Aljabar Boolean

IF2120 Matematika Diskrit

Oleh: Rinaldi Munir
Program Studi Informatika, STEI-ITB

Pengantar

- Aljabar Boolean ditemukan oleh George Boole, pada tahun 1854.

- Boole melihat bahwa himpunan dan logika proposisi mempunyai sifat-sifat yang serupa (perhatikan kemiripan hukum-hukum aljabar logika dan hukum-hukum aljabar himpunan).

- Dalam buku *The Laws of Thought*, Boole memaparkan aturan-aturan dasar logika.

- Aturan dasar logika ini membentuk struktur matematika yang disebut **aljabar Boolean.**

- Aplikasi: perancangan rangkaian pensaklaran, rangkaian digital, dan rangkaian *IC* (*integrated circuit*) komputer

Definisi Aljabar Boolean

DEFINISI. Misalkan B adalah himpunan yang didefinisikan pada dua operator biner, + dan ·, dan sebuah operator uner, ’. Misalkan 0 dan 1 adalah dua elemen yang berbeda dari B. Maka, tupel

$$\langle B, +, \cdot, ', 0, 1 \rangle$$

disebut **aljabar Boolean** jika untuk setiap $a, b, c \in B$ berlaku aksioma berikut:

1. Identitas
   (i) $a + 0 = a$
   (ii) $a \cdot 1 = a$

2. Komutatif
   (i) $a + b = b + a$
   (ii) $a \cdot b = b . a$

3. Distributif
   (i) $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$
   (ii) $a + (b \cdot c) = (a + b) \cdot (a + c)$

4. Komplemen
   Untuk setiap $a \in B$ terdapat elemen unik $a' \in B$ sehingga
   (i) $a + a' = 1$
   (ii) $a \cdot a' = 0$

- Berhubung elemen-elemen *B* tidak didefinisikan nilainya (kita bebas menentukan anggota-anggota *B*), maka terdapat banyak sekali aljabar boolean.

- Untuk mempunyai sebuah aljabar Boolean, orang harus memperlihatkan:
  1. elemen-elemen himpunan *B*,
  2. kaidah/aturan operasi untuk dua operator biner dan operator uner,
  3. himpunan *B*, bersama-sama dengan dua operator tersebut, memenuhi keempat aksioma di atas

- Aljabar himpunan dan aljabar logika proposisi juga merupakan aljabar Boolean karena memenuhi empat aksioma di atas.

- Dengan kata lain, aljabar himpunan dan aljabar proposisi adalah himpunan bagian (*subset*) dari aljabar Boolean.

- Pada aljabar proposisi misalnya:

  - *B* berisi semua proposisi dengan *n* peubah.
  - dua elemen unik berbeda dari *B* adalah **T** dan **F**,
  - operator biner: $\vee$ dan $\wedge$, operator uner: ~
  - semua aksioma pada definisi di atas dipenuhi

Dengan kata lain <*B*, $\vee$, $\wedge$, ~, **F**, **T**> adalah aljabar Booelan

Aljabar Boolean 2-Nilai

- Merupakan aljabar Boolean yang paling popular, karena aplikasinya luas.
- Pada aljabar 2-nilai:

  (i) $B = \{0, 1\}$,

  (ii) operator biner: + dan ·,   operator uner: ’

  (iii) Kaidah untuk operator biner dan operator uner:

a	b	$a \cdot b$
0	0	0
0	1	0
1	0	0
1	1	1

a	b	$a + b$
0	0	0
0	1	1
1	0	1
1	1	1

a	a'
0	1
1	0

  (iv) Keempat aksioma di atas dipenuhi

Ekspresi Boolean

- Ekspresi Boolean dibentuk dari elemen-elemen *B* dan/atau peubah-peubah yang dapat dikombinasikan satu sama lain dengan operator +, ·, dan ’.

- **Contoh 1:**

  0

  1

  a

  b

  $a + b$

  $a \cdot b$

  $a' \cdot (b + c)$

  $a \cdot b' + a \cdot b \cdot c' + b'$, dan sebagainya

Hukum-hukum Aljabar Boolean

1. Hukum identitas: (i) $a + 0 = a$ (ii) $a \cdot 1 = a$	2. Hukum idempoten: (i) $a + a = a$ (ii) $a \cdot a = a$
3. Hukum komplemen: (i) $a + a' = 1$ (ii) $aa' = 0$	4. Hukum dominansi: (i) $a \cdot 0 = 0$ (ii) $a + 1 = 1$
5. Hukum involusi: (i) $(a')' = a$	6. Hukum penyerapan: (i) $a + ab = a$ (ii) $a(a + b) = a$
7. Hukum komutatif: (i) $a + b = b + a$ (ii) $ab = ba$	8. Hukum asosiatif: (i) $a + (b + c) = (a + b) + c$ (ii) $a (b c) = (a b) c$
9. Hukum distributif: (i) $a + (b c) = (a + b) (a + c)$ (ii) $a (b + c) = a b + a c$	10. Hukum De Morgan: (i) $(a + b)' = a'b'$ (ii) $(ab)' = a' + b'$
11. Hukum 0/1 (i) $0' = 1$ (ii) $1' = 0$	

Contoh 2: Buktikan bahwa untuk sembarang elemen *a* dan *b* dari aljabar Boolean maka kesamaaan berikut:

$$a + a'b = a + b \quad \text{dan} \quad a(a' + b) = ab$$

adalah benar.

Penyelesaian:

(i) $a + a'b = (a + ab) + a'b$ (Hukum Penyerapan)

$= a + (ab + a'b)$ (Hukum Asosiatif)

$= a + (a + a')b$ (Hukum Distributif)

$= a + 1 \cdot b$ (Hukum Komplemen)

$= a + b$ (Hukum Identitas)

(ii) $a(a' + b) = a\,a' + ab$ (Hukum Distributif)

$= 0 + ab$ (Hukum Komplemen)

$= ab$ (Hukum Identitas)

Fungsi Boolean

- Contoh-contoh fungsi Boolean:

  $f(x) = x$

  $f(x, y) = x'y + xy' + y'$

  $f(x, y) = x' y'$

  $f(x, y) = (x + y)'$

  $f(x, y, z) = xyz'$

- Setiap peubah di dalam fungsi Boolean, termasuk dalam bentuk komplemennya, disebut **literal**.

- Fungsi $h(x, y, z) = xyz'$ terdiri dari 3 buah literal, yaitu x, y, dan z'.

- Jika diberikan $x = 1$, $y = 1$, $z = 0$, maka nilai fungsinya:

  $h(1, 1, 0) = 1 \cdot 1 \cdot 0' = (1 \cdot 1) \cdot 1 = 1 \cdot 1 = 1$

Bentuk Kanonik

- Ekspresi Boolean yang menspesifikasikan suatu fungsi dapat disajikan dalam dua bentuk berbeda.

- Pertama, sebagai **penjumlahan dari hasil kali** dan kedua sebagai **perkalian dari hasil jumlah**.

- **Contoh 3:**

$$f(x, y, z) = x'y'z + xy'z' + xyz$$

dan

$$g(x, y, z) = (x + y + z)(x + y' + z)(x + y' + z')(x' + y + z')(x' + y' + z)$$

adalah dua buah fungsi yang sama.

- *Minterm*: suku (*term*) di dalam ekspresi boolean mengandung literal yang lengkap dalam bentuk hasil kali

- *Maxterm*: suku (*term*) di dalam ekspresi boolean mengandung literal yang lengkap dalam bentuk hasil jumlah.

- **Contoh 4:**

  $f(x, y, z) = x'y'z + xy'z' + xyz$ → 3 buah *minterm*: $x'y'z$, $xy'z'$, xyz

  $g(x, y, z) = (x + y + z)(x + y' + z)(x + y' + z')(x' + y + z')(x' + y' + z)$
  → 5 buah *maxterm*: $(x + y + z)$, $(x + y' + z)$, $(x + y' + z')$, $(x' + y + z')$, dan $(x' + y' + z)$

- Misalkan peubah (*variable*) fungsi Boolean adalah x, y, dan z
  Maka:

  $x'y$ → bukan *minterm* karena literal tidak lengkap

  $y'z'$ → bukan minterm karena literal tidak lengkap

  $xy'z$, xyz', $x'y'z$ → *minterm* karena literal lengkap

  $(x + z)$ → bukan maxterm karena literal tidak lengkap

  $(x' + y + z')$ → *maxterm* karena literal lengkap

  $(xy' + y' + z)$ → bukan *maxterm*

- Ekspresi Boolean yang dinyatakan sebagai penjumlahan dari satu atau lebih *minterm* atau perkalian dari satu atau lebih *maxterm* disebut dalam **bentuk kanonik**.

- Jadi, ada dua macam bentuk kanonik:
  1. Penjumlahan dari hasil kali (*sum-of-product* atau SOP)
  2. Perkalian dari hasil jumlah (*product-of-sum* atau POS)

- Fungsi $f(x, y, z) = x'y'z + xy'z' + xyz$ dikatakan dalam bentuk SOP

- Fungsi $g(x, y, z) = (x + y + z)(x + y' + z)(x + y' + z')(x' + y + z')$

  $(x' + y' + z)$

  dikatakan dalam bentuk POS

Cara membentuk *minterm* dan *maxterm*:

- Untuk *minterm*, setiap peubah yang bernilai 0 dinyatakan dalam bentuk komplemen, sedangkan peubah yang bernilai 1 dinyatakan tanpa komplemen.

- Sebaliknya, untuk *maxterm*, setiap peubah yang bernilai 0 dinyatakan tanpa komplemen, sedangkan peubah yang bernilai 1 dinyatakan dalam bentuk komplemen.

- Cara membentuk *minterm* dan *maxterm* dari tabel kebenaran untuk dua peubah:

		Minterm		*Maxterm*	
x	y	Suku	Lambang	Suku	Lambang
0	0	$x'y'$	m_0	$x + y$	M_0
0	1	$x'y$	m_1	$x + y'$	M_1
1	0	xy'	m_2	$x' + y$	M_2
1	1	$x\,y$	m_3	$x' + y'$	M_3

- Cara membentuk *minterm* dan *maxterm* dari tabel kebenaran untuk tiga peubah:

			Minterm		*Maxterm*	
x	y	z	Suku	Lambang	Suku	Lambang
0	0	0	$x'y'z'$	m_0	$x + y + z$	M_0
0	0	1	$x'y'z$	m_1	$x + y + z'$	M_1
0	1	0	$x'y z'$	m_2	$x + y'+z$	M_2
0	1	1	$x'y z$	m_3	$x + y'+z'$	M_3
1	0	0	$x y'z'$	m_4	$x'+ y + z$	M_4
1	0	1	$x y'z$	m_5	$x'+ y + z'$	M_5
1	1	0	$x y z'$	m_6	$x'+ y'+ z$	M_6
1	1	1	$x y z$	m_7	$x'+ y'+ z'$	M_7

- Jika diberikan sebuah tabel kebenaran, kita dapat membentuk fungsi Boolean dalam bentuk kanonik (SOP atau POS) dari tabel tersebut dengan cara:

  - mengambil *minterm* dari setiap nilai fungsi yang bernilai 1 (untuk SOP)

    atau

  - mengambil *maxterm* dari setiap nilai fungsi yang bernilai 0 (untuk POS).

Contoh 5: Tinjau fungsi Boolean yang dinyatakan oleh Tabel di bawah ini. Nyatakan fungsi tersebut dalam bentuk kanonik SOP dan POS

x	y	z	$f(x, y, z)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Penyelesaian:

- **SOP**

  Kombinasi nilai-nilai peubah yang menghasilkan nilai fungsi sama dengan 1 adalah 001, 100, dan 111, maka fungsi Booleannya dalam bentuk kanonik SOP adalah

  $$f(x, y, z) = x'y'z + xy'z' + xyz$$

  atau (dengan menggunakan lambang *minterm*),

  $$f(x, y, z) = m_1 + m_4 + m_7 = \sum (1, 4, 7)$$

- **POS**

x	y	z	$f(x, y, z)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Kombinasi nilai-nilai peubah yang menghasilkan nilai fungsi sama dengan 0 adalah 000, 010, 011, 101, dan 110, maka fungsi Booleannya dalam bentuk kanonik POS adalah

$$f(x, y, z) = (x + y + z)(x + y' + z)(x + y' + z')(x' + y + z')(x' + y' + z)$$

atau dalam bentuk lain,

$$f(x, y, z) = M_0 M_2 M_3 M_5 M_6 = \prod(0, 2, 3, 5, 6)$$

Contoh 6: Nyatakan fungsi Boolean $f(x, y, z) = x + y'z$ dalam bentuk kanonik SOP dan POS.

<u>Penyelesaian</u>:

(a) SOP

Lengkapi terlebih dahulu literal untuk setiap suku agar jumlahnya sama.

$$
\begin{aligned}
x &= x(y + y') \\
&= xy + xy' \\
&= xy\,(z + z') + xy'(z + z') \\
&= xyz + xyz' + xy'z + xy'z'
\end{aligned}
$$

dan

$$y'z = y'z\,(x + x') = \mathrm{xy'z} + \mathrm{x'y'z}$$

$$
\begin{aligned}
\text{Jadi } f(x, y, z) &= x + y'z \\
&= xyz + xyz' + xy'z + xy'z' + xy'z + x'y'z \\
&= x'y'z + xy'z' + xy'z + xyz' + xyz
\end{aligned}
$$

atau $f(x, y, z) = m_1 + m_4 + m_5 + m_6 + m_7 = \Sigma\ (1,4,5,6,7)$

(b) POS

$f(x, y, z) = x + y'z$

$= (x + y')(x + z)$

Lengkapi terlebih dahulu literal pada setiap suku agar jumlahnya sama:

$x + y' = x + y' + zz'$

$= (x + y' + z)(x + y' + z')$

$x + z = x + z + yy'$

$= (x + y + z)(x + y' + z)$

Jadi, $f(x, y, z) = (x + y' + z)(x + y' + z')(x + y + z)(x + y' + z)$

$= (x + y + z)(x + y' + z)(x + y' + z')$

atau $f(x, y, z) = M_0M_2M_3 = \prod(0, 2, 3)$

Contoh 7: Nyatakan fungsi Boolean $f(x, y, z) = xy + x'z$ dalam bentuk kanonik POS.

Penyelesaian:

$$
\begin{aligned}
f(x, y, z) &= xy + x'z \\
&= (xy + x')(xy + z) \\
&= (x + x')(y + x')(x + z)(y + z) \\
&= (x' + y)(x + z)(y + z)
\end{aligned}
$$

Lengkapi literal untuk setiap suku agar jumlahnya sama:

$$
\begin{aligned}
x' + y &= x' + y + zz' &&= (x' + y + z)(x' + y + z') \\
x + z &= x + z + yy' &&= (x + y + z)(x + y' + z) \\
y + z &= y + z + xx' &&= (x + y + z)(x' + y + z)
\end{aligned}
$$

Jadi, $f(x, y, z) = (x + y + z)(x + y' + z)(x' + y + z)(x' + y + z')$

atau $f(x, y, z) = M_0 M_2 M_4 M_5 = \prod(0,2,4,5)$

Konversi Antar Bentuk Kanonik

Misalkan *f* adalah fungsi Boolean dalam bentuk SOP dengan tiga peubah:

$$f(x, y, \text{z}) = \Sigma\ (1, 4, 5, 6, 7)$$

dan *f '* adalah fungsi komplemen dari *f*,

$$f\,'(x, y, z) = \Sigma\ (0, 2, 3) = m_0 + m_2 + m_3$$

Dengan menggunakan hukum De Morgan, kita dapat memperoleh fungsi *f* dalam bentuk POS:

$$\begin{aligned} f\,(x, y, z) &= (f\,'(x, y, z))' = (m_0 + m_2 + m_3)' = m_0' \,.\, m_2' \,.\, m_3' \\ &= (x'y'z')'\ (x'y\,z')'\ (x'y\,z)' \\ &= (x + y + z)\ (x + y' + z)\ (x + y' + \text{z}') \\ &= M_0\, M_2\, M_3 = \prod\ (0,2,3) \end{aligned}$$

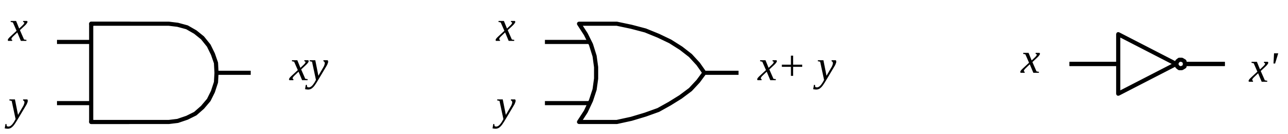
Jadi, $f(x, y, \text{z}) = \Sigma\ (1, 4, 5, 6, 7) = \prod\ (0,2,3)$.

<u>Kesimpulan</u>: $m_j' = M_j$

Rangkaian Logika

- Fungsi Boolean dapat juag direpresentasikan dalam bentuk rangkaian logika.

- Ada tiga gerbang logika dasar: gerbang AND, gerbang OR, dan gerbang NOT

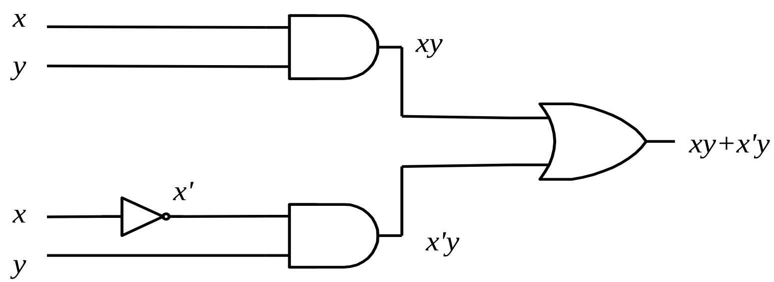


Gerbang AND dua-masukan   Gerbang OR dua-masukan   Gerbang NOT (*inverter*)

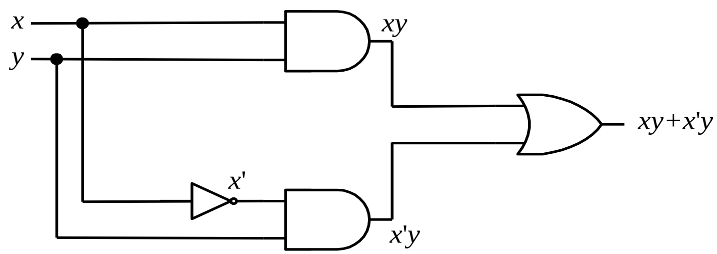
Contoh 8: Nyatakan fungsi $f(x, y, z) = xy + x'y$ ke dalam rangkaian logika.

Penyelesaian: Ada beberapa cara penggambaran

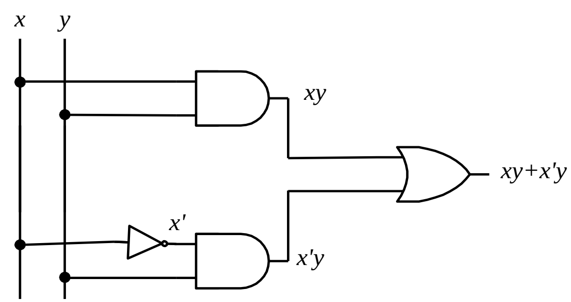
Cara pertama:



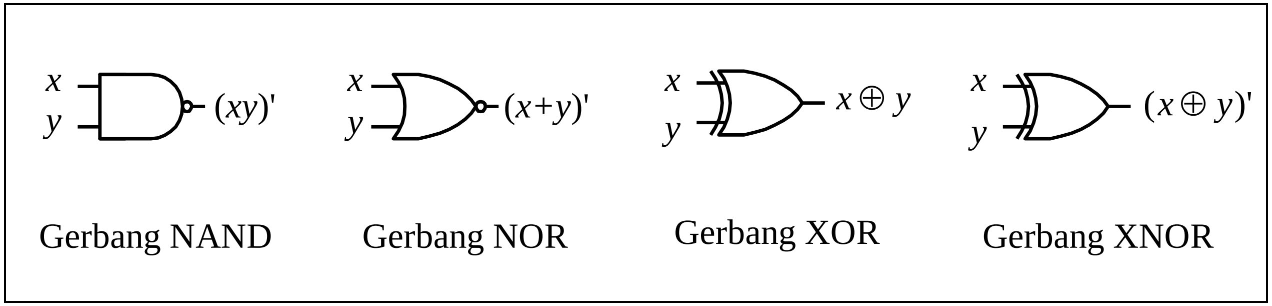
Cara kedua:



Cara ketiga:



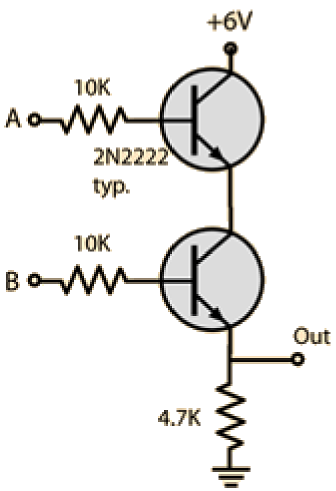
- Gerbang logika turunan: NAND, NOR, XOR, dan XNOR



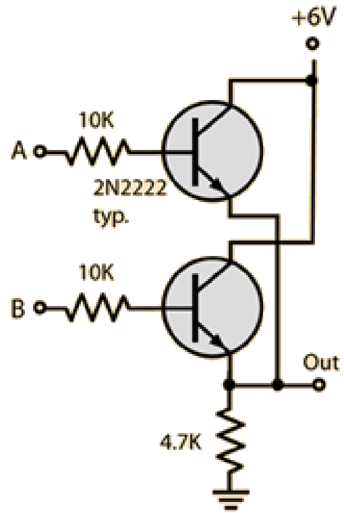
Keempat gerbang di atas merupakan kombinasi dari gerbang-gerbang dasar, misalnya gerbang NOR disusun oleh kombinasi gerbang OR dan gerbang NOT:

Selain itu, dengan menggunakan hukum De Morgan, kita juga dapat membuat gerbang logika yang ekivalen dengan gerbang NOR dan NAND di atas:

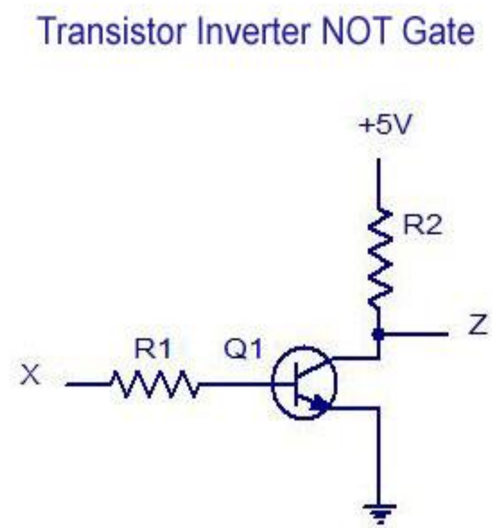
Transistor untuk gerbang logika



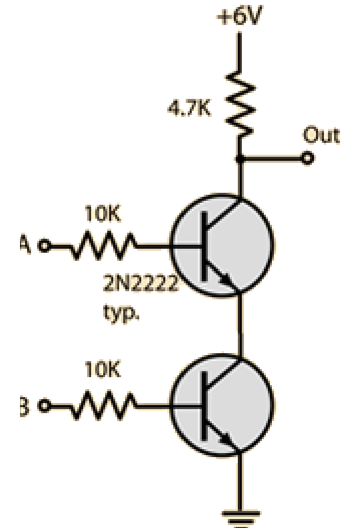
AND



OR



NOT



NAND

Sumber gambar: http://hyperphysics.phy-astr.gsu.edu/hbase/electronic/trangate.html#c3

Penyederhanaan Fungsi Boolean

- Menyederhanakan fungsi Boolean artinya mencari bentuk fungsi lain yang ekivalen tetapi dengan jumlah literal atau operasi yang lebih sedikit.

- Contoh: $f(x, y) = x'y + xy' + y'$ disederhanakan menjadi $f(x, y) = x' + y'$.

- Dipandang dari segi aplikasi aljabar Boolean, fungsi Boolean yang lebih sederhana berarti rangkaian logikanya juga lebih sederhana (menggunakan jumlah gerbang logika lebih sedikit).

- Tiga metode yang dapat digunakan untuk menyederhanakan fungsi Boolean:
  1. Secara aljabar, menggunakan hukum-hukum aljabar Boolean.
  2. Metode Peta Karnaugh.
  3. Metode Quine-McCluskey (metode tabulasi)

- Yang dibahas hanyalah **Metode Peta Karnaugh**

Peta Karnaugh

- Peta Karnaugh (atau *K-map*) merupakan metode grafis untuk menyederhanakan fungsi Boolean.

- Metode ini ditemukan oleh Maurice Karnaugh pada tahun 1953. Peta Karnaugh adalah sebuah diagram/peta yang terbentuk dari kotak-kotak (berbentuk bujursangkar) yang bersisian.

- Tiap kotak merepresentasikan sebuah *minterm*.

- Tiap kotak dikatakan bertetangga jika *minterm-minterm* yang merepresentasikannya berbeda hanya 1 buah literal.

Peta Karnaugh dengan dua peubah

m_0	m_1
m_2	m_3

Penyajian 1

		y 0	1
x	0	$x'y'$	$x'y$
	1	xy'	xy

Penyajian 2

	y'	y
x'	$x'y'$	$x'y$
x	xy'	xy

Penyajian 3

Peta Karnaugh dengan tiga peubah

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6

		yz 00	01	11	10
x	0	$x'y'z'$	$x'y'z$	$x'yz$	$x'yz'$
	1	$xy'z'$	$xy'z$	xyz	xyz'

Peta Karnaugh dengan empat peubah

m_0	m_1	m_3	m_2
m_4	m_5	m_7	m_6
m_{12}	m_{13}	m_{15}	m_{14}
m_8	m_9	m_{11}	m_{10}

wx \ *yz*	00	01	11	10
00	*w'x'y'z'*	*w'x'y'z*	*w'x'yz*	*w'x'yz'*
01	*w'xy'z'*	*w'xy'z*	*w'xyz*	*w'xyz'*
11	*wxy'z'*	*wxy'z*	*wxyz*	*wxyz'*
10	*wx'y'z'*	*wx'y'z*	*wx'yz*	*wx'yz'*

Cara mengisi peta Karnaugh

- Kotak yang menyatakan *minterm* diisi “1”
- Sisanya diisi “0”

- Contoh: $f(x, y, z) = x'yz' + xyz' + xyz$

x \ yz	00	01	11	10
0	0	0	0	1
1	0	0	1	1

Contoh: $f(x, y, z) = xz' + y$

xz': Irisan antara:

x → semua kotak pada baris ke-2

z' → semua kotak pada kolom ke-1 dan kolom ke-4

y:

y → semua kotak pada kolom ke-3 dan kolom ke-4

x \ yz	00	01	11	10
0	0	0	1	1
1	1	0	1	1

$xz' + y$

Pengisian peta Karnaugh dari tabel kebenaran

x	y	z	$f(x, y, z)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Tinjau hanya nilai fungsi yang memberikan 1. Fungsi Boolean yang merepresentasikan tabel kebenaran adalah $f(x, y) = x'y'z + xy'z' + xy'z + xyz$.

	yz 00	01	11	10
x 0	0	1	0	0
1	1	1	1	0

Teknik Minimisasi Fungsi Boolean dengan Peta Karnaugh

- Penggunaan Peta Karnaugh dalam penyederhanaan fungsi Boolean dilakukan dengan cara menggabungkan kotak-kotak yang bernilai 1 dan saling bersisian.

- Kelompok kotak yang bernilai 1 dapat membentuk:

  - pasangan (dua),

  - kuad (empat),

  - oktet (delapan).

Pasangan

$wx \backslash yz$	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	1	1
10	0	0	0	0

Bukti secara aljabar:

$$
\begin{aligned}
f(w, x, y, z) &= wxyz + wxyz' \\
&= wxy(z + z') \\
&= wxy(1) \\
&= wxy
\end{aligned}
$$

Sebelum disederhanakan: $f(w, x, y, z) = wxyz + wxyz'$

Sesudah disederhanakan: $f(w, x, y, z) = wxy$

Kuad (1)

$wx \backslash yz$	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	0	0	0	0

Bukti secara aljabar (kuad = 2 buah pasangan):

$$
\begin{aligned}
f(w, x, y, z) &= wxy' + wxy \\
&= wx(z' + z) \\
&= wx(1) \\
&= wx
\end{aligned}
$$

Sebelum: $f(w, x, y, z) = wxy'z' + wxy'z + wxyz + wxyz'$

Sesudah: $f(w, x, y, z) = wx$

Kuad (2)

$wx \backslash yz$	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	0	0
10	1	1	0	0

Sebelum: $f(w, x, y, z) = wxy'z' + wxy'z + wx'y'z' + wx'y'z$

Sesudah: $f(w, x, y, z) = wy'$

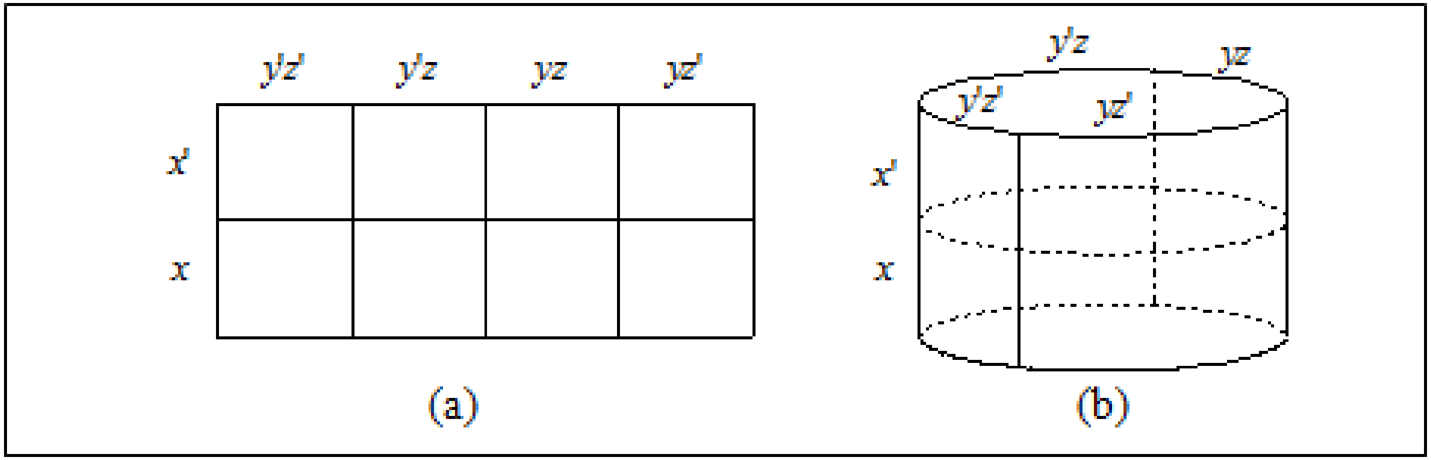
Oktet

wx \ yz	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	1	1	1	1
10	1	1	1	1

Sebelum: $f(w, x, y, z) = wxy'z' + wxy'z + wxyz' + wxy'z +$
$wx'y'z' + wx'y'z + wx'yz + wx'yz'$

Sesudah: $f(w, x, y, z) = w$

Penggulungan (1)



Gambar (a) Peta Karnaugh "normal" dengan 3 peubah

(b) Peta Karnaugh dengan sisi kiri dan sisi kanan ditautkan (seperti digulung).

Penggulungan (2)

Contoh: Sederhanakan $f(x, y, z) = x'yz + xy'z' + xyz + xyz'$.

$x \backslash yz$	00	01	11	10
0	0	0	1	0
1	1	0	1	1

Sebelum: $f(x, y, z) = x'yz + xy'z' + xyz + xyz'$

Sesudah: $f(x, y, z) = yz + xz'$

Ketidakunikan Hasil Penyederhanaan

Hasil penyederhanaan dengan peta Karnaugh tidak selalu unik.
Artinya, mungkin terdapat beberapa bentuk fungsi minimasi yang berbeda meskipun jumlah literal dan jumlah *term*-nya sama

Kemungkinan pengelompokan I:

wx \ yz	00	01	11	10
00	0	0	1	1
01	0	1	0	0
11	1	0	1	1
10	1	1	1	0

f(w,x,y,z) = w'x'y + w'xy'z + wxy + wy'z' + wx'z

Kemungkinan pengelompokan II:

wx \ yz	00	01	11	10
00	0	0	1	1
01	0	1	0	0
11	1	0	1	1
10	1	1	1	0

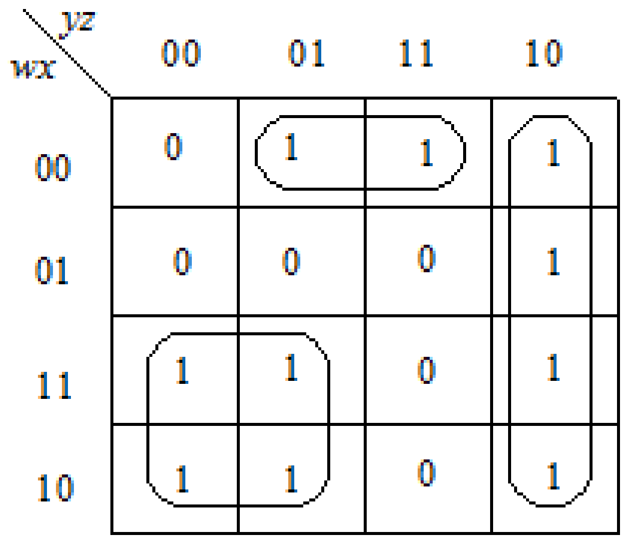
f(w,x,y,z) = w'x'y + w'xy'z + wxz' + wyz + wx'y'

Tips menyederhanakan dengan Peta Karnaugh

- Kelompokkan 1 yang bertetangga sebanyak mungkin

- Dimulai dengan mencari oktet sebanyak-banyaknya terlebih dahulu, kemudian kuad, dan terakhir pasangan.

Contoh minimisasi 1:



Hasil penyederhanaan: $f(w, x, y, z) = wy' + yz' + w'x'z$

Contoh minimisasi 2:

wx \ yz	00	01	11	10
00	0	1	1	0
01	0	1	1	1
11	0	1	1	1
10	1	1	1	0

Hasil penyederhanaan: $f(w, x, y, z) = z + xy + wx'y'$

Contoh minimisasi 3:

wx \ yz	00	01	11	10
00	0	0	0	0
01	0	0	1	0
11	1	1	1	1
10	0	1	1	1

Hasil penyederhanaan: $f(w, x, y, z) = wx + wz + wy + xyz$

Contoh minimisasi 4:

Tentukan bentuk sederhana dari fungsi Boolean yang merepresentasikan tabel kebenaran berikuit dalam bentuk baku SOP dan bentuk baku POS.

x	y	z	$f(x, y, z)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

Penyelesaian:

(a) Bentuk baku SOP: kelompokkan 1

$x \backslash yz$	00	01	11	10
0	0	1	1	0
1	1	0	0	1

Fungsi minimasi: $f(x, y, z) = x'z + xz'$

(b) Bentuk baku POS: kelompokkan 0

$x \backslash yz$	00	01	11	10
0	0	1	1	0
1	1	0	0	1

Fungsi minimasi: $f(x, y, z) = (x' + z')(x + z)$

Contoh minimisasi 5:

Minimisasi fungsi Boolean $f(x, y, z) = \Sigma\,(0, 2, 4, 5, 6)$

<u>Penyelesaian</u>:

Peta Karnaugh untuk fungsi tersebut adalah:

$x \backslash yz$	00	01	11	10
0	1	0	0	1
1	1	1	0	1

Hasil penyederhanaan: $f(x, y, z) = z' + xy'$

Contoh minimisasi 6

Minimisasi $f(w, x, y, z) = w'x'y' + x'yz' + w'xyz' + wx'y'$

<u>Penyelesaian</u>:

$wx \backslash yz$	00	01	11	10
00	1	1	0	1
01	0	0	0	1
11	0	0	0	0
10	1	1	0	1

Hasil penyederhanaan: $f(w, x, y, z) = x'y' + x'z' + w'yz'$

Contoh minimisasi 7

Minimisasi fungsi Boolean $f(w, x, y, z) = \Sigma\ (0,1,2,4,5,6,8,9,12,13,14)$

Penyelesaian:

wx \ yz	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	0	1
10	1	1	0	0

Hasil penyederhanaan: $f(w, x, y, z) = y' + w'z' + xz'$

Contoh minimisasi 8

Sederhanakan fungsi $f(w,x,y,z) = (w + x')(w + x + y)(w' + x' + y')(w' + x + y + z')$.
Hasil penyederhanaan dalam bentuk baku SOP dan POS.

Penyelesaian:

wx	00	01	11	10
00	0	0	1	1
01	0	0	0	0
11	1	1	0	0
10	1	0	1	1

Hasil penyederhanaan

SOP: $f(w, x, y, z) = x'y + wxy' + wy'z'$ (garis penuh)

POS: $f(w, x, y, z) = (x' + y')(w + y)(x + y + z')$ (garis putus-putus)

Contoh minimisasi 9

Sederhanakan fungsi $f(x, y, z, t) = xy' + xyz + x'y'z' + x'yzt'$

<u>Penyelesaian</u>:

Pengelompokan yang berlebihan

$xy \backslash zt$	00	01	11	10
00	1	1	0	0
01	0	0	0	1
11	0	0	1	1
10	1	1	1	1

Pengelompokan yang benar

$xy \backslash zt$	00	01	11	10
00	1	1	0	0
01	0	0	0	1
11	0	0	1	1
10	1	1	1	1

Fungsi minimasi: $f(x, y, z, t) = y'z' + xz + yzt'$

Contoh minimisasi 10

Minimasi fungsi yang telah dipetakan ke peta Karnaugh di bawah ini dalam bentuk baku SOP dan bentuk baku POS.

$wx \backslash yz$	00	01	11	10
00	0	0	1	0
01	1	1	1	0
11	0	1	1	0
10	0	1	1	0

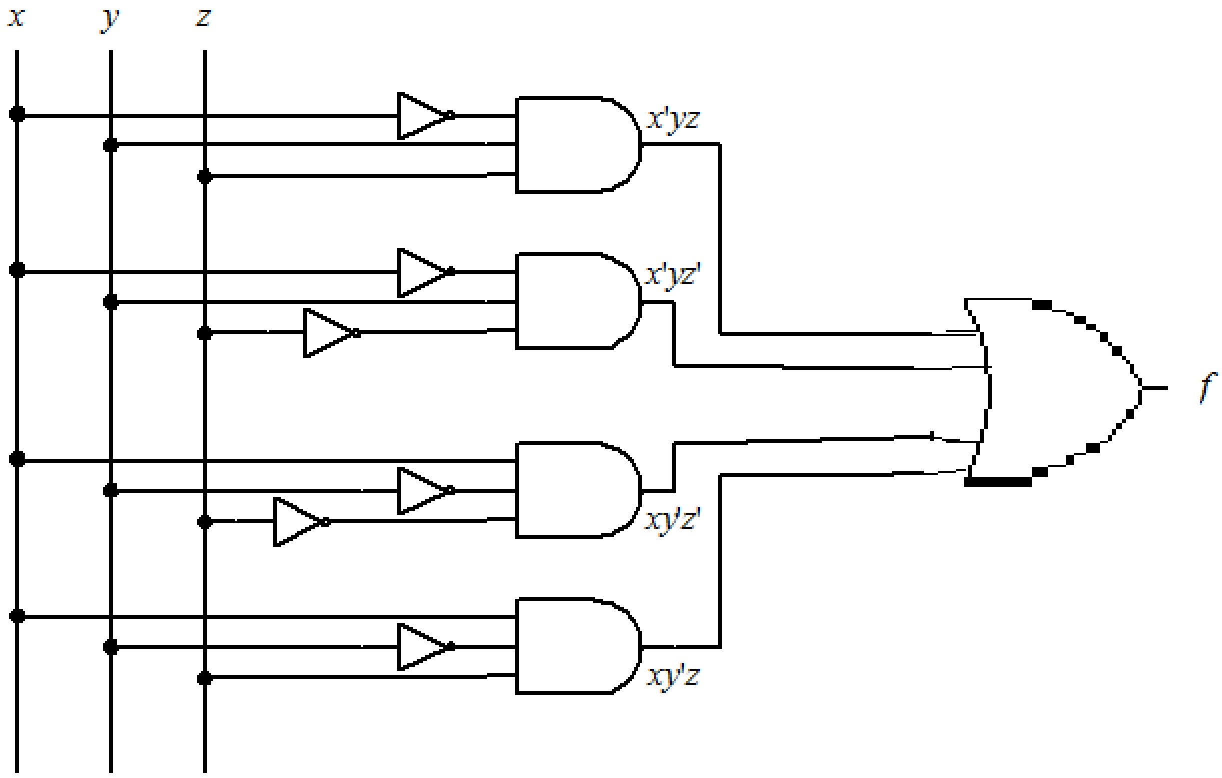
Penyelesaian:

SOP : $f(w, x, y, z) = yz + wz + xz + w'xy'$ (garis penuh)

POS: $f(w, x, y, z) = (y' + z)(w' + z)(x + z)(w + x + y)$ (garis putus-putus

Contoh minimimisasi 11

Sederhanakan rangkaian logika berikuit:



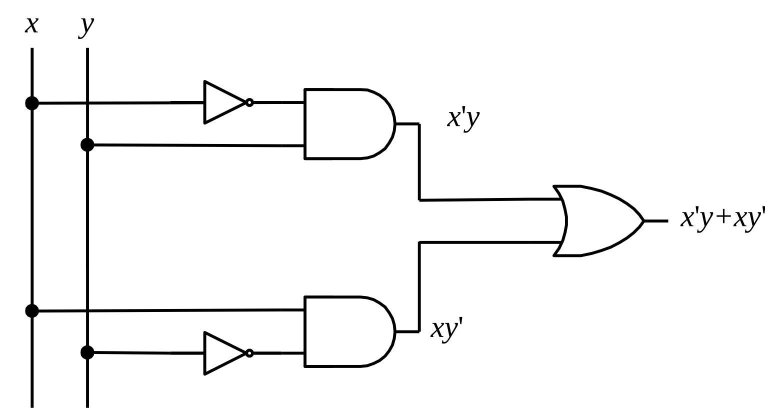
Penyelesaian: Fungsi yang berkoresponden dengan rangkaian logika tsb: $f(x, y, z) = x'yz + x'yz' + xy'z' + xy'z$

x \ yz	00	01	11	10
0	1	0	1	1
1	1	1	0	0

Fungsi Boolean hasil minimisasi:

$$f(x, y, z) = x'y + xy'$$

Rangkaian logika hasil penyederhanaan:



Keadaan *don’t care*

- Keadaan *don’t care* adalah kondisi nilai peubah yang tidak diperhitungkan oleh fungsinya.

- Artinya nilai 1 atau 0 dari peubah *don’t care* tidak berpengaruh pada hasil fungsi tersebut.

- Contoh:

  - peraga digital angka desimal 0 sampai 9.

  - Jumlah bit yang diperlukan untuk merepresentasikan = 4 bit.

  - Bit-bit untuk angka 10-15 tidak terpakai

w	*x*	*y*	*z*	Desimal
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	*X*
1	0	1	1	*X*
1	1	0	0	*X*
1	1	0	1	*X*
1	1	1	0	*X*
1	1	1	1	*X*

don't care

- Dalam menyederhanakan Peta Karnaugh yang mengandung keadaan *don't care*, ada dua hal penting sebagai pegangan.
- Pertama, kita anggap semua nilai *don't care* (*X*) sama dengan 1 dan kemudian membentuk kelompok sebesar mungkin yang melibatkan angka 1 termasuk tanda *X* tersebut.
- Kedua, semua nilai *X* yang tidak termasuk dalam kelompok tersebut kita anggap bernilai 0.
- Dengan cara ini, keadaan-keadaan *X* telah dimanfaatkan semaksimal mungkin, dan kita boleh melakukannya secara bebas.

Contoh: Sebuah fungsi Boolean, *f*, dinyatakan dengan tabel berikut. Minimisasi fungsi *f* sesederhana mungkin.

w	x	y	z	$f(w, x, y, z)$
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	0	0	0	X
1	0	0	1	X
1	0	1	0	X
1	0	1	1	X
1	1	0	0	X
1	1	0	1	X
1	1	1	0	X
1	1	1	1	X

Penyelesaian:

$wx \backslash yz$	00	01	11	10
00	1	0	1	0
01	1	1	1	0
11	X	X	X	X
10	X	0	X	X

Hasil penyederhanaan: $f(w, x, y, z) = xz + y'z' + yz$

Contoh: Minimisasi fungsi Boolean berikut (dalam bentuk baku SOP dan bentuk baku POS): $f(w, x, y, z) = \Sigma\ (1, 3, 7, 11, 15)$ dengan kondisi *don't care* adalah $d(w, x, y, z) = \Sigma\ (0, 2, 5)$.

<u>Penyelesaian</u>:

wx \ yz	00	01	11	10
00	X	1	1	X
01	0	X	1	0
11	0	0	1	0
10	0	0	1	0

Hasil penyederhanaan:

SOP: $f(w, x, y, z) = yz + w'z$ (kelompok garis penuh)

POS: $f(w, x, y, z) = z\ (w' + y)$ (kelompok garis putus-putus)

Perancangan Rangkaian Logika

1. *Majority gate* merupakan sebuah rangkaian digital yang keluarannya sama dengan 1 jika mayoritas masukannya bernilai 1 (mayoritas = 50% + 1). Keluaran sama dengan 0 jika tidak memenuhi hal tersebut di atas. Dengan bantuan tabel kebenaran, carilah fungsi Boolean yang diimplementasikan dengan *3-input majority gate*. Sederhanakan fungsinya, lalu gambarkan rangkaian logikanya.

Penyelesaian:

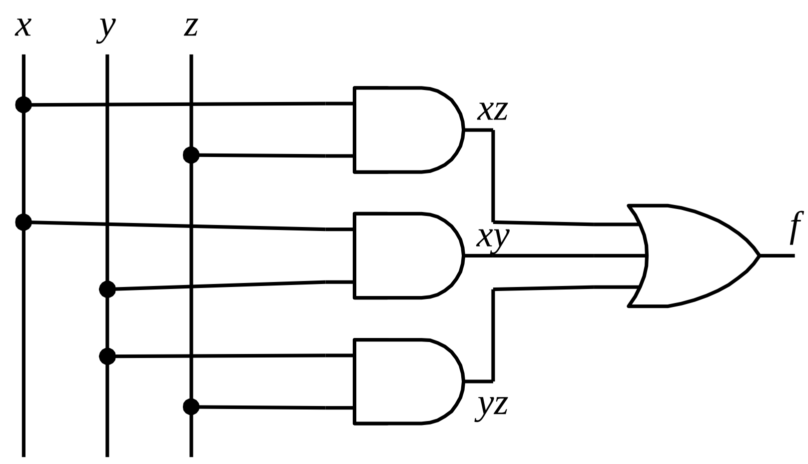
Tabel kebenaran:

x	y	z	$f(x, y, z)$
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

x \ yz	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$f(x, y, z) = xz + xy + yz$

Rangkaian logika:



2. Gunakan Peta Karnaugh untuk merancang rangkaian logika yang dapat menentukan apakah sebuah angka desimal yang direpresentasikan dalam bit biner merupakan bilangan genap atau bukan (yaitu, memberikan nilai 1 jika genap dan 0 jika tidak).

Penyelesaian:

Angka desimal: 0 .. 9 (direpresentasikan dalam 4 bit biner, misalkan $a_0a_1a_2a_3$).

Fungsi $f(a_0, a_1, a_2, a_3)$ bernilai 1 jika representasi desimal dari $a_0a_1a_2a_3$ menyatakan bilangan genap, dan bernilai 0 jika tidak genap.

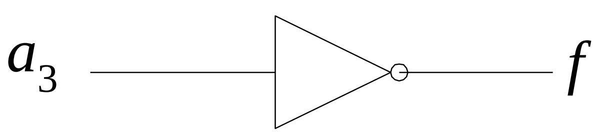
Tabel kebenaran:

a_0	a_1	a_2	a_3	*Desimal*	$f(a_0, a_1, a_2, a_3)$
0	0	0	0	0	1
0	0	0	1	1	0
0	0	1	0	2	1
0	0	1	1	3	0
0	1	0	0	4	1
0	1	0	1	5	0
0	1	1	0	6	1
0	1	1	1	7	0
1	0	0	0	8	1
1	0	0	1	9	0
1	0	1	0	10	**X**
1	0	1	1	11	*X*
1	1	0	0	12	*X*
1	1	0	1	13	*X*
1	1	1	0	14	*X*
1	1	1	1	15	*X*

a_2a_3	00	01	11	10
00	1	0	0	1
01	1	0	0	1
11	*X*	*X*	*X*	*X*
10	1	0	*X*	*X*

$$f(a_0, a_1, a_2, a_3) = a_3'$$

Rangkaian logika:



3. Di dalam unit aritmetika komputer (*Arithmetic Logical Unit – ALU*) terdapat rangkaian penjumlah (*adder*). Salah satu jenis rangkaian penjumlah adalah penjumlah-paruh (*half adder*). Rangkaian ini menjumlahkan 2 bit masukan dengan keluarannya adalah *SUM* (jumlah) dan *CARRY* (pindahan).

x	*y*	*SUM*	*CARRY*
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Peta Karnaugh untuk *SUM*:

x \ *y*	0	1
0	0	1
1	1	0

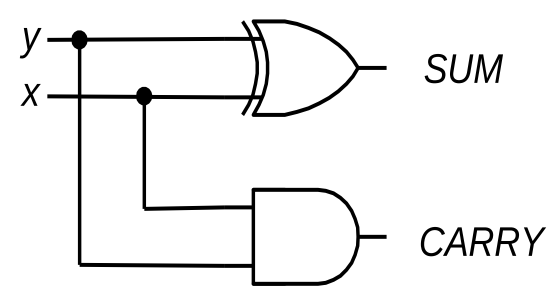
$SUM = x'y + xy' = x \oplus y$

Peta Karnaugh untuk *CARRY*:

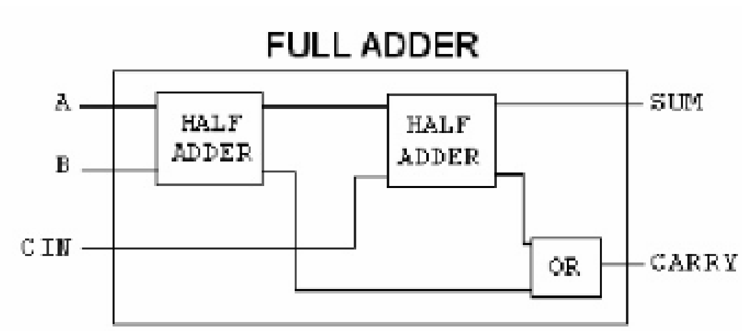
x \ *y*	0	1
0	0	0
1	0	1

$CARRY = xy$

Rangkaian logika:

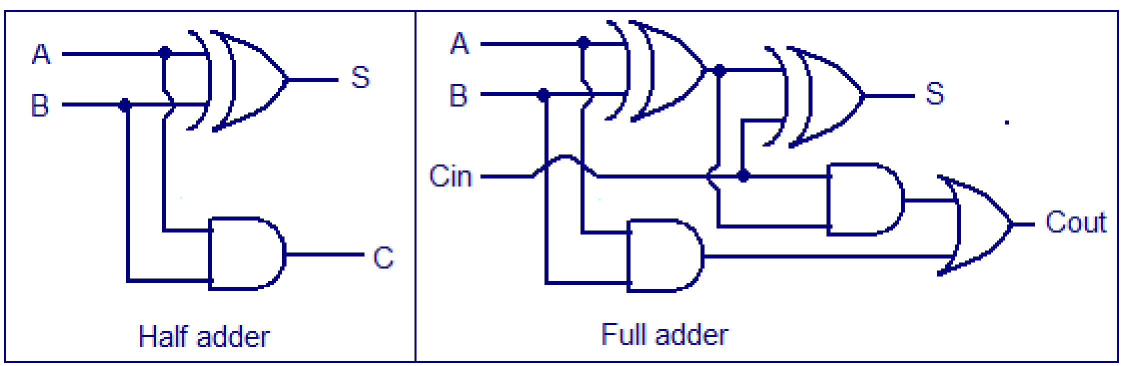


Sekedar pengetahuan, di bawah ini rangkaian untuk *full adder*



Full adder using 2-Half adder

Full Adder –Truth Table				
Input			Output	
A	B	Carry in	Sum	Carry
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



Sumber gambar: http://www.circuitstoday.com/ripple-carry-adder

4. Buatlah rangkaian logika yang menerima masukan dua-bit dan menghasilkan keluaran berupa kudrat dari masukan. Sebagai contoh, jika masukannya 11 (3 dalam sistem desimal), maka keluarannya adalah 1001 (9 dalam sistem desimal).

Penyelesaian:

Misalkan 2-bit masukan kita simbolkan dengan *xy*, dan kuadratnya (4-bit) kita simbolkan dengan *abcd*.

Tabel kebenaran:

Masukan		Keluaran			
w	*x*	*a*	*b*	*c*	*d*
0	0	0	0	0	0
0	1	0	0	0	1
1	0	0	1	0	0
1	1	1	0	0	1

		y = 0	y = 1
x	0	0	0
	1	0	1

$a(x, y) = xy$

		y = 0	y = 1
x	0	0	0
	1	1	0

$b(x, y) = xy'$

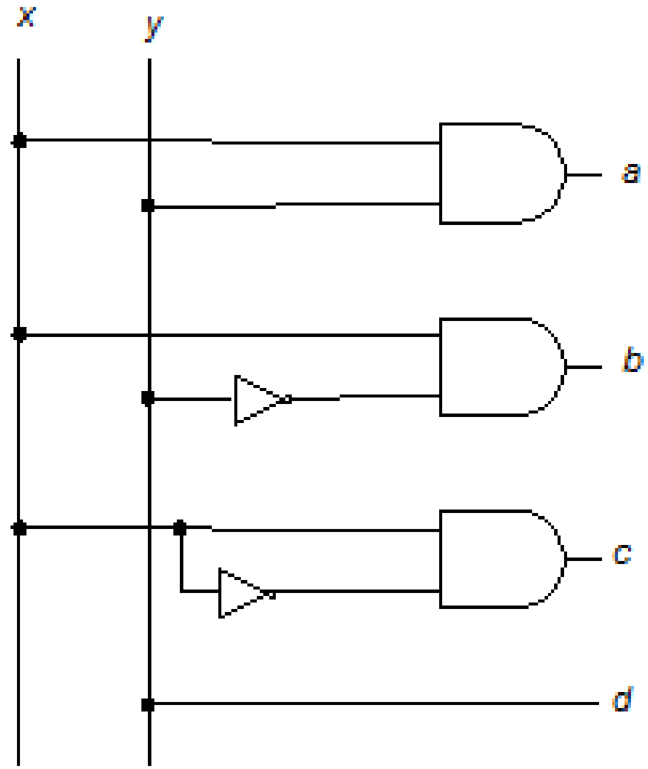
		y = 0	y = 1
x	0	0	0
	1	0	0

$c(x, y) = 0 = xx'$

		y = 0	y = 1
x	0	0	1
	1	0	1

$d(x, y) = y$

Rangkaian logikanya pengkuadrat 2-bit biner:



5. Sebuah instruksi dalam sebuah program adalah

if A > B **then writeln**(A) **else writeln**(B);

Nilai *A* dan *B* yang dibandingkan masing-masing panjangnya dua bit (misalkan a_1a_2 dan b_1b_2).

(a) Buatlah rangkaian logika (yang sudah disederhanakan tentunya) yang menghasilkan keluaran 1 jika *A* > *B* atau 0 jika tidak.

(b) Gambarkan kembali rangkaian logikanya jika hanya menggunakan gerbang *NAND* saja (petunjuk: gunakan hukum de Morgan)

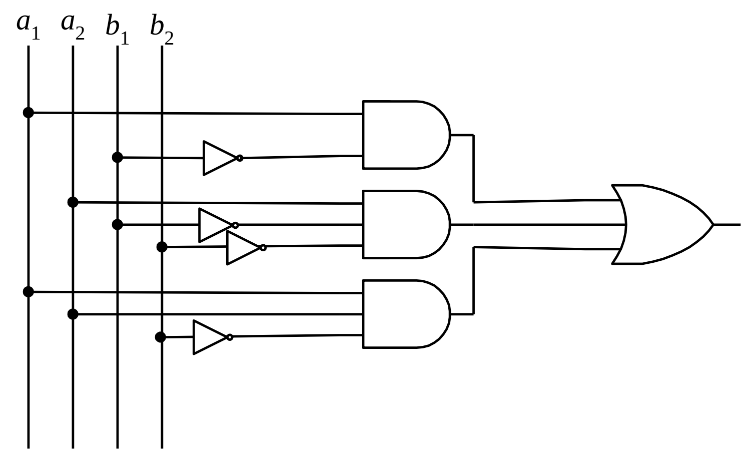
Penyelesaian:

(a)

Desimal		Biner				
A	B	a_1	a_2	b_1	b_2	$f(a_1, a_2, b_1, b_2)$
0	0	0	0	0	0	0
0	1	0	0	0	1	0
0	2	0	0	1	0	0
0	3	0	0	1	1	0
1	0	0	1	0	0	1
1	1	0	1	0	1	0
1	2	0	1	1	0	0
1	3	0	1	1	1	0
2	0	1	0	0	0	1
2	1	1	0	0	1	1
2	2	1	0	1	0	0
2	3	1	0	1	1	0
3	0	1	1	0	0	1
3	1	1	1	0	1	1
3	2	1	1	1	0	1
3	3	1	1	1	1	0

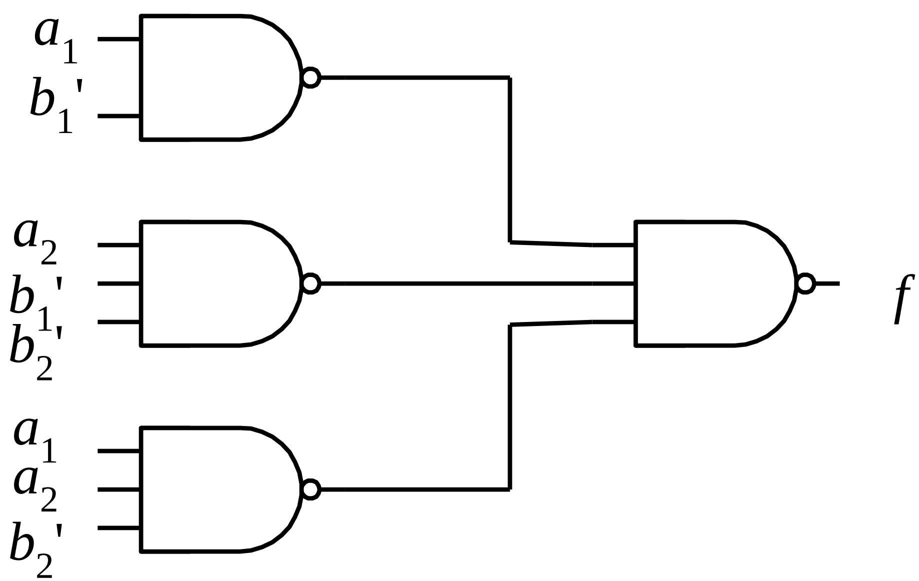
b_1b_2	00	01	11	10
00	0	0	0	1
01	1	0	0	0
11	1	1	0	1
10	1	1	0	0

$f(a_1, a_2, b_1, b_2) = a_1b_1' + a_2b_1'b_2' + a_1a_2b_2'$



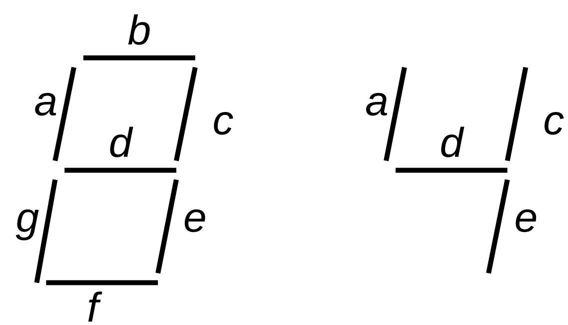
(b) $f(a_1, a_2, b_1, b_2) = a_1b_1' + a_2b_1'b_2' + a_1a_2b_2'$
$= ((a_1b_1')' (a_2b_1'b_2')' (a_1a_2b_2')')'$ (De Morgan)

Rangkaian logika:



Latihan

Sebuah Peraga angka digital disusun oleh tujuh buah segmen (selanjutnya disebut *dekoder tujuh-segmen*).



dekoder 7-segmen   angka 4

Piranti tersebut mengubah masukan 4-bit menjadi keluaran yang dapat menunjukkan angka desimal yang dinyatakannya (misalnya, jika masukan adalah 0100 (angka 4 dalam desimal), maka batang/segmen yang menyala adalah a, d, c, dan e). Tulislah fungsi Boolean untuk setiap segmen, dan gambarkan rangkaian kombinasionalnya.