Sistem Digital

Sistem Angka dan konversinya

Sistem angka yang biasa kita kenal adalah system decimal yaitu system bilangan berbasis 10, tetapi system yang dipakai dalam computer adalah biner.

Sistem Biner adalah system bilangan yang hanya menggunakan dua symbol (0,1). Bilangan ini biasanya dikatakan mempunyai radiks 2 dan biasa disebut bilangan berbasis 2, setiap biner digit disebut bit.

Mengapa menggunakan system Biner ?

- Penggunaan system angka-biner pada dasarnya disebabkan karena kesederhanaan cara, dimana digit biner 0 dan 1 berhubungan dengan implementasi fisis. Digit biner 0 dan 1 dapat dengan mudah dinyatakan oleh tegangan komponen digital sebagai rendah (low) atau tinggi (high)
- System biner hanya dapat mengolah angka biner atau angka terkode biner dari system bilangan lain seperti decimal. Pembatasan semua dari system digital (biner) ini mengakibatkan bahwa angka-angka yang diberikan dalam bentuk lain harus di konversi kan ke bentuk biner dahulu sebelum diolah oleh suatu system digital pada akhir proses hasilnya (dalam bentuk biner) dapat dikonversikan kembali ke bentuk system angka aslinya.

Setiap angka integral N dan n digit dari baris r dapat dinyatakan sebagai berikut:

$$N_r = a_n r^n + a_{n-1} r^{n-1} + \ldots + a_1 r^1 + a_0 r^0 = \sum_{k=0}^{n} a_k r^k$$

Dimana a_i , i = 0,1,2,3,….,n,adalah digit dalam posisi ke (i +1) dari kanan. Untuk baris r, $a_i \in \{0,1,2,3,\ldots,r-1\}$

System angka decimal, octal, dan biner sbb;

Sistem angka	Baris / radiks	Bentuk Umum	Contoh
Desimal	r = 10	$\sum_{k=0}^{n} a_k 10^k$	1975_{10}
Biner	r = 2	$\sum_{k=0}^{n} b_k 2^k$	$1975_{10} = 11110110111_2$
Oktal	r = 8	$\sum_{k=0}^{n} c_k 8^k$	$1975_{10} = 3667_8$

Konversi Desimal ke biner :

Metode Cibar-Cibur (The Dibble-Dabble Method)

Banyak cara yang digunakan untuk mengkonversikan angka decimal ke angka biner dan angka biner ke angka decimal ekivalennya, akan tetapi yang paling popular adalah metode cibar-cibur (the dibble-dabble method). Cara yang dipakai untuk mengkonversi bilangan decimal ke biner dengan pembagian ulang angka decimal oleh 2, menghasilkan deretan dari sisa 0 atau 1. Deretan sisa tersebut bila dibaca dari arah terbalik akan menghasilkan angka biner ekivalen dari angka decimal yang di konversikan

Contoh : konversikan **1975_{10}** = ……$_2$

2|1975	**sisa**
2|987	**1**
2|493	**1**
2|246	**1**
2|123	**0**
2|61	**1**
2|30	**1**
2|15	**0**
2|7	**1**
2|3	**1**
2|1	**1**
0	**1**

dibaca terbalik, dari bawah ke atas

$1975_{10} = 11110110111_2$

Konversi Biner ke Desimal

Konversikan **110111_2** = ……$_{10}$

$$110111_2 = \begin{array}{cccccc} 1 & 1 & 0 & 1 & 1 & 1 \\ \hline 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \end{array} \text{ x}$$

$$32 + 16 + 0 + 4 + 2 + 1 = 55_{10}$$

Konversi octal ke biner

Konversi angka octal ke biner dapat dikerjakan dengan mengkonversi masing-masing bit dari angka octal ke angka biner 3-bit, kemudian tinggal menderetkan secara berurutan.

Contoh : konversikan 3667_8 = ……..$_2$

3 6 6 7

011 | 110 | 110 | 111

$3667_8 = 11110110111_2$

Konversi biner ke octal

Cara konversi biner ke octal adalah dengan membagi deretan bilangan biner ke dalam 3-bit biner kemudian mengkonversi masing-masing 3- bit biner tadi ke bilangan octal

Contoh : konversikan 10011100111001_2 =……$_8$

010 | 011 | 100 | 111 | 001

2 3 4 7 1

$10011100111001_2 = 23471_8$

Konversi decimal ke octal

Konversi decimal ke octal dapat dilakukan dengan metode cibar-cibur. Dapat juga dilakukan dengan terlebih dahulu mengkonversi decimal ke biner, kemudian dari biner ke octal.

Bilangan Hexadesimal

Bilangan yang mempunyai radiks 16 atau system bilangan berbasis 16, bilangan hexadecimal menggunakan symbol 0-9, A untuk cacahan 10, B untuk cacahan 11,C untuk cacahan 12, D untuk cacahan 13, E untuk cacahan 14 ,dan F untuk cacahan 15.

Keuntungan dari system hexadecimal adalah kegunaannya dalam pengubahan secara langsung dari bilangan biner 4-bit. Tiap bilangan biner 4-bit dari 0000 sampai 1111 dapat diwakili oleh suatu digit hexadecimal yang unik.

Contoh : Konversikan $\mathbf{2B6_{16}}$ = ……$_{10}$

B = 11

$\mathbf{2B6_{16}}$ = **2 11 6**

16^2 16^1 16^0 x

512 + 176 + 6 = 694_{10}

Konversikan $\mathbf{45_{10}}$ = ……$_{16}$

16| 45 sisa

16| 2 13 → dalam hexadecimal direpresentasikan dengan D

0 2

45_{10} = $2D_{16}$

konversikan **$2B6_{16}$** = ……$_{2}$

2 B 6

0010 1011 0110 = 1010110110_2

Aritmatika Biner

Penambahan Biner ,

Aturan dalam penambahan biner

Masukan				keluaran	
A		**B**		**Jumlah**	**Carry Out (Co)**
0	+	**0**	=	**0**	**0**
0	+	**1**	=	**1**	**0**
1	+	**0**	=	**1**	**0**
1	+	**1**	=	**0**	**1**

Contoh **:**

		1	**0**	**0**	**4**			**1**	**0**	**1**	**5**
	+	**0**	**1**	**0**	**+ 2**	**+**		**0**	**1**	**1**	**+3**
		1	**1**	**0**	**6**		**1**	**0**	**0**	**0**	**8**

1 1 1 ◄— Co

Pengurangan Biner,

Aturan dalam pengurangan Biner

Masukan				keluaran	
A		**B**		**Selisih**	**Borrow Out (Bo)**
0	-	**0**	=	**0**	**0**
0	-	**1**	=	**1**	**1**
1	-	**0**	=	**1**	**0**
1	-	**1**	=	**0**	**0**

Contoh **:**

		1	**0**	**0**	**4**			**1**	**0**	**1**	**5**
	-	**0**	**1**	**0**	**- 2**	**-**		**0**	**1**	**1**	**- 3**
		0	**1**	**0**	**2**			**0**	**1**	**0**	**2**

1 ◄— Bo **1 ◄— Bo**

System Angka

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

Representasi dan penambahan dari angka biner bertanda (Signed binary number)

Suatu angka biner bertanda n-bit terdiri dari dua bagian : bagian yang menyatakan tanda dari angka dan bagian yang menyatakan besaran (magnitude). Bit pertama dari angka disebut *bit tanda*, yang menyatakan tanda dari angka , dimana 0 menyatakan bahwa " angka adalah positip " dan 1 menyatakan bahwa "angka adalah negatif "

Terdapat beberapa cara untuk menyatakan besaran dari angka bertanda dalam system digital. Tiga bentuk dari angka (biner) bertanda yang popular adalah :

1. **Sistem angka besaran bertanda (signed-magnitude number system)**
   Dalam bentuk ini angka positip dan angka negatip dinyatakan dengan suatu bit tanda diikuti oleh besaran dalam biner

Contoh : **+ 15** **0** **1111** **= 01111**

Bit tanda **besaran (magnitude)**

- 15 **1** **1111** **= 11111**

2. **Sistem Angka komplemen bertanda-1 (signed-1's complement number system)**

   Angka positip dalam system ini sama dengan angka positip dalam system angka besaran bertanda, akan tetapi angka negatipnya berbeda, yang dinyatakan dalam komplemen-1 (semua bit biner di representasikan terbalik, 0 ke 1 dan 1 ke 0)

   Contoh **: + 15 = 01111**

   - 15 = 10000

3. **Sistem Angka komplemen bertanda-2 (Signed-2's complement number system)**

   Dalam system ini angka positip dinyatakan dalam bentuk yang sama seperti dalam dua system angka sebelumnya, sedangkan angka negatipnya dinyatakan dalam bentuk komplemen-2

   Contoh : **+ 15 = 01111**

   - 15 = 10000 → 1's complement

   1

   10001 → 2's complement

Kode biner berbobot

Kode BCD (Binary Coded Decimal)

	BCD			
Desimal	**8-an**	**4-an**	**2-an**	**1-an**
0	**0**	**0**	**0**	**0**
1	**0**	**0**	**0**	**1**
2	**0**	**0**	**1**	**0**

3	**0**	**0**	**1**	**1**
4	**0**	**1**	**0**	**0**
5	**0**	**1**	**0**	**1**
6	**0**	**1**	**1**	**0**
7	**0**	**1**	**1**	**1**
8	**1**	**0**	**0**	**0**
9	**1**	**0**	**0**	**1**

BCD 8421

Contoh : konversikan $\mathbf{150_{10}}$ = ……..$_2$

$\mathbf{150_{10}}$ = **1 5 0**

0001 0101 0000 → $\mathbf{101010000_2}$

Kode Biner tak berbobot

Kode xs3 (exses 3),

Kode ekses 3 berhubungan dengan BCD 8421 disebabkan oleh sifat biner terkode desimalnya, dengan kata lain masing- masing kelompok 4 bit dalam kode XS3 sama dengan suatu digit decimal tertentu, XS3 selalu tiga angka lebih besar daripada BCD 8421.

	8421	**BCD**	**XS3**	**BCD**
Desimal	**10-an**	**1-an**	**10-an**	**1-an**
0		**0000**	**0011**	**0011**
1		**0001**	**0011**	**0100**
2		**0010**	**0011**	**0101**
3		**0011**	**0011**	**0110**
4		**0100**	**0011**	**0111**
5		**0101**	**0011**	**1000**
6		**0110**	**0011**	**1001**
7		**0111**	**0011**	**1010**
8		**1000**	**0011**	**1011**
9		**1001**	**0011**	**1100**
10	**0001**	**0000**	**0100**	**0011**
11	**0001**	**0001**	**0100**	**0100**

Contoh : **62_{10}** =……**XS3**

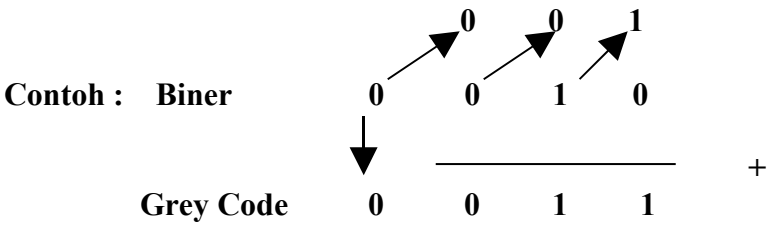
	6	**2**		
	+3	**+3**		**tiap digit tambah dengan 3**
	9	**5**		
	↓	↓		**Ubah ke Biner (BCD XS3)**
	1001	**0101**	=	**10010101 XS3**

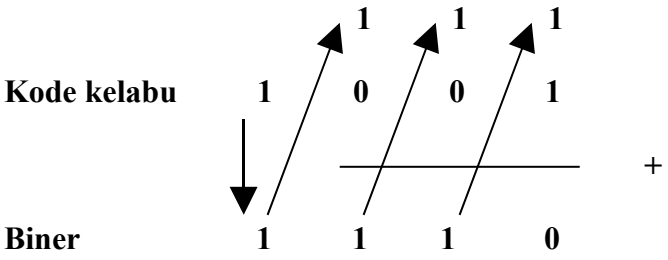
XS3	**1000**	**1100**	
	-0011	**- 0011**	**kurangi dengan 3**
BCD	**0101**	**1001**	
	↓	↓	
Decimal	**5**	**9**	**ubah ke decimal**

Kode Kelabu (Grey code)
Kode biner yang tak berbobot, kode kelabu bukan merupakan kode jenis BCD.
Kenaikan hitungan dilakukan hanya dengan pengubahan keadaan satu bit saja.

Desimal	Biner	Kode kelabu	Biner	Biner	Kode kelabu
0	**0000**	**0000**	**8**	**1000**	**1100**
1	**0001**	**0001**	**9**	**1001**	**1101**
2	**0010**	**0011**	**10**	**1010**	**1111**
3	**0011**	**0010**	**11**	**1011**	**1110**
4	**0100**	**0110**	**12**	**1100**	**1010**
5	**0101**	**0111**	**13**	**1101**	**1011**
6	**0110**	**0101**	**14**	**1110**	**1001**
7	**0111**	**0100**	**15**	**1111**	**1000**

Grey Code





Kode Alfa Numerik,

Kode Alfa numeric adalah kode yang dapat menyatakan baik angka maupun huruf.

Bit-bit dapat juga dikodekan untuk menyatakan huruf-huruf alphabet, bilangan dan tanda baca, salah satu kode 7-bit seperti itu adalah Kode Standard Amerika untuk pertukaran Informasi (American Standard Code for Information Interchange, ASCII)

Kode – kode yang lain adalah :

1.7-bit BCDIC (Binary Coded Decimal interchange Code)

2.8-bit EBCDIC (Extended BCDIC)

3.7-bit selektrik, digunakan untuk mengontrol perputaran bola pada mesin tik IBM

4.12-bit Hollerith, digunakan pada kartu kertas.

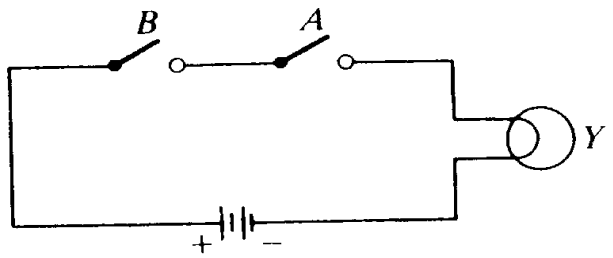
Gerbang Digital

Gerbang logika (logic gate) merupakan dasar pembentuk system digital. Gerbang Logika merupakan rangkaian elektronika, gerbang berfungsi untuk mengontrol arus informasi, biasanya dalam bentuk pulsa tegangan.

Gerbang AND

Disebut juga gerbang " Semua atau tidak satu pun "

Dalam rangakaian di bawah ini . Lampu (Y) hanya akan menyala jika kedua saklar masukan (A dan B) tertutup. Semua kemungkinan kombinasi untuk saklar A dan B di tunjukkan dalam table kebenaran.

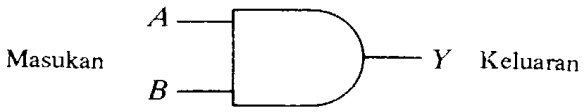


(*a*) Rangkaian AND yang menggunakan saklar

Saklar masukan		Nyala keluaran
B	*A*	*Y*
buka	buka	tidak
buka	tutup	tidak
tutup	buka	tidak
tutup	tutup	ya

(*b*) Tabel kebenaran

Simbol Logika Standard untuk gerbang AND



(*a*) Simbl gerbang-AND

Masukan		Keluaran
B	*A*	*Y*
0	0	0
0	1	0
1	0	0
1	1	1

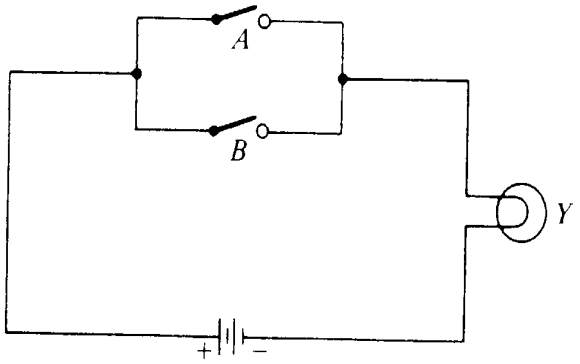
0 = tegangan rendah
1 = tegangan tinggi

(*b*) Tabel kebenaran AND

Pernyataan bolean untuk gerbang AND di atas : A . B = Y atau AB = Y

Gerbang OR

Sering disebut gerbang " Setiap atau semua " , dalam rangkain di gambar , Lampu (Y) akan menyala bila saklar A atau B tertutup

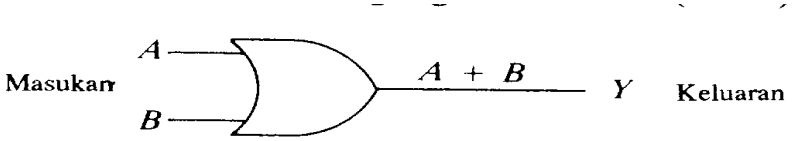


(*a*) Rangkaian OR yang menggunakan saklar

Saklar masukan		Nyala keluaran
B	*A*	*Y*
buka	buka	tidak
buka	tutup	ya
tutup	buka	ya
ttup	tutup	ya

(*b*) Tabel kebenaran

Simbol Standard Gerbang Logika OR



(*a*) Simbol gerbang-OR

Masukan		Keluaran
B	*A*	*Y*
0	0	0
0	1	1
1	0	1
1	1	1

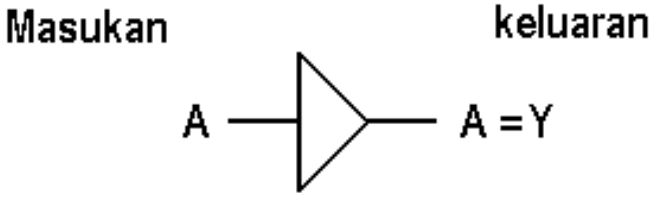
0 = tegangah rendah
1 = tegangan tinggi

(*b*) Tabel kebenaran OR

Pernyataan Bolean untuk gerbang OR : A + B = Y

Buffer

Mempunyai satu masukan dan satu keluaran, dimana output selalu sama dengan input

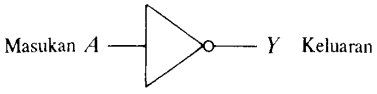


Masukan	Keluaran
A	Y
0	0
1	1

Tabel Kebenaran Buffer

Gerbang Not (Inverter)

Disebut juga pembalik, hanya mempunyai satu masukan dan satu keluaran, dimana output selalu merupakan kebalikan inputnya.



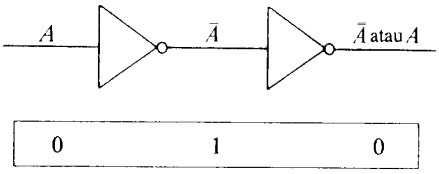
(*a*) Simbol gerbang-NOT

Masukan	Keluatan
A	*Y*
0	1
1	0

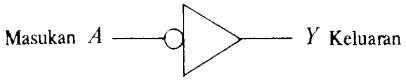
(*b*) Tabel kebenaran gerbang-NOT

$\overline{\overline{A}} = A$

(*c*) Pernyataan Boolean NOT

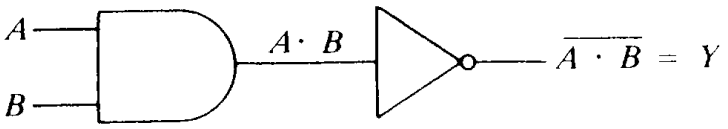


(*d*) Inversi ganda

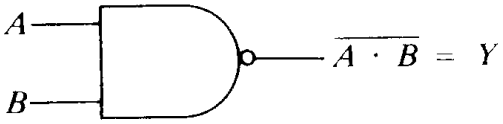


(e) Alternative inverter symbol

Gerbang NAND (Not – AND)



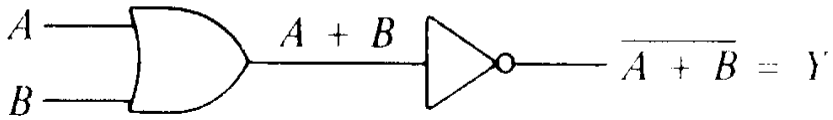
Masukan Keluaran



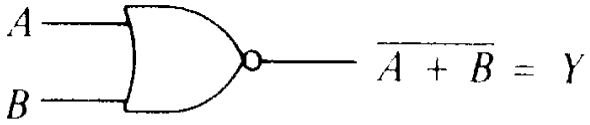
Masukan		Keluaran	
B	*A*	AND	NAND
0	0	0	1
0	1	0	1
1	0	0	1
1	1	1	0

Tabel kebenaran Gerbang NAND

Gerbang NOR (Not – OR)



Masukan Keluaran



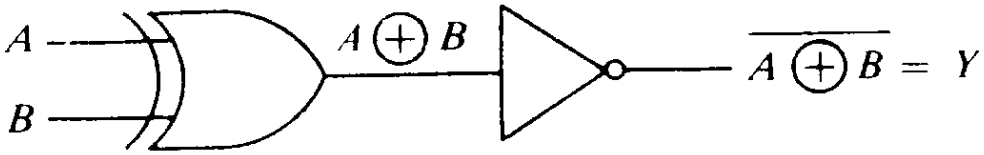
Masukan		Keluaran	
B	*A*	OR	NOR
0	0	0	1
0	1	1	0
1	0	1	0
1	1	1	0

Tabel Kebenaran Gerbang NOR

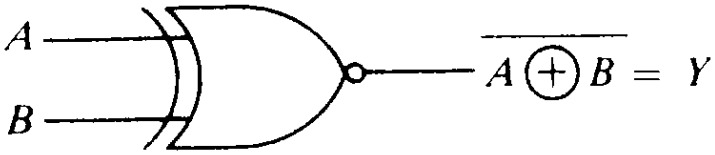
Gerbang XOR (OR – Ekslusif)

Simbol logika standar untuk gerbang XOR

Gerbang XNOR (NOR – ekslusif)



Masukan Keluaran

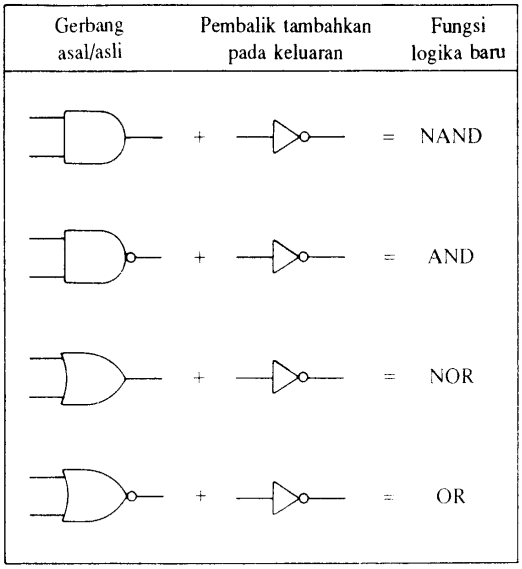


Gerbang XNOR

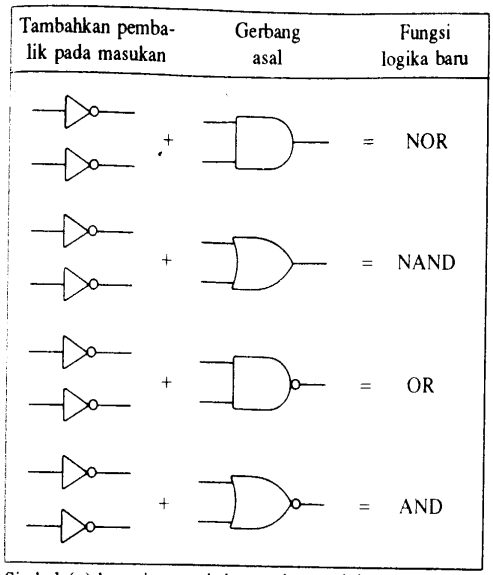
Masukan		Keluaran	
B	*A*	XOR	XNOR
0	0	0	1
0	1	1	0
1	0	1	0
1	1	0	1

Tabel Kebenaran gerbang-XOR dan gerbang-XNOR

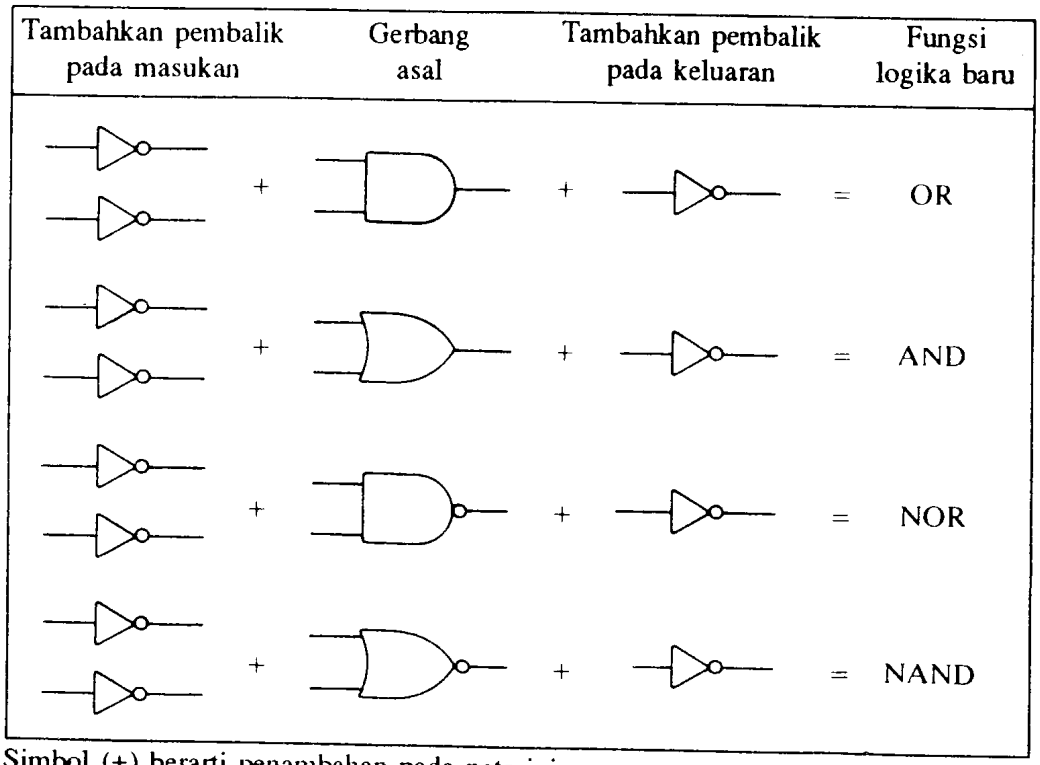
Pengubahan Gerbang dengan menggunakan pembalik



Simbol (+) berarti penambahan pada peta ini

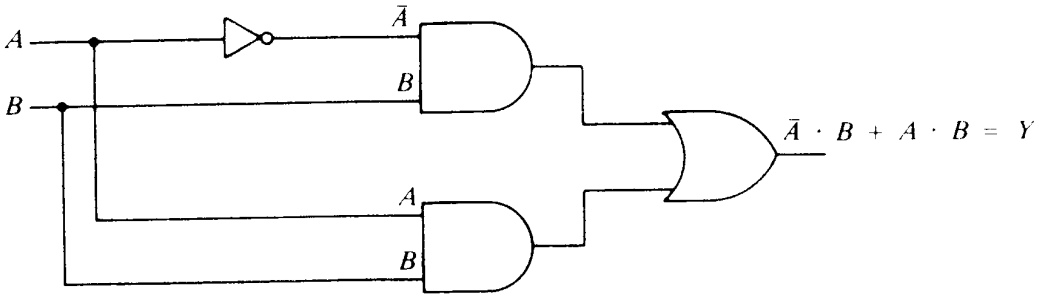


Simbol (+) berarti penambahan pada peta ini.

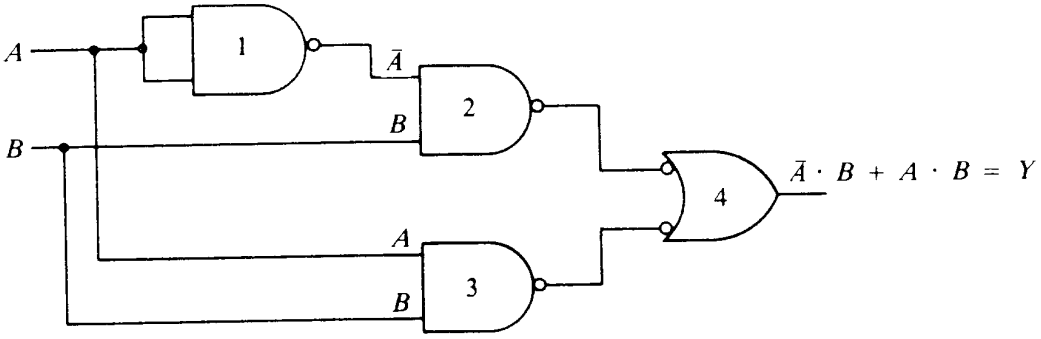


Simbol (+) berarti penambahan pada peta ini

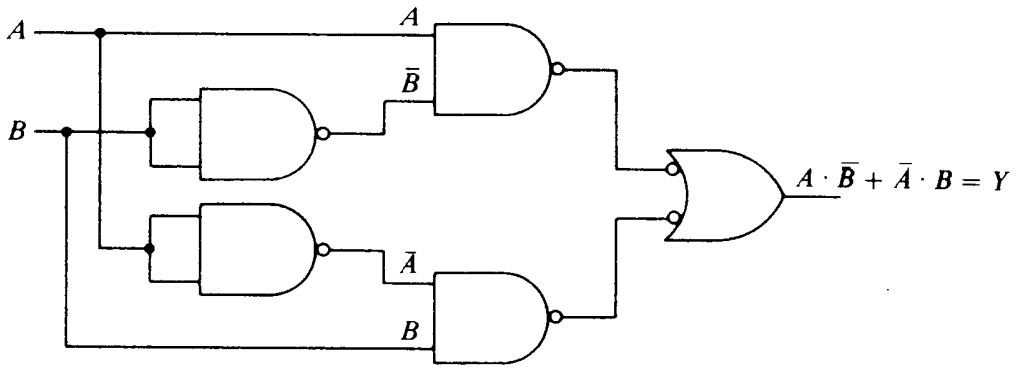
Kombinasi Gerbang Logika

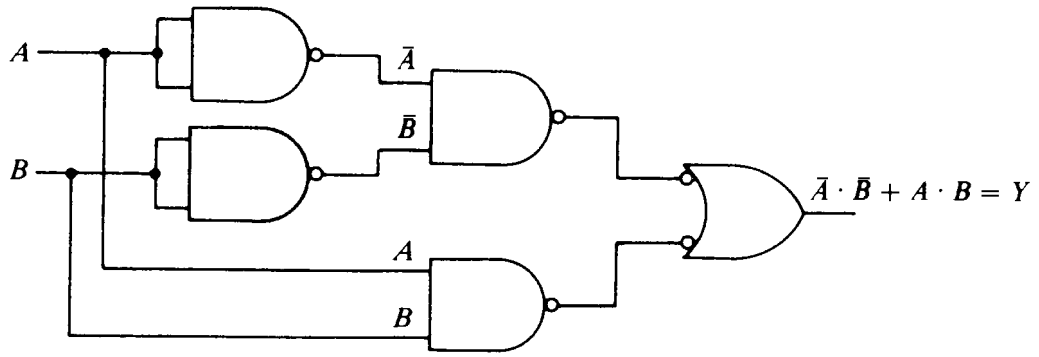


(*a*) Rangkaian logis AND-OR



(*b*) Rangkain logika NAND ekivalen





Perancangan Rangkaian logika

Prosedur yang lazim pada perancangan logika al :

1.Menyusun tabel kebenaran

2.Menyatakan Aljabar Boolean yang ditentukan dari table kebenaran.

3.Perancangan rangkaian logika dari pernyataan Boolean

Aljabar Boolean Jumlah Dari Perkalian

Sering disebut sebagai bentuk MINTERM

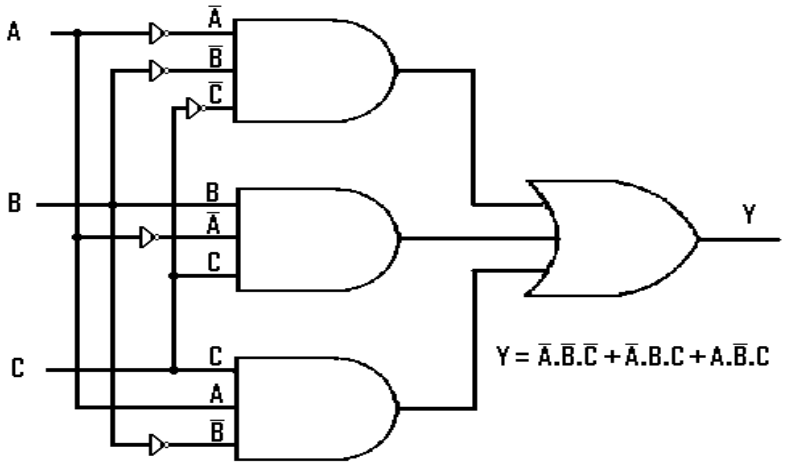
- Perhatikan semua kombinasi masukan yang menghasilkan keluaran 1 (satu)
- Operasi AND-kan setiap masukan yang menghasilkan keluaran 1 (satu)
- Operasi OR-kan semua kemungkinan kombinasi masukan untuk membentuk Aljabar Boolean yang lengkap

Contoh :

Diberikan tabel kebenaran sebagai berikut ,Tuliskan pernyataan Boolean minterm rancangan rangkaian logika nya :

A	B	C	Y	
0	0	0	1	→ $\bar{A}.\bar{B}.\bar{C}$
0	0	1	0	
0	1	0	0	
0	1	1	1	→ $\bar{A}.B.C$
1	0	0	0	
1	0	1	1	→ $A.\bar{B}.C$
1	1	0	0	
1	1	1	0	

Pernyataan Boolean : $\bar{A}.\bar{B}.\bar{C}+\bar{A}.B.C+A.\bar{B}.C$



Rangkaian logika dari pernyataan Boolean $\bar{A}.\bar{B}.\bar{C}+\bar{A}.B.C+A.\bar{B}.C$

Aljabar Boolean Perkalian Dari Jumlah

Sering disebut sebagai bentuk MAKSTERM

- Perhatikan semua kombinasi masukan yang menghasilkan keluaran 0 (nol)
- Operasi OR-kan setiap masukan yang menghasilkan keluaran 0 (nol)
- Operasi AND-kan semua kemungkinan kombinasi masukan untuk membentuk Aljabar Boolean yang lengkap

Contoh :

Diberikan tabel kebenaran sebagai berikut ,Tuliskan pernyataan Boolean minterm rancangan rangkaian logika nya :

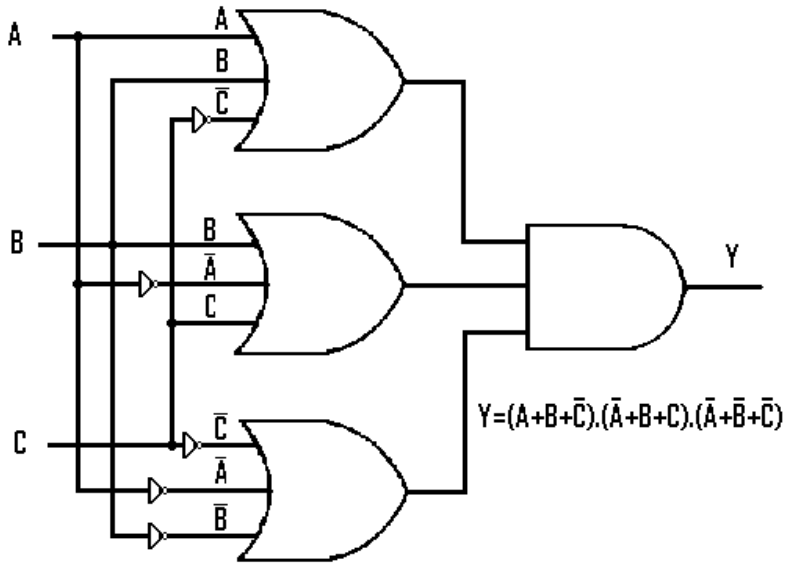
A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

→ $A+B+\bar{C}$

→ $\bar{A}.B.C$

→ $\bar{A}+\bar{B}+\bar{C}$

Pernyataan Boolean : $(A+B+\bar{C}).(\bar{A}+B+C).(\bar{A}+\bar{B}+\bar{C})$



Rangkain Logika pernyataan Boolean: $(A+B+\bar{C}).(\bar{A}+B+C).(\bar{A}+\bar{B}+\bar{C})$

Boolean Postulates in 0 and 1

OR	AND	NOT
0 + 0 = 0	0 · 0 = 0	$\bar{0} = 1$
0 + 1 = 1	0 · 1 = 0	$\bar{1} = 0$
1 + 0 = 1	1 · 0 = 0	
1 + 1 = 1	1 · 1 = 1	

Boolean Theorems in One Variable

OR	AND	NOT
A + 0 = A	A · 0 = 0	$\bar{\bar{A}} = A$
A + 1 = 1	A · 1 = A	
A + A = A	A · A = A	
$A + \bar{A} = 1$	$A \cdot \bar{A} = 0$	

Boolean Theorems in More Than One Variable

Commutation rules:
A + B = B + A
A · B = B · A

Association rules:
A + (B + C) = (A + B) + C
A · (B · C) = (A · B) · C

DeMorgan's theorems:
$\overline{A + B} = \bar{A} \cdot \bar{B}$
$\overline{A \cdot B} = \bar{A} + \bar{B}$

Absorption rules:
A + (A · B) = A
A · (A + B) = A

Distribution rules:
A · (B + C) = (A · B) + (A · C)
A + (B · C) = (A + B) · (A + C)

Teori De Morgan

(*a*) Fungsi NOR (*b*) Fungsi NAND

Untuk pengubahan situasi And menjadi OR atau sebaliknya diperlukan empat langkah yang berdasarkan teori De-Morgan

1.Ubah semua OR ke AND dan semua AND ke OR

2.Lengkapi setiap Variabel individual (Tambahkan tanda strip diatas tiap variabel)

3.lengkapi semua fungsi (Tambahkan tanda strip diatasnya)

4.Hilangkan semua kelompok dari tanda – strip di atas yang berjumlah genap

Contoh : penggunaan teori DeMorgan

Pernyataan minterm $\overline{\overline{C} \cdot \overline{B} \cdot \overline{A} + C \cdot B \cdot \overline{A}} = Y$

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

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

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

keempat Hilangkan strip di atas yang berjumlah genap

Pernyataan maksterm $(C + B + A) \cdot (\overline{C} + \overline{B} + A) = Y$

Pernyataan maksterm $(C + B + A) \cdot (\overline{C} + \overline{B} + A) = Y$

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

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

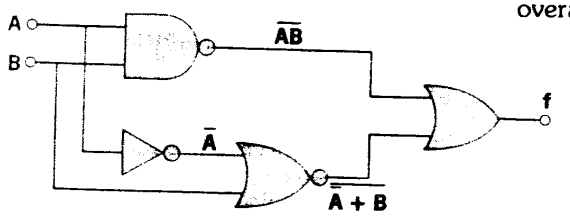
ketiga $\overline{(\overline{C} \cdot \overline{B} \cdot \overline{A}) + (\overline{\overline{C}} \cdot \overline{\overline{B}} \cdot \overline{A})}$

keempat Hilangkan strip di atas yang berjumlah genap

Pernyataan minterm $\overline{\overline{C} \cdot \overline{B} \cdot \overline{A} + C \cdot B \cdot \overline{A}} = Y$

Contoh 1

Analyze the logic circuit



The suboutputs are noted on the diagram. The overall function can be simplified as follows:

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

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

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

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

$= \overline{AB}$ (DeMorgan's rule)

A	B	$\overline{AB}$	$\overline{\overline{A} + B}$	f
0	0	1	0	1
0	1	1	0	1
1	0	1	1	1
1	1	0	0	0

Contoh 2

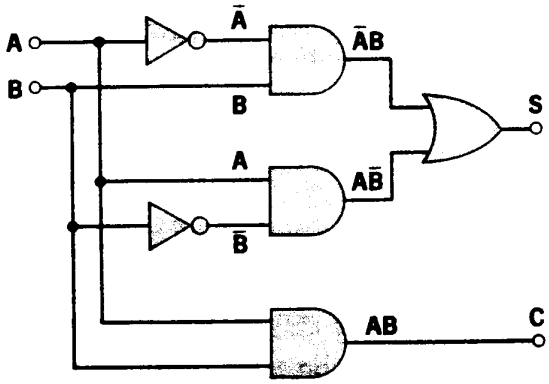
$$\bar{A}B + A\bar{B} = \overline{(A + \bar{B})(\bar{A} + B)} \quad \text{(DeMorgan's theorems)}$$

$$= \overline{A\bar{A} + AB + \bar{A}\bar{B} + B\bar{B}} \quad \text{(Multiplication)}$$

$$= \overline{\bar{A}\bar{B} + AB} \quad (A\bar{A} = 0 \text{ and } B\bar{B} = 0)$$

$$= (A + B)(\bar{A} + \bar{B}) \quad \text{(DeMorgan's theorems)}$$

$$= (A + B)\overline{AB} \quad \text{(De Morgan's theorems)}$$



A =	1	1	0	0	
+B =	1	0	1	0	
	1̄0	1̄	1̄	0̄	

(a) Addition

A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

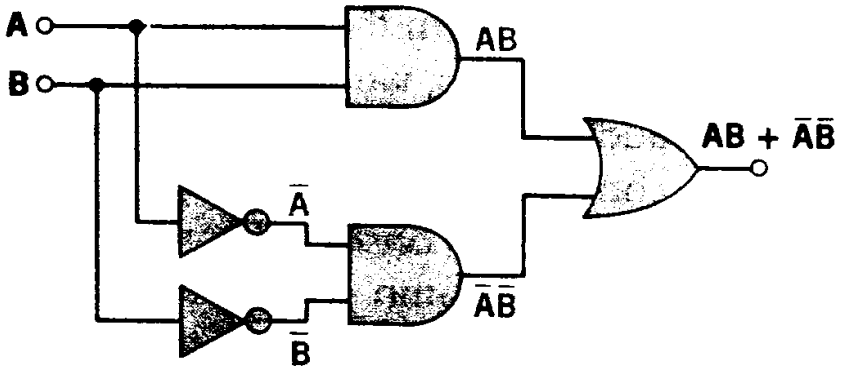
(b) Truth table

(c) Half-adder circuit

Contoh 3 :

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

This is an equality comparator in that the output is 1 if A and B are equal. This function is highly useful in digital computer operation. Straightforward synthesis results in the circuit

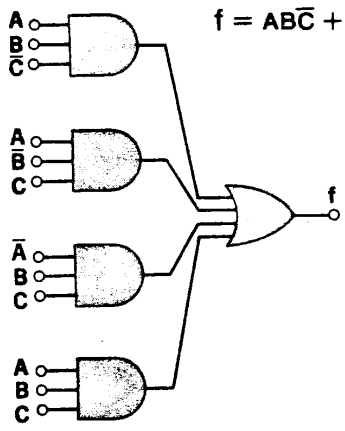


An equality comparator.

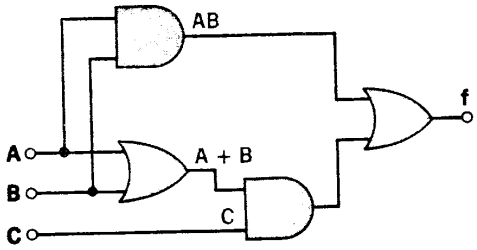
Contoh 4:

A	B	C	f
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

(a) Truth table



(b) Circuit realization



(c) After Boolean simplification

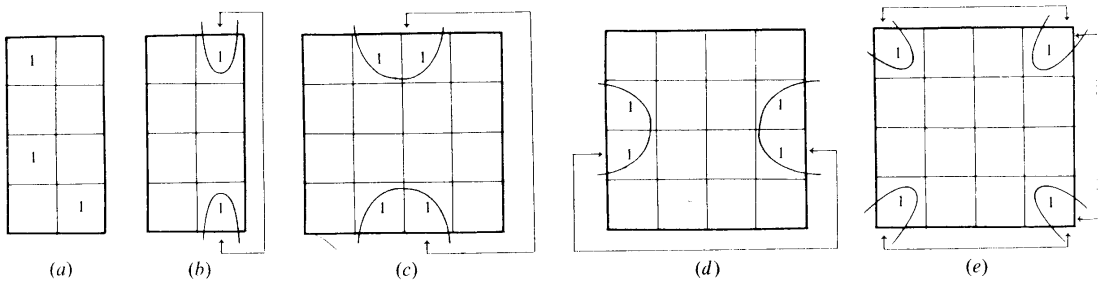
since $ABC + ABC = ABC$. Factoring by the distribution rule yields

$$f = AB(\bar{C} + C) + C(AB + A\bar{B} + \bar{A}B)$$

Since $\bar{C} + C = 1$ and $AB + A\bar{B} + \bar{A}B = A + B$, the function becomes

$$f = AB + C(A + B)$$

Peta Karnaugh, Metode penyederhanaan rangkaian logika



Beberapa variasi pelingkaran yang tidak biasa

Penggunaan Peta dengan Pernyataan Minterm

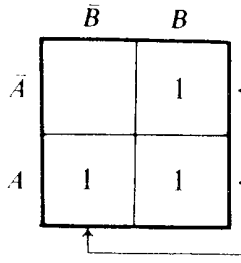
(*a*)

Masukan		Keluaran
A	*B*	*Y*
0	0	0
0	1	1
1	0	1
1	1	1

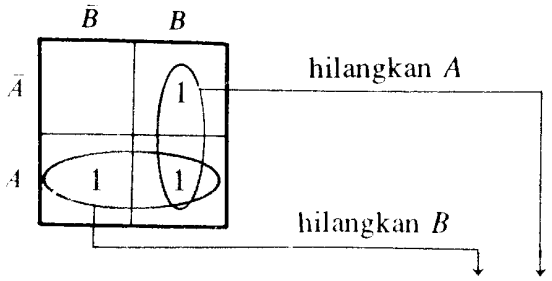
$\bar{A} \cdot B$
$A \cdot \bar{B}$
$A \cdot B$

(*b*) Pernyataan minterm: $A \cdot B + A \cdot \bar{B} + \bar{A} \cdot B = Y$

(*c*) Penggambaran satuan pada peta



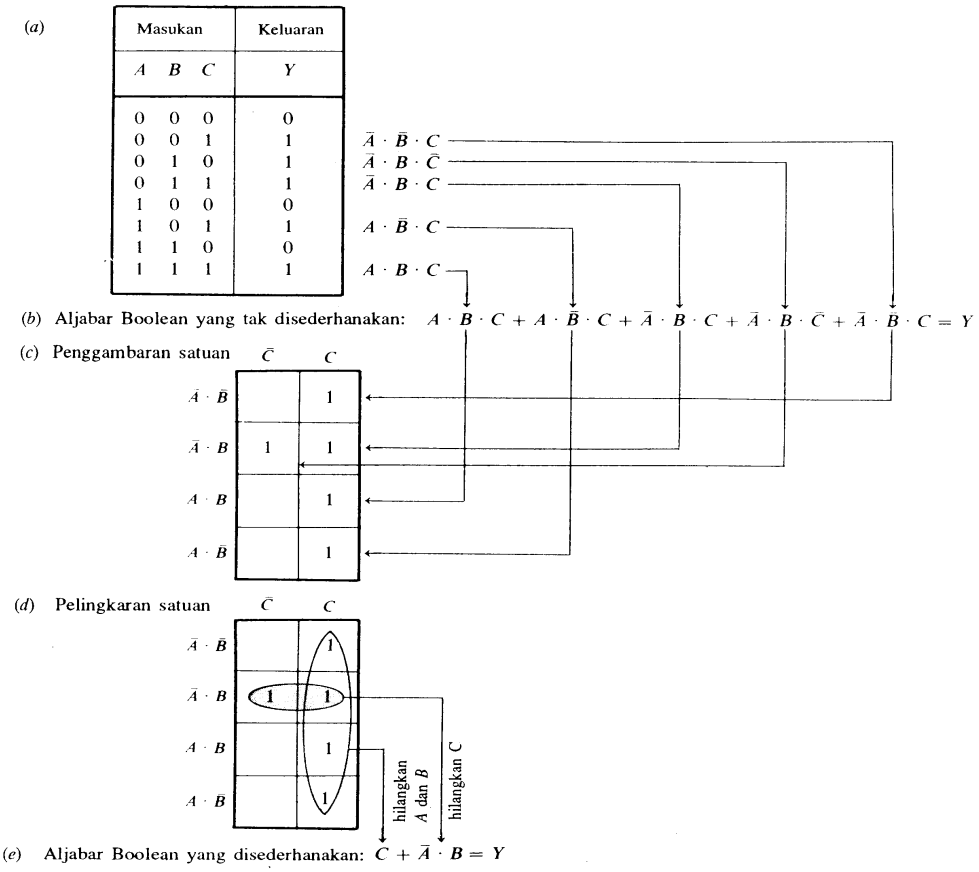
(*d*) Pelingkaran satuan



(*e*) Penghilangan variabel-variabel untuk membentuk aljabar Boolean yang disederhanakan: $A + B = Y$

(*a*)

Masukan			Keluaran
A	*B*	*C*	*Y*
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



(*b*) Aljabar Boolean yang tak disederhanakan: $A \cdot \bar{B} \cdot C + A \cdot \bar{B} \cdot C + \bar{A} \cdot B \cdot C + \bar{A} \cdot B \cdot \bar{C} + \bar{A} \cdot \bar{B} \cdot C = Y$

(*c*) Penggambaran satuan

(*d*) Pelingkaran satuan

(*e*) Aljabar Boolean yang disederhanakan: $\bar{C} + \bar{A} \cdot B = Y$

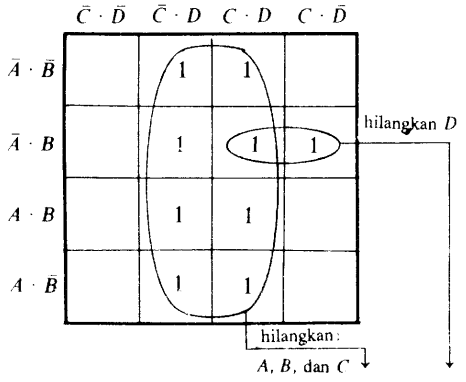
(*a*)

Masukan				Keluaran
A	*B*	*C*	*D*	*Y*
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

(*b*) Pernyataan minterm yang tak disederhanakan

$$\bar{A} \cdot \bar{B} \cdot \bar{C} \cdot D + \bar{A} \cdot \bar{B} \cdot C \cdot D + \bar{A} \cdot B \cdot \bar{C} \cdot D + \bar{A} \cdot B \cdot C \cdot \bar{D} + \bar{A} \cdot B \cdot C \cdot D + A \cdot \bar{B} \cdot \bar{C} \cdot D + A \cdot \bar{B} \cdot C \cdot D + A \cdot B \cdot \bar{C} \cdot D + A \cdot B \cdot C \cdot D = Y$$

(*c*) Penggambaran dan pelingkaran satuan pada peta



(*d*) Aljabar Boolean yang disederhanakan: $D + \bar{A} \cdot B \cdot C = Y$

Penggunaan Peta Karnaugh dengan pernyataanMaksterm

(*a*)

Masukan			Keluaran
A	B	C	Y
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

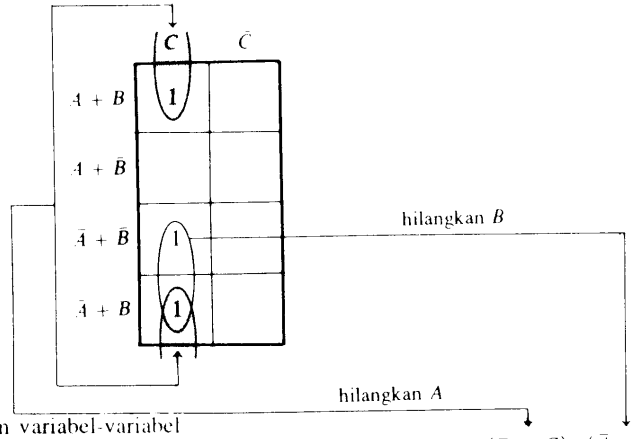
balik → $A + B + C$

balik → $\bar{A} + B + C$

balik → $\bar{A} + \bar{B} + C$

(*b*) Penyusunan pernyataan maksterm yang tak disederhanakan: $(A + \bar{B} + C) \cdot (A + B + C) \cdot (\bar{A} + \bar{B} + C) = Y$

(*c*) Pemetaan



(*d*) Penghilangan variabel-variabel untuk menghasilkan pernyataan yang disederhanakan: $(B + C) \cdot (\bar{A} + C) = Y$

Pemetaan dengan pernyataan maksterm.

Bilangan BCD (8421)				Ekivalen desimal
D	*C*	*B*	*A*	
8-an	4-an	2-an	1-an	
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	tidak digunakan
1	0	1	1	tidak digunakan
1	1	0	0	tidak digunakan
1	1	0	1	tidak digunakan
1	1	1	0	tidak digunakan
1	1	1	1	tidak digunakan

Gambar 5-43 Tabel bilangan BCD

Masukan				Keluaran	
D	*C*	*B*	*A*		
8-an	4-an	2-an	1-an	*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	1	$D \cdot \bar{C} \cdot \bar{B} \cdot A$
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	

Gambar 5-44

$D \cdot \bar{C} \cdot \bar{B} \cdot A = Y$

(*a*) Aljabar Boolean yang tidak disederhanakan

$D \cdot A = Y$

(*c*) Aljabar Boolean yang disederhanakan

Penggunaan suatu peta

	$\bar{C} \cdot \bar{D}$	$\bar{C} \cdot D$	$C \cdot D$	$C \cdot \bar{D}$
$\bar{A} \cdot \bar{B}$			X	
$\bar{A} \cdot B$		X	X	
$A \cdot B$		X	X	
$A \cdot \bar{B}$		1	X	

(*b*) Peta

Rangkaian Logika , terbagi atas

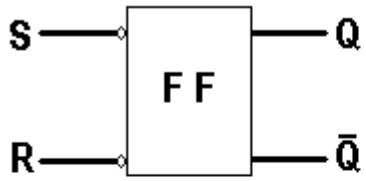
a.rangkaian logika kombinasional (rangkaian dasar nya gerbang logika)

b.Rangkaian logika sekuensial (rangkaian dasar nya Flip-flop)

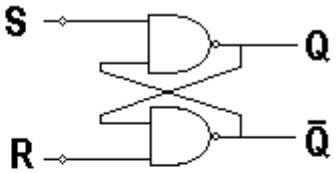
Flip – Flop (rangkaian logika yang dapat menghitung secara sekuen / berurutan dari nilai terkecil hingga nilai terbesar dan sebaliknya, Flip-Flop selalu mempunyai dua kondisi keluaran yang selalu dalam keadaan berlawanan Q dan komplemen Q, Flip dan Flop)

Macam – macam flip-flop

RS – FF (Reset Set Flip-flop)



Simbol

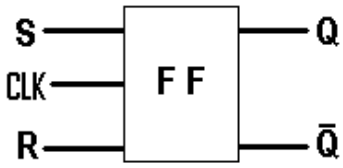


Rangkaian Dasar

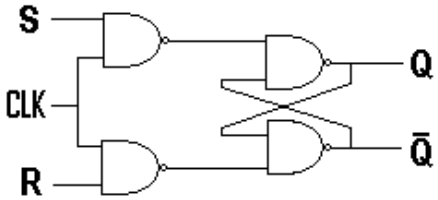
S	R	Q	Q̄
0	0	1	1
0	1	1	0
1	0	0	1
1	1	tetap	

Tabel kebenaran

Clocked RS – FF (RS FF yang beroperasi sinkron, berdasarkan pulsa detak / clock)



Simbol

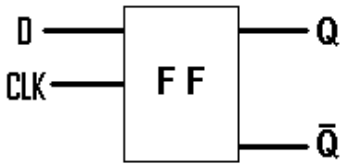


Rangkaian Dasar

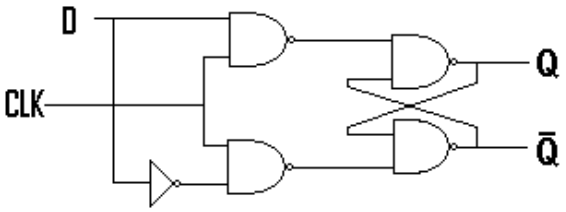
Mode operasi	Input			Output	
	CLK	S	R	Q	Q̄
Tetap	↑	0	0	tetap	
Reset	↑	0	1	0	1
Set	↑	1	0	1	0
Terlarang	↑	1	1	1	1

Tabel kebenaran

D – FF (Data / Delay Flip Flop)



Simbol

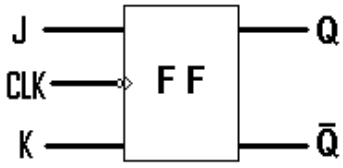


Rangkaian Dasar

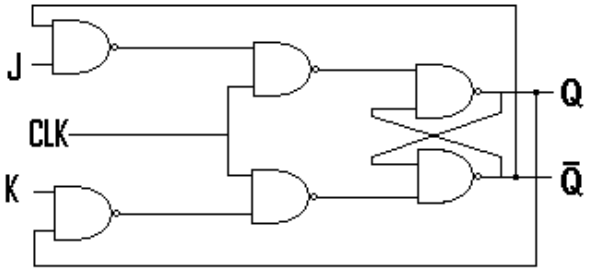
Mode operasi	Input				Output	
	Asinkron		Sinkron			
	PR	CLR	CLK	D	Q	Q̄
Set Asinkron	0	1	X	X	1	0
Reset Asinkron	1	0	X	X	0	1
Terlarang	0	0	X	X	1	1
Set	1	1	↑	1	1	0
Reset	1	1	↑	0	0	1

Tabel kebenaran

JK – FF



Simbol

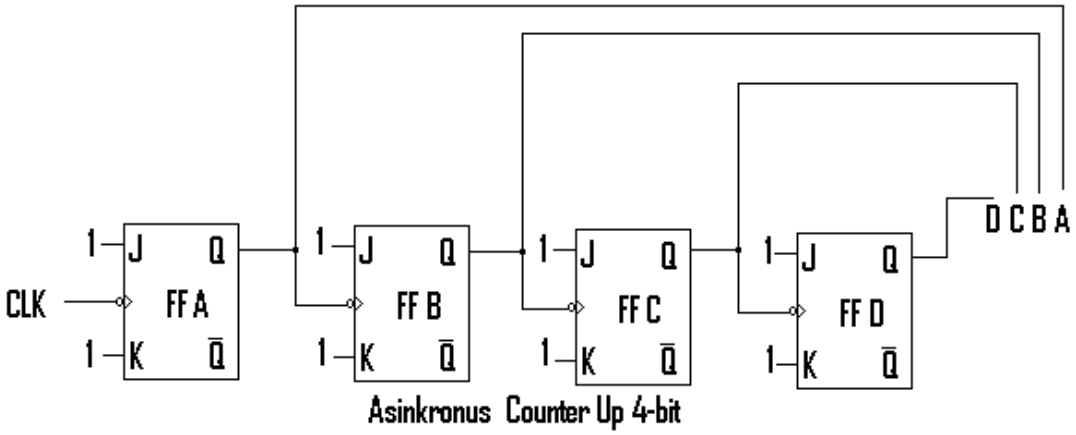


Rangkaian Dasar

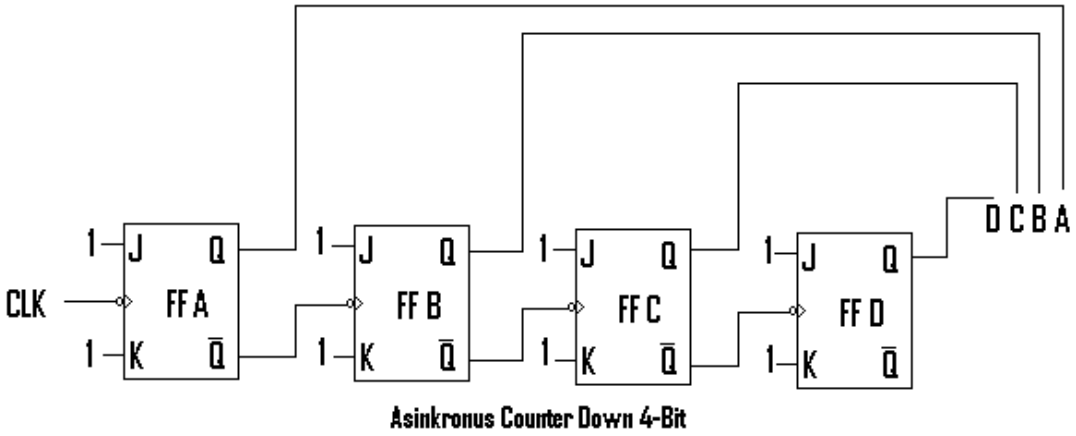
Mode operasi	Input			Output	
	CLK	J	K	Q	Q̄
Tetap	↓	0	0	Tetap	
Reset	↓	0	1	0	1
Set	↓	1	0	1	0
Toggle	↓	1	1	Toggle	

Tabel kebenaran

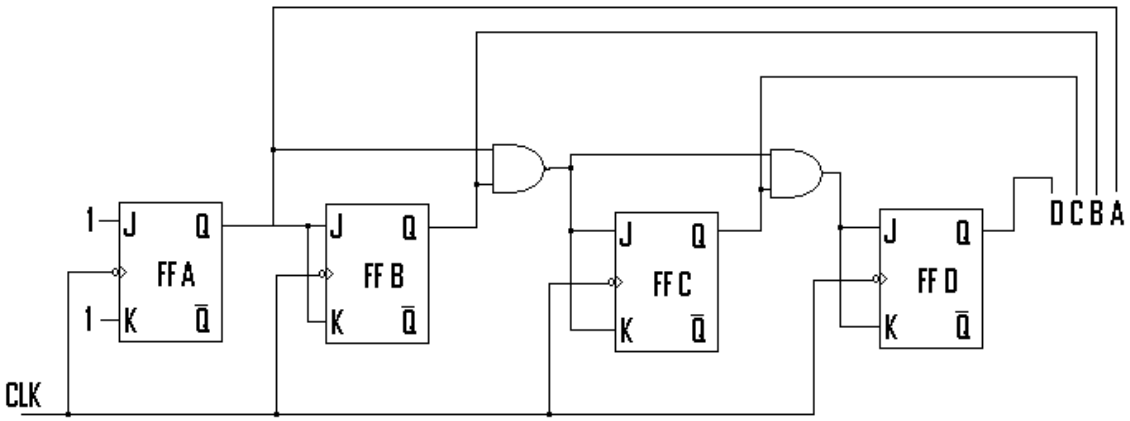
Counter (rangkaian logika sekuensial yang di bentuk dari flip-flop) mempunyai karakteristik untuk melakukan cacahan / counter / hitungan berurutan (sekuen) ke atas (dari nilai terkecil hingga terbesar) atau hitungan ke bawah (dari nilai terbesar sampai nilai terkecil), terbagi menjadi counter Asinkron (yg beroperasi tidak serentak dengan pulsa clock) serta Counter Sinkron (yang beroperasi serentak dengan pulsa clock)



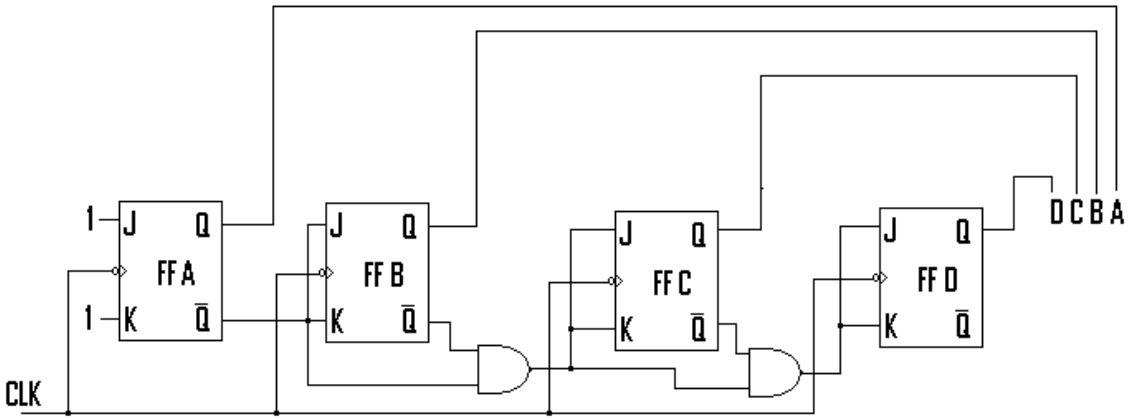
Asinkronus Counter Up 4-bit



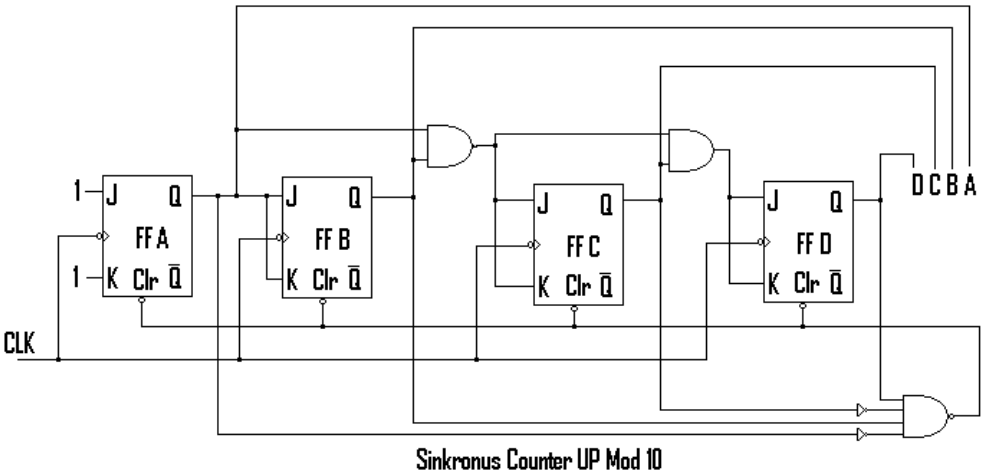
Asinkronus Counter Down 4-Bit



Sinkronus Counter Up 4-Bit

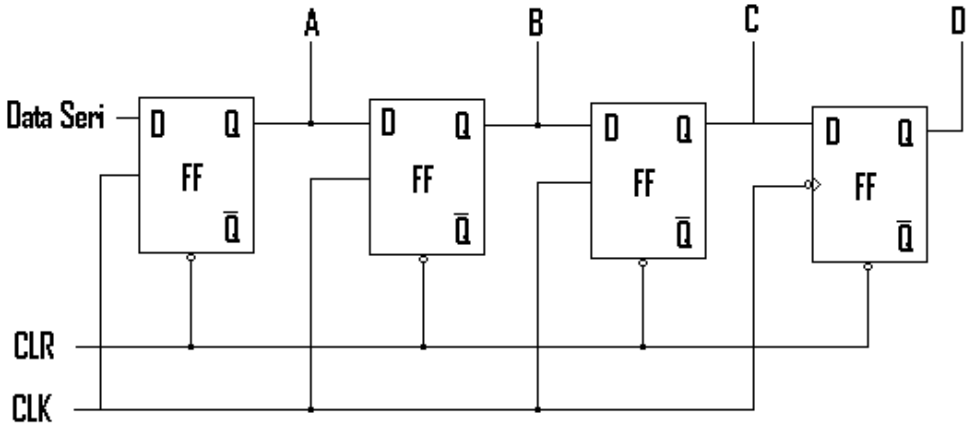


Sinkronus Counter Down 4-Bit

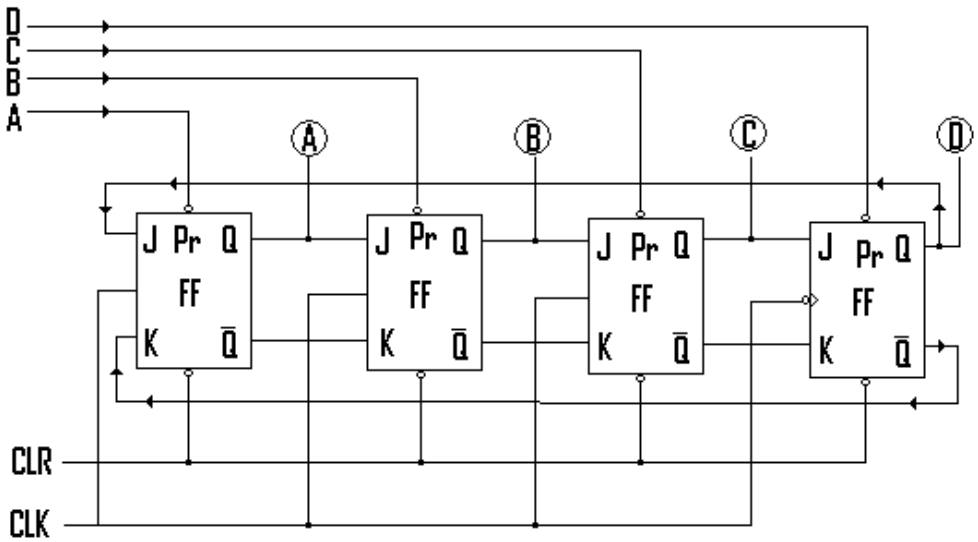


Sinkronus Counter UP Mod 10

REGISTER. (rangkaian logika sekuensial yang berfungsi sebagai penyimpan bit / memori). Data-data biner dapat dimasukkan secara seri maupun parallel dan dapat dikeluarkan secara seri maupun parallel juga .



Register geser Seri



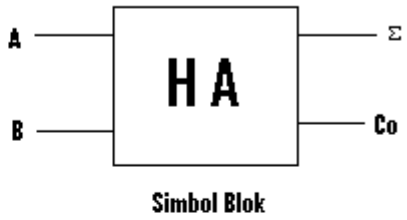
Register Geser Paralel

Aritmatika Biner

Penjumlahan Biner,

Input		Output	
A	B	Jumlah (Σ)	Carry out (Co)
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1
		Xor	And

Tabel kebenaran



Simbol Blok

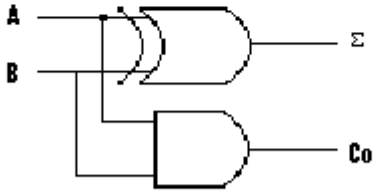


Diagram Logika

HALF ADDER (Penambah Setengah)

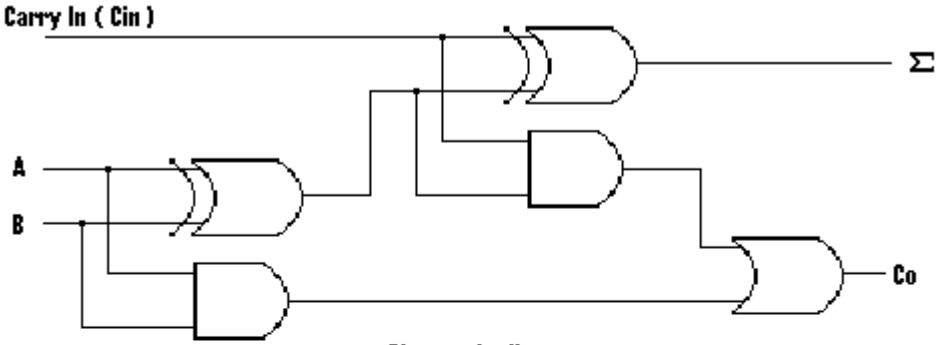
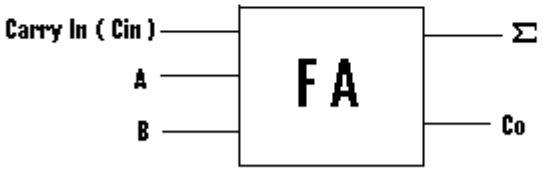
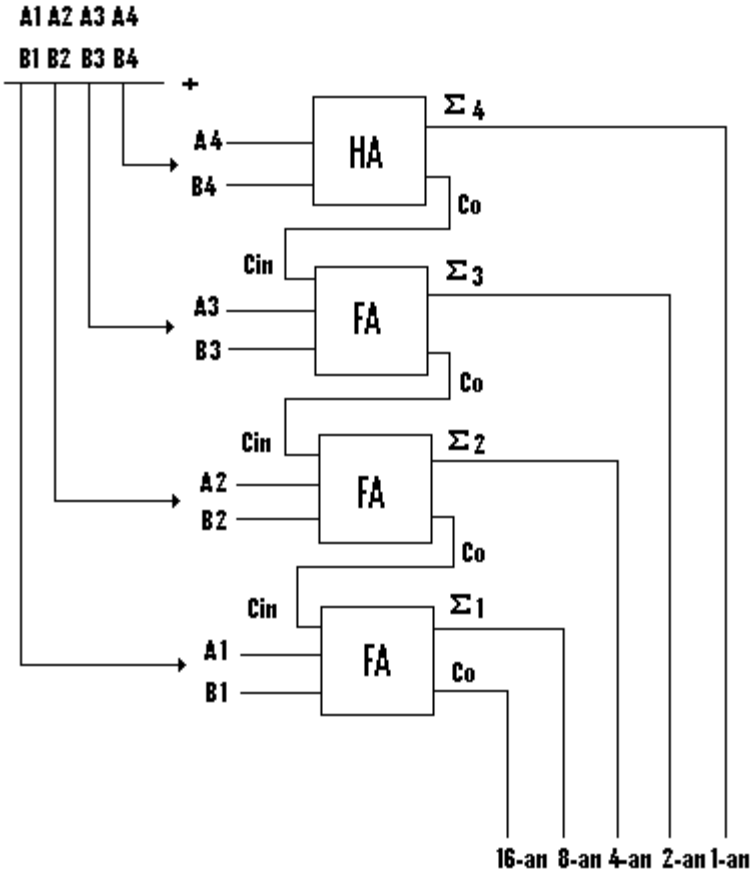


Diagram Logika

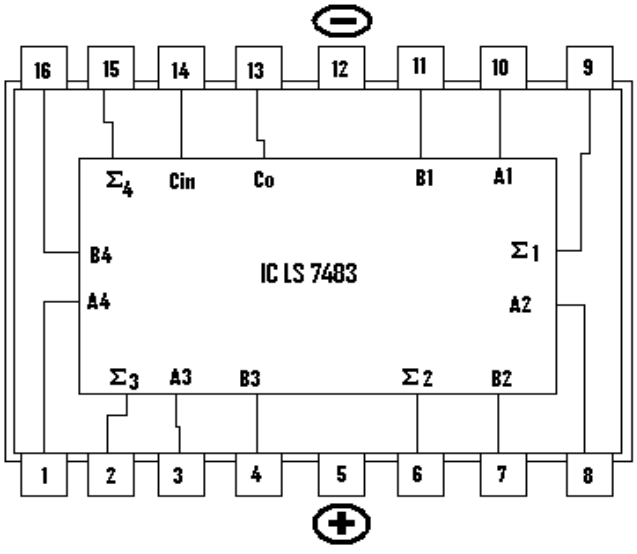


Simbol Blok

FULL ADDER (Penambah Penuh)



Penambah 4-Bit



Pengurangan Biner

Input		Output	
A	B	Selisih (Di)	Borrow Out (Bo)
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0
		Xor	$\bar{A}$ and B

Tabel kebenaran

Simbol Blok

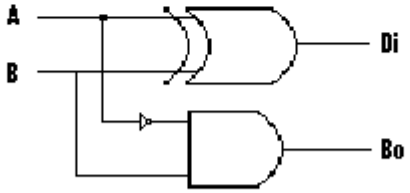


Diagram Logika

HALF SUBSTRATCTOR (Pengurang Setengah)

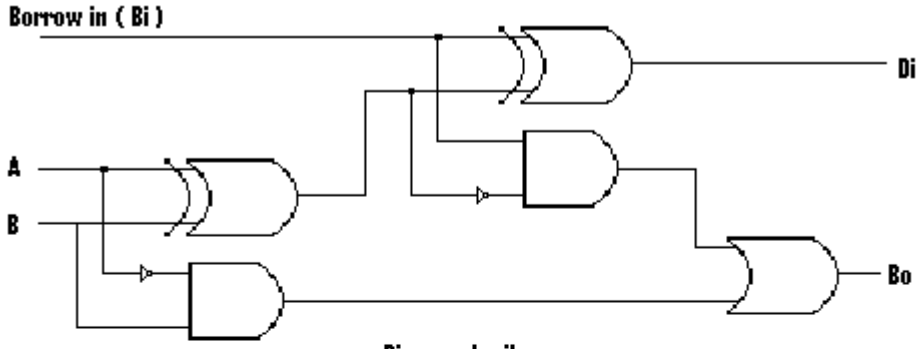
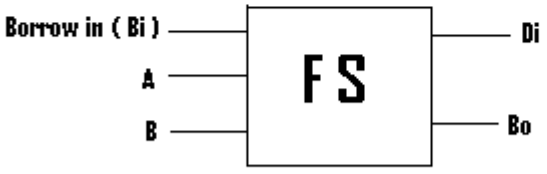
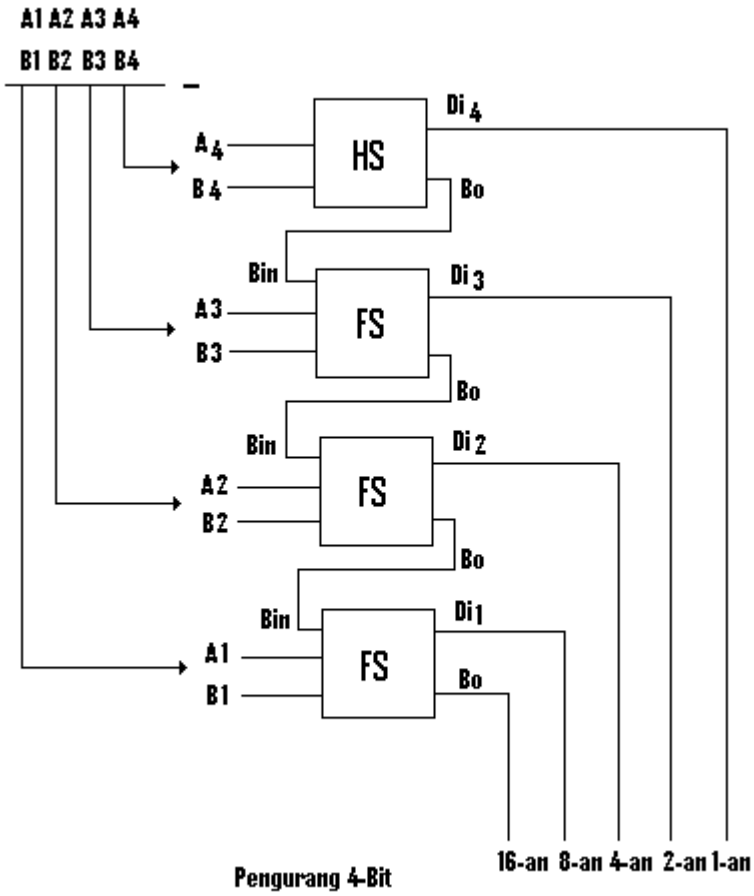


Diagram Logika

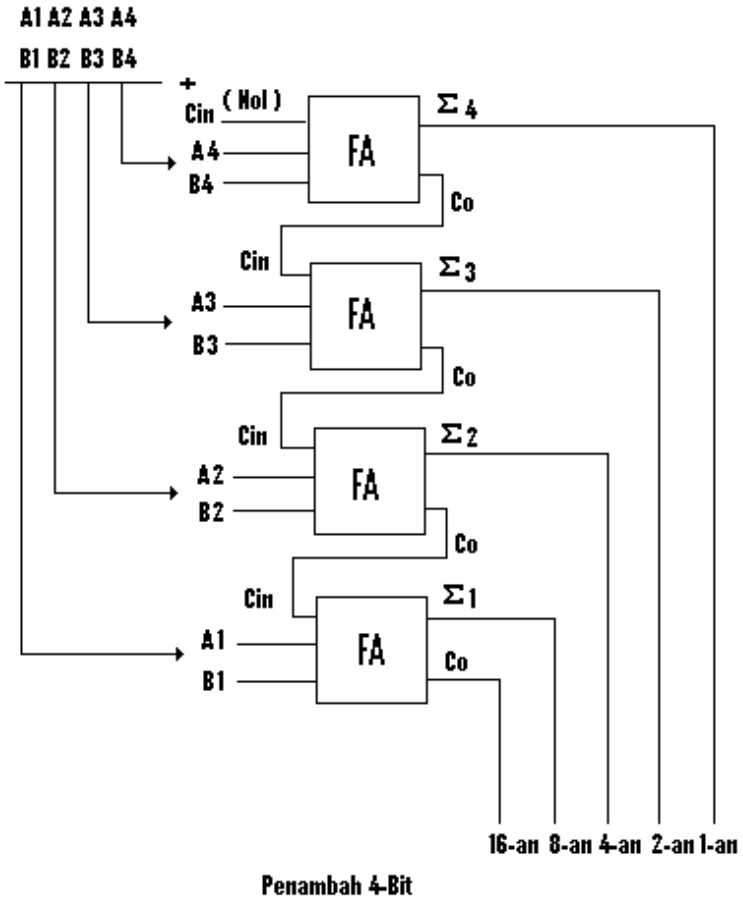


Simbol Blok

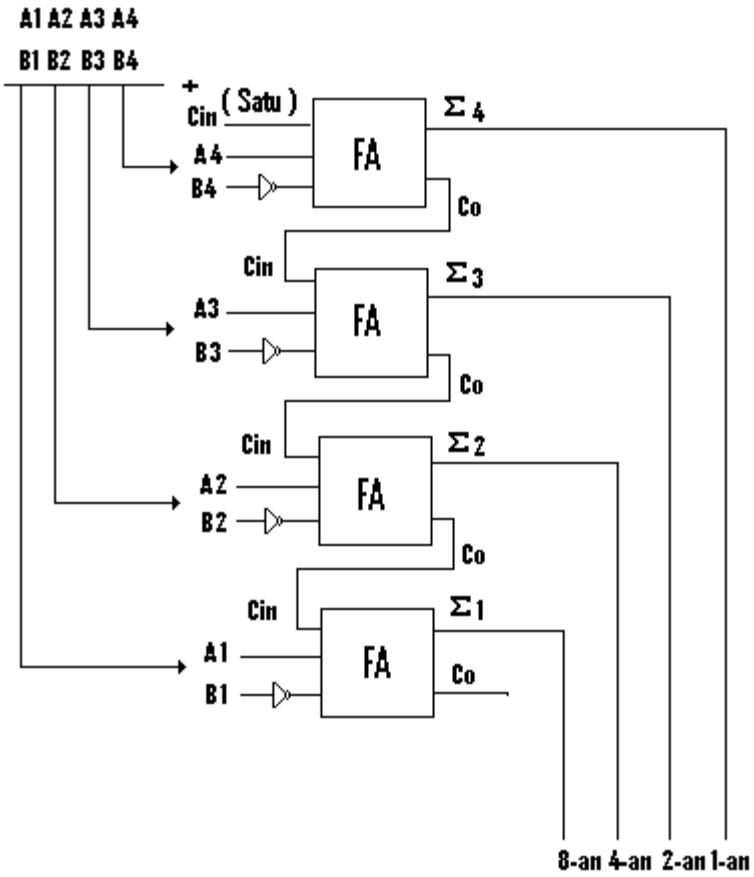
FULL SUBSTRACTOR (Pengurang penuh)


A1 A2 A3 A4
B1 B2 B3 B4
A4
B4
HS
Di 4
Bo
Bin
A3
B3
FS
Di 3
Bo
Bin
A2
B2
FS
Di 2
Bo
Bin
A1
B1
FS
Di 1
Bo
16-an
8-an
4-an
2-an
1-an
Pengurang 4-Bit

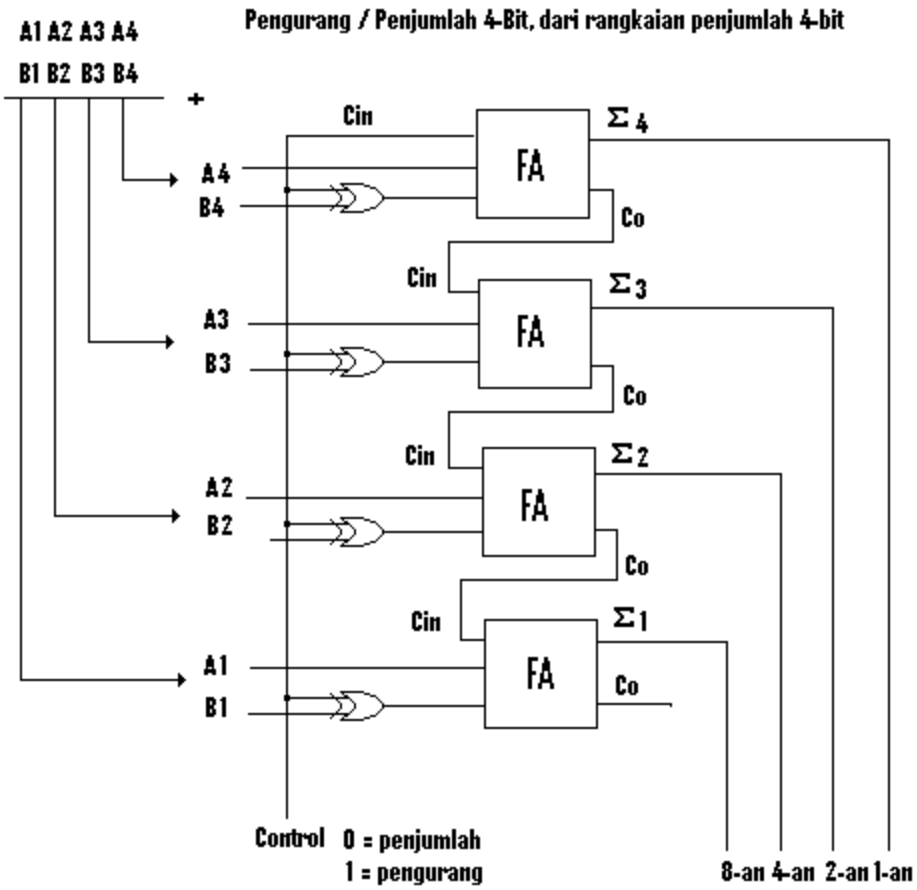
Modifikasi Rangkaian Penambah 4-Bit



Penambah 4-Bit



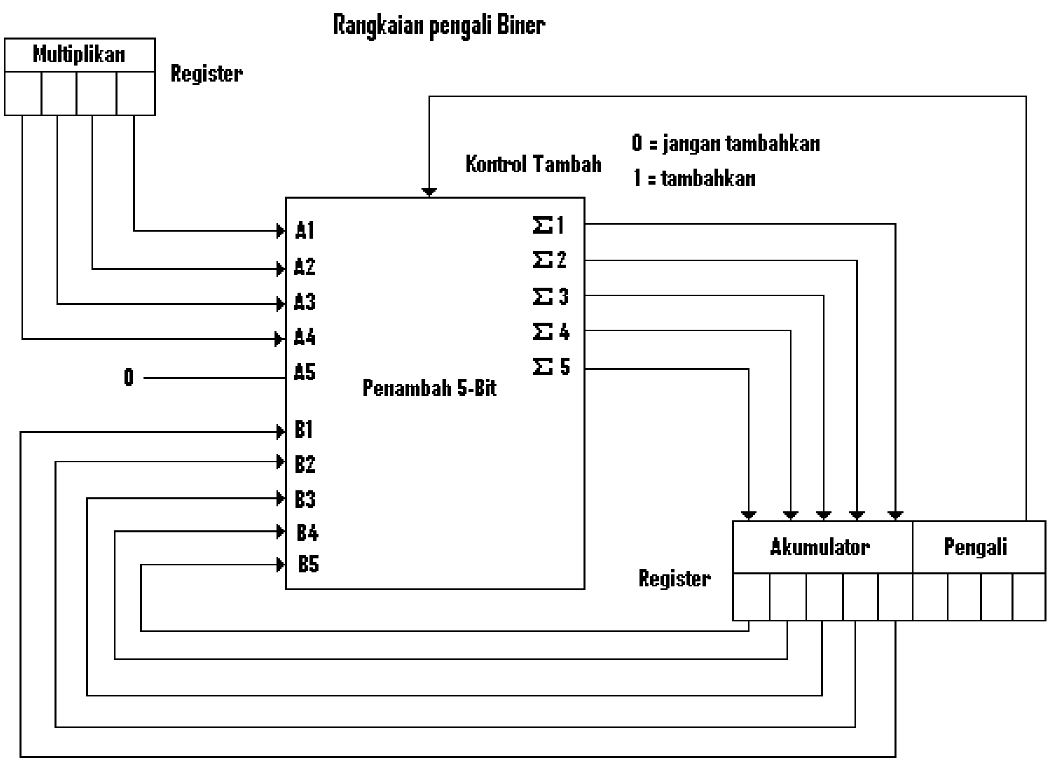
Pengurang 4-bit, dari rangkaian penambah 4-bit


Pengurang / Penjumlah 4-Bit, dari rangkaian penjumlah 4-bit
A1 A2 A3 A4
B1 B2 B3 B4
+
Cin
FA
Σ4
Co
A4
B4
Cin
FA
Σ3
Co
A3
B3
Cin
FA
Σ2
Co
A2
B2
Cin
FA
Σ1
Co
A1
B1
Control
0 = penjumlah
1 = pengurang
8-an
4-an
2-an
1-an

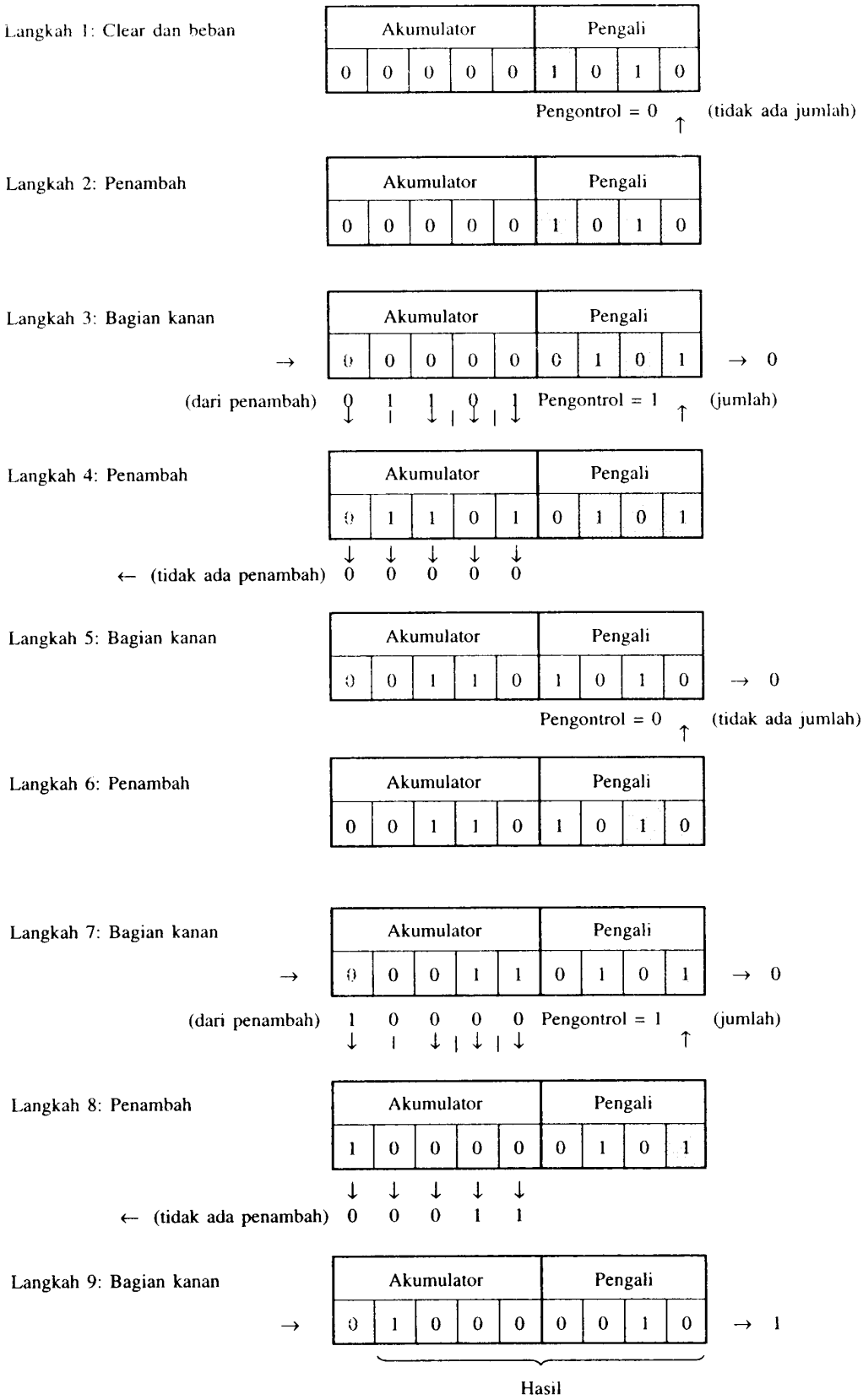
Perkalian biner

```
  13              1101      ( 13 )
  10              1010  X   ( 10 )
 ----  X         ------
  00              0000
 13              1101
 ----           0000
 130           1101
               ---------
               10000010     ( 130 )
               ↓      ↓
              128     2
```

Metode yang digunakan dalam implementasi perkalian biner ke rangkaian digital adalah : **Add and shift** , metode tambah dan geser.



Contoh : 13 (1101) X 10 (1010)



(*b*) Operasi pengali biner

Referensi Buku :

1.Prinsip – Prinsip Digital, Roger L. Tokheim, Sutisna, penerbit Erlangga, Jakarta

2.Rangkaian Digital dan Rancangan Logika, Samuel C. Lee, Sutisna, Penerbit Erlanga, Jakarta