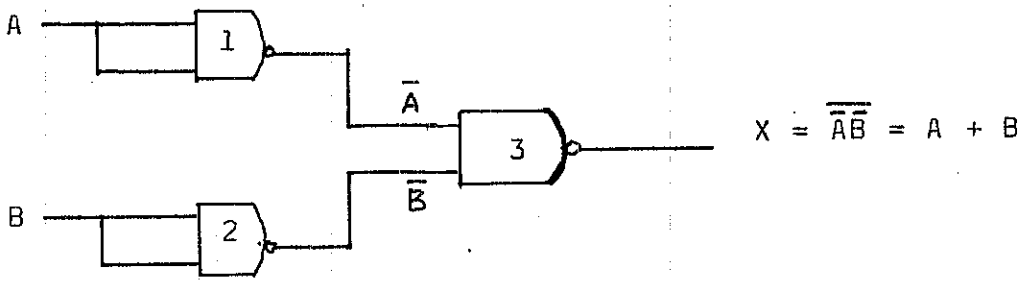
(a)

INVERTER



(b)

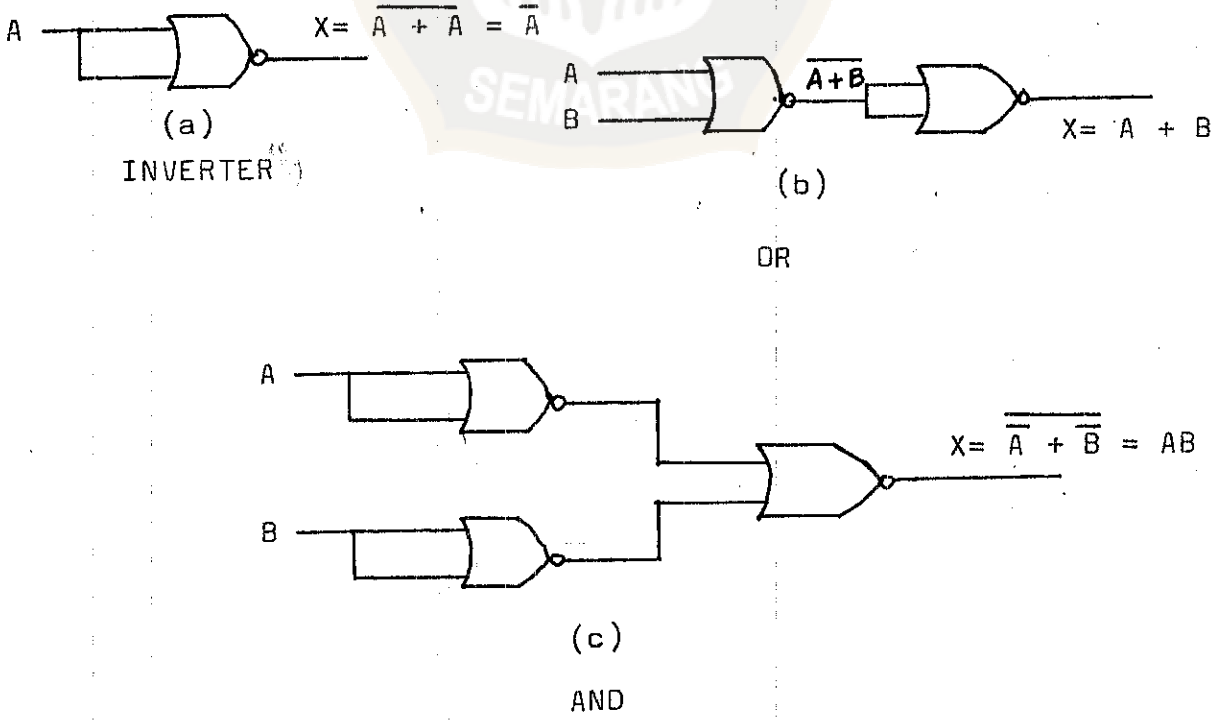
AND



(c) OR

Gambar 3.13 Pintu NAND dapat digunakan untuk melaksanakan setiap operasi Boole.

Dengan cara yang sama dapat ditunjukkan bahwa pintu NOR da pat disusun untuk melaksanakan setiap operasi Boole. Hal ini ditun jukkan pada Gambar 3.14 . Bagian (a) menunjukkan sebuah pintu NOR dengan kedua masukannya dihubungkan menjadi satu mempunyai sifat - sebagai sebuah inverter, karena keluarannya adalah $X = A + A = A$.



Gambar 3.14 Pintu NOR dapat digunakan untuk melaksanakan setiap- operasi Boole.

Pada Gambar 3.14b dua pintu NOR disusun sedemikian rupa - sehingga terlaksana operasi OR. Pintu NOR 2 digunakan sebagai in-

verter untuk mengubah $\overline{A + B}$ menjadi $\overline{\overline{A + B}} = A + B$, yang memenuhi persyaratan fungsi OR.

Operasi AND dapat dilaksanakan dengan menggunakan pintu - NOR seperti ditunjukkan pada Gambar 3.14c. Disini pintu NOR 1 dan 2 digunakan sebagai inverter untuk menginversikan masukannya, sehingga keluaran akhir adalah $X= \overline{\overline{A} + \overline{B}}$, yang dapat disederhanakan menjadi $X = A.B$ dengan menggunakan Teorema De Morgan's.

Kenyataan bahwa pintu NAND dapat digunakan untuk melaksanakan semua operasi Boole berarti bahwa setiap rangkaian logika tidak memandang bagaimanapun kompleksnya, dapat dilaksanakan hanya dengan menggunakan pintu NAND. Hal yang sama berlaku untuk pintu NOR. Karakteristik pintu NAND dan NOR ini sangat penting. Misalnya hal ini berarti bahwa pabrik yang merancang dan memproduksi suatu pesawat digital hanya membutuhkan persediaan satu jenis pintu. Dalam BAB IV akan dijelaskan bagaimana setiap rancangan rangkaian logika dapat dilaksanakan hanya dengan menggunakan pintu - NAND atau NOR.

3.7 RANGKAIAN EKSKLUSIF-OR DAN EKSKLUSIF-NOR

Dua rangkaian logika khusus yang sering dijumpai dalam sistem digital adalah rangkaian eksklusif-OR dan eksklusif-NOR.

EKSKLUSIF - OR

Perhatikan rangkaian logika Gambar 3.15a . Bentuk persamaan keluaran dari rangkaian ini adalah :

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

Tabel kebenaran yang menyertainya menunjukkan bahwa X=1 untuk dua kasus A=0, B=1 (suku $\overline{A}B$) dan A=1, B=0 (suku $A\overline{B}$). Dengan kata lain , rangkaian ini menghasilkan keluaran tinggi apabila kedua masukannya berada pada tingkat yang berlawanan. Rangkaian ini disebut eksklusif OR , yang seterusnya disingkat EX-OR.

Kombinasi khusus pintu logika seperti ini sangat sering terjadi dan sangat berguna dalam pemakaian tertentu. Rangkaian -

EX-OR diberi simbol seperti ditunjukkan pada Gambar 3.15b . Simbol ini dianggap mengandung semua logika yang terdapat didalam rangkaian EX-OR nya dan oleh karena itu mempunyai persamaan logika dan tabel kebenaran yang sama. Rangkaian EX-OR ini umumnya dikenal sebagai pintu EX-OR, dan dapat dipandang sebagai sebuah jenis pintu logika yang lain, disamping pintu yang telah dibahas terdahulu.

Pintu EX-OR hanya mempunyai dua masukan ; tidak ada pintu EX-OR tiga masukan atau empat masukan. Dua masukan tersebut digabung sedemikian rupa sehingga $X = \bar{A}B + A\bar{B}$. Cara singkat yang kadang-kadang digunakan untuk menunjukkan bentuk persamaan keluaran EX-OR adalah :

$$X = A \oplus B$$

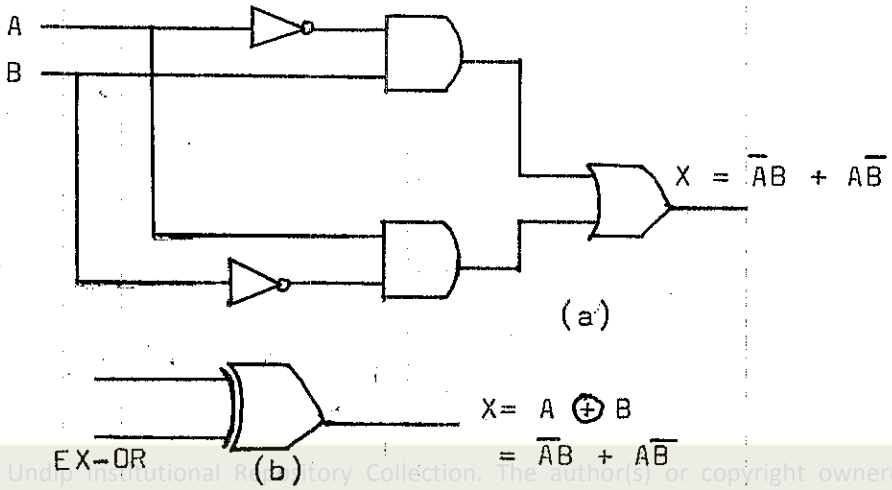
Disini simbol $\oplus$ menyatakan operasi pintu EX-OR

Karakteristik pintu EX-OR diikhtisarkan seperti berikut ini :

1. Hanya mempunyai dua masukan dan keluarannya adalah :

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

2. Keluarannya tinggi hanya apabila dua masukannya berada pada tingkat yang berlawanan.



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

Gambar 3.15 (a) Rangkaian Eksklusif-OR dan Tabel kebenaran
(b) Simbol pintu Eksklusif-OR

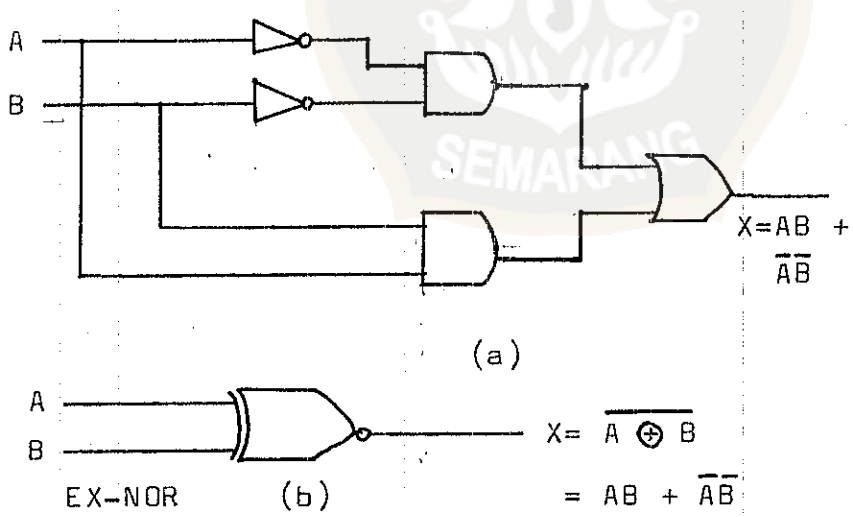
EKSKLUSIF-NOR

Rangkaian eksklusif NOR (disingkat EX-NOR) bekerjanya sepenuhnya berlawanan dengan rangkaian EX-OR. Gambar 3.16a menunjukkan rangkaian EX-NOR dan tabel kebenaran yang menyertainya. Bentuk persamaan keluarannya adalah :

$$X = AB + \overline{A}\,\overline{B}$$

yang bersama-sama dengan tabel kebenarannya menunjukkan bahwa X akan 1 untuk dua kasus : A=B=1 (suku AB) dan A=B=0 (suku $\overline{A}\,\overline{B}$). Dengan kata lain, rangkaian ini menghasilkan keluaran tinggi apabila kedua masukannya berada pada tingkat yang sama.

Jelas bahwa keluaran rangkaian EX-NOR persis kebalikan dari keluaran rangkaian EX-OR. Simbol pintu EX-NOR diperoleh hanya dengan menambahkan lingkaran kecil pada keluaran simbol EX-NOR (Gambar 3.16b)



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

Gambar 3.16 (a) Rangkaian Eksklusif-NOR
(b) Simbol pintu Eksklusif-NOR

Pintu EX-NOR juga hanya mempunyai dua masukan, dan pintu ini menggabungkan kedua masukannya sedemikian rupa sehingga keluarannya adalah :

$$X = AB + \overline{A}\,\overline{B}$$

Cara singkat untuk menunjukkan persamaan keluaran EX-NOR adalah :

$$X = \overline{A \oplus B}$$

Yang merupakan kebalikan operasi EX-OR. Pintu EX-NOR diikhtisar - kan seperti berikut ini :

1. Hanya mempunyai dua masukan dan keluarannya adalah :

$$X = AB + \overline{A}\,\overline{B} = \overline{A \oplus B}$$

2. Keluarannya tinggi hanya apabila dua masukannya berada pada - tingkat yang sama.

Secara aljabar dapat dibuktikan bahwa keluaran EX-NOR per sis kebalikan keluaran EX-OR :

$$\overline{AB + \overline{A}\,\overline{B}} = \overline{AB}.\overline{\overline{A}\,\overline{B}} \qquad \text{(menggunakan Teorema De Morgan's)}$$

$$= (\overline{A} + \overline{B}).(A + B)$$

$$= \overline{A}A + \overline{A}B + A\overline{B} + B\overline{B} \qquad \text{(mengalikan)}$$

Suku pertama dan terakhir sama dengan 0 (ingat bahwa $\overline{A}.A=0$) Ak sioma B_4b, sehingga hasil akhirnya adalah AB + AB yang merupakan pernyataan EX-OR.

Contoh 3.5 Sederhanakanlah $X = \overline{A}BC + BC\overline{D} + A\overline{B}D + A\overline{C}D$

Jawab :

Faktor yang sama dikeluarkan,

$$X = BC(\overline{A} + \overline{D}) + AD(\overline{B} + \overline{C}) \qquad \text{Teorema De Morgan's}$$

$$= BC(\overline{AD}) + AD(\overline{BC})$$

Persamaan ini hendaknya dipandang sebagai penggabungan - EX-OR atas suku BC dan AD . Yaitu,

$$X = BC \oplus AD$$

Persamaan ini dapat dilaksanakan dengan menggunakan dua pintu AND dua masukan untuk suku BC dan AD dan satu pintu EX-OR. Pernyataan aslinya untuk X memerlukan empat pintu AND tiga masukan, sebuah - pintu OR empat masukan dan empat inverter.