

# Sistem Bilangan

Ada beberapa sistem bilangan yang digunakan dalam sistem digital. Yang paling umum adalah sistem bilangan desimal, biner, oktal, dan heksadesimal. Sistem bilangan desimal merupakan sistem bilangan yang paling familier dengan kita karena berbagai kemudahannya yang kita pergunakan sehari-hari. Sistem bilangan biner merupakan sistem bilangan yang paling banyak digunakan dalam sistem digital karena sistem bilangan ini secara langsung dapat mewakili logika yang ada. Sementara itu sistem bilangan oktal dan heksadesimal biasanya banyak digunakan dalam sistem digital untuk memperpendek penyajian suatu bilangan yang tadinya disajikan dalam sistem bilangan biner.

Secara umum bilangan dapat dibagi menjadi beberapa kategori. Dari segi koma desimal (*point*), bilangan dapat dibagi menjadi bilangan bulat (*integer number/fixed-point number*) dan bilangan pecahan (*floating-point number*). Dan dari segi tanda, bilangan dapat dibagi menjadi bilangan tak bertanda (*unsigned number*) dan bilangan bertanda (*signed number*). Pada bab ini akan dijelaskan bilangan bulat tak bertanda (*unsigned integer*), bilangan bulat bertanda (*signed integer*) dan bilangan pecahan tak bertanda (*floating-point number*). Dengan mempelajari beberapa karakteristik suatu sistem bilangan tersebut akan membantu kita untuk lebih memahami sistem bilangan yang lain.

## 2.1 Desimal

Sistem bilangan desimal disusun dari 10 angka atau lambang. Dengan menggunakan lambang-lambang tersebut sebagai digit pada sebuah bilangan, kita dapat mengekspresikan suatu kuantitas. Kesepuluh lambang tersebut adalah:

$$D = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$$

Sistem bilangan desimal disebut juga sistem bilangan basis 10 atau radix 10 karena mempunyai 10 digit. Sistem bilangan ini bersifat alamiah karena pada kenyataannya manusia mempunyai 10 jari. Kata digit itu sendiri diturunkan dari kata bahasa Latin *finger*.

Ciri suatu bilangan menggunakan sistem bilangan desimal adalah adanya tambahan subskrip *des* atau *10* atau tambahan *D* di akhir suatu bilangan. Contoh:  $357_{des} = 357_{10} = 357D$ . Namun karena bilangan desimal sudah menjadi bilangan yang digunakan sehari-hari, subskrip tersebut biasanya dihilangkan.

Sistem bilangan desimal merupakan sebuah sistem nilai-posisi. Pada sistem ini, nilai sebuah digit tergantung pada posisinya. Representasi bilangan desimal bulat  $m$  digit adalah sebagai berikut,

$$(d_{m-1} \dots d_i \dots d_2 d_1 d_0) \text{ dengan } d_i \in D$$

Sehingga suatu bilangan desimal  $m$  digit akan mempunyai nilai:

$$Z = \sum_{i=0}^{m-1} d_i \cdot 10^i$$

Contoh:

- Bilangan 357.

Pada bilangan tersebut, digit 3 berarti 3 *ratusan*, 5 berarti 5 *puluhan*, dan 7 berarti 7 *satuan*. Sehingga, 3 mempunyai arti paling besar di antara tiga digit yang ada. Digit ini bertindak sebagai digit paling berarti (*Most Significant Digit, MSD*). Sedangkan 7 mempunyai arti paling kecil di antara tiga digit yang ada dan disebut digit paling tidak berarti (*Least Significant Digit, LSD*).

Untuk bilangan desimal pecahan, representasi nilainya menjadi sebagai berikut,

$$(d_{m-1} \dots d_i \dots d_2 d_1, d_0 d_{-1} \dots d_n) \text{ dengan } d_i \in D$$

Sehingga suatu bilangan desimal pecahan akan mempunyai nilai:

$$Z = \sum_{i=n}^{m-1} d_i \cdot 10^i$$

Koma desimal digunakan untuk memisahkan bagian bulat dan pecahan bilangan. Posisi relatif terhadap koma desimal memberikan arti yang dapat dinyatakan sebagai pangkat dari 10.

Contoh:

- Bilangan 35,27.

Bilangan ini mempunyai arti 3 puluhan ditambah 5 satuan ditambah 2 per sepuluh ditambah 7 per seratusan. Koma desimal memisahkan pangkat positif dari 10 dengan pangkat negatifnya.

$$35,27 = 3 \times 10^{+1} + 5 \times 10^0 + 2 \times 10^{-1} + 7 \times 10^{-2}$$

Secara umum dapat dikatakan, nilai suatu bilangan desimal merupakan penjumlahan dari perkalian setiap digit dengan nilai posisinya.

2.2 Biner

Sistem digital hanya mengenal dua logika, yaitu 0 dan 1. Logika 0 biasanya mewakili kondisi mati dan logika 1 mewakili kondisi hidup. Pada sistem bilangan biner, hanya dikenal dua lambang, yaitu 0 dan 1. Karena itu, sistem bilangan biner paling sering digunakan untuk merepresentasikan kuantitas dan mewakili keadaan dalam sistem digital maupun sistem komputer.

Digit bilangan biner disebut *binary digit* atau *bit*. Empat bit dinamakan *nibble* dan delapan bit dinamakan *byte*. Sejumlah bit yang dapat diproses komputer untuk mewakili suatu karakter (dapat berupa huruf, angka atau lambang khusus) dinamakan *word*. Sebuah komputer dapat memproses data satu *word* yang terdiri dari 4 sampai 64 bit. Sebagai contoh, sebuah komputer yang menggunakan mikroprosesor 32 bit dapat menerima, memproses, menyimpan dan mengirim data atau instruksi dalam format 32 bit.

Jika komputer digunakan untuk memproses karakter, maka karakter (yang meliputi huruf, angka, tanda baca dan karakter kontrol) tersebut harus diformat dalam bentuk kode alfanumerik. Format baku ASCII (*American Standard Code for Information Interchange*) menggunakan format data tujuh bit untuk mewakili semua karakter yang ada termasuk tanda baca dan penanda kontrol. Dengan format tujuh bit, maka ASCII dapat menampung  $2^7 = 128$  data.

Sistem bilangan biner merupakan sistem bilangan basis dua. Pada sistem bilangan ini hanya dikenal dua lambang, yaitu:

$$B = \{ 0, 1 \}$$

Ciri suatu bilangan menggunakan sistem bilangan biner adalah adanya tambahan subskrip *bin* atau 2 atau tambahan huruf *B* di akhir suatu bilangan. Contoh:  $1010011_{bin} = 1010011_2 = 1010011B$ .

Representasi bilangan biner bulat m bit adalah sebagai berikut,

$$(b_{m-1} \dots b_1 \dots b_2 \ b_1 \ b_0) \text{ dengan } b_i \in B$$

Sehingga suatu bilangan biner m bit akan mempunyai nilai:

$$Z = \sum_{i=0}^{m-1} b_i \cdot 2^i$$

Bit paling kiri dari suatu bilangan biner bertindak sebagai bit paling berarti (*Most Significant Bit*, MSB), sedangkan bit paling kanan bertindak sebagai bit paling tidak berarti (*Least Significant Bit*, LSB).

Untuk bilangan biner pecahan, representasi nilainya menjadi sebagai berikut,

$$(b_{m-1} \dots b_1 \dots b_2 \ b_1 \ , \ b_0 \ b_{-1} \dots \ b_n) \text{ dengan } b_i \in B$$

Sehingga suatu bilangan biner pecahan akan mempunyai nilai:

$$Z = \sum_{i=n}^{m-1} b_i \cdot 2^i$$

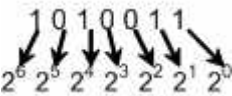
Persamaan tersebut dapat digunakan untuk mengonversi suatu bilangan biner ke bilangan desimal.

a) Konversi Bilangan Biner ke Desimal

Konversi bilangan biner ke desimal dilakukan dengan menjumlahkan hasil perkalian semua bit biner dengan beratnya.

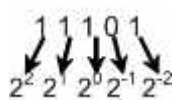
Contoh:

•  $1010011_{bin} = 83_{des}$



$$\begin{aligned} 1010011_{bin} &= 1 \times 2^6 + 0 \times 2^5 + 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 \\ &= 64 + 0 + 16 + 0 + 0 + 2 + 1 \\ &= 83_{des} \end{aligned}$$

•  $111,01_{bin} = 7,25_{des}$



$111,01_{bin} = 1 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$   
 $= 4 + 2 + 1 + 0 + 0,25$   
 $= 7,25_{des}$

b) Konversi Bilangan Desimal ke Biner

Konversi Bilangan Desimal Bulat ke Biner

Konversi bilangan desimal bulat ke biner dilakukan dengan membagi secara berulang-ulang suatu bilangan desimal dengan 2. Sisa setiap pembagian merupakan bit yang didapat.

Contoh:

- $625_{des} = \dots_{bin}$   

$625 / 2 = 312$	sisa	1
$312 / 2 = 156$		0
$156 / 2 = 78$		0
$78 / 2 = 39$		0
$39 / 2 = 19$		1
$19 / 2 = 9$		1
$9 / 2 = 4$		1
$4 / 2 = 2$		0
$2 / 2 = 1$		0
$1 / 2 = 0$		1

  
 $625_{des} = 1001110001_{bin}$

Konversi Bilangan Desimal Pecahan ke Biner

Bilangan desimal real dapat dapat pula dikonversi ke bilangan real biner. Konversi dilakukan dengan cara memisahkan antara bagian bulat dan bagian pecahannya. Konversi bagian bulat dapat dilakukan seperti cara di atas. Sedangkan konversi bagian pecahan dilakukan dengan mengalikan pecahan tersebut dengan 2. Kemudian bagian pecahan dari hasil perkalian ini dikalikan dengan 2. Langkah ini diulang hingga didapat hasil akhir 0. Bagian bulat dari setiap hasil perkalian merupakan bit yang didapat.

Contoh:

- $625,1875_{des} = \dots_{bin}$   
 $625_{des} = 1001110001_{bin}$   
 $0,1875_{des} = \dots_{bin}$   

$0,1875 \times 2 = 0,375$	0
$0,375 \times 2 = 0,75$	0
$0,75 \times 2 = 1,5$	1
$0,5 \times 2 = 1$	1

  
 $0,1875_{des} = 0011_{bin}$   
 $625,1875_{des} = 1001110001,0011_{bin}$
- $0,1_{des} = \dots_{bin}$   

$0,1 \times 2 = 0,2$	0
$0,2 \times 2 = 0,4$	0
$0,4 \times 2 = 0,8$	0
$0,8 \times 2 = 1,6$	1
$0,6 \times 2 = 1,2$	1
$0,2 \times 2 = 0,4$	0
$0,4 \times 2 = 0,8$	0
$0,8 \times 2 = 1,6$	1
$0,6 \times 2 = 1,2$	1
$0,6 \times 2 = \dots$	...

  
 $0,1_{des} = 0,000110011\dots_{bin}$

2.3 Oktal

Sistem bilangan oktal merupakan sistem bilangan basis delapan. Pada sistem bilangan ini terdapat delapan lambang, yaitu:

$$O = \{ 0, 1, 2, 3, 4, 5, 6, 7 \}$$

Ciri suatu bilangan menggunakan sistem bilangan oktal adalah adanya tambahan subskrip *okt* atau *8* atau tambahan huruf *O* di akhir suatu bilangan. Contoh:  $1161_{\text{okt}} = 1161_8 = 1161O$ .

Representasi suatu bilangan oktal bulat *m* digit adalah sebagai berikut,

$$(o_{m-1} \dots o_i \dots o_2 o_1 o_0) \text{ dengan } o_i \in O$$

Sehingga suatu bilangan oktal bulat *m* digit akan mempunyai nilai:

$$Z = \sum_{i=0}^{m-1} o_i \cdot 8^i$$

Untuk bilangan oktal pecahan, representasi nilainya menjadi sebagai berikut,

$$(o_{m-1} \dots o_i \dots o_2 o_1 o_0 o_{-1} \dots o_n) \text{ dengan } o_i \in O$$

Sehingga suatu bilangan oktal pecahan akan mempunyai nilai:

$$Z = \sum_{i=n}^{m-1} o_i \cdot 8^i$$

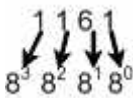
Persamaan tersebut dapat digunakan untuk mengonversi suatu bilangan oktal ke bilangan desimal.

a) Konversi Bilangan Oktal ke Desimal

Konversi bilangan oktal ke desimal dilakukan dengan menjumlahkan hasil perkalian semua digit oktal dengan beratnya.

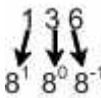
Contoh:

•  $1161_{\text{okt}} = 625_{\text{des}}$



$$\begin{aligned} 1160_{\text{okt}} &= 1 \times 8^3 + 1 \times 8^2 + 6 \times 8^1 + 1 \times 8^0 \\ &= 512 + 64 + 48 + 1 \\ &= 83_{\text{des}} \end{aligned}$$

•  $13,6_{\text{okt}} = 11,75_{\text{des}}$



$$\begin{aligned} 13,6_{\text{okt}} &= 1 \times 8^1 + 3 \times 8^0 + 6 \times 8^{-1} \\ &= 8 + 3 + 0,75 \\ &= 11,75_{\text{des}} \end{aligned}$$

b) Konversi Bilangan Desimal ke Oktal

Konversi bilangan bulat desimal ke oktal dilakukan dengan membagi secara berulang-ulang suatu bilangan desimal dengan 8. Sisa setiap pembagian merupakan digit oktal yang didapat. Contoh:

•  $625_{\text{des}} = \dots_{\text{okt}}$

$$\begin{array}{rcl} 625 / 8 & = & 78 \text{ sisa } 1 \\ 78 / 8 & = & 9 \quad 6 \\ 9 / 8 & = & 1 \quad 1 \\ 1 / 8 & = & 0 \quad 1 \end{array}$$

$625_{\text{des}} = 1161_{\text{bin}}$

Konversi bilangan desimal pecahan ke oktal dilakukan dengan cara hampir sama dengan konversi bilangan desimal pecahan ke biner, yaitu dengan mengalikan suatu bilangan desimal pecahan dengan 8. Bagian pecahan dari hasil perkalian ini dikalikan dengan 8. Langkah ini diulang hingga didapat hasil akhir 0. Bagian bulat dari setiap hasil perkalian merupakan digit yang didapat.

•  $0,75_{\text{des}} = 0,6_{\text{okt}}$

$$0,75 \times 8 = 6,00$$

•  $0,1_{\text{des}} = 0,064 \dots \dots \text{okt}$

$0,1 \times 8 = 0,8$   
 $0,8 \times 8 = 6,4$   
 $0,4 \times 8 = 4,8$

c) Konversi Bilangan Oktal ke Biner

Konversi bilangan oktal ke biner lebih mudah dibandingkan dengan konversi bilangan oktal ke desimal. Satu digit oktal dikonversi ke 3 bit biner. Tabel 2.1 dapat digunakan untuk membantu proses pengonversian ini.

Contoh:

- $1161_{\text{okt}} = 1001110001_{\text{bin}}$   

1

1

6

1

001

001

110

001
- $0,064_{\text{okt}} = 0,000110011_{\text{bin}}$   

0

6

4

000

110

011

d) Konversi Bilangan Biner ke Oktal

Konversi bilangan biner ke oktal lebih mudah dibandingkan konversi bilangan desimal ke oktal. Untuk bagian bulat, kelompokkan setiap tiga bit biner dari paling kanan, kemudian konversikan setiap kelompok ke satu digit oktal. Dan untuk bagian pecahan, kelompokkan setiap tiga bit biner dari paling kiri, kemudian konversikan setiap kelompok ke satu digit oktal. Proses ini merupakan kebalikan dari proses konversi bilangan oktal ke biner.

Contoh:

- $1001110001_{bin} = 1161_{okt}$   
 $001\ 001\ 110\ 001$   
    ↓   ↓   ↓   ↓  
    1   1   6   1
- $0,000110011_{bin} = 0,064_{okt}$   
 $000\ 110\ 011$   
    ↓   ↓   ↓  
    0   6   4

2.4 Heksadesimal

Sistem bilangan heksadesimal merupakan sistem bilangan basis enam belas. Meskipun pada sistem digital dan komputer operasi secara fisik dikerjakan secara biner, namun untuk representasi data banyak digunakan format bilangan heksadesimal karena format ini lebih praktis, mudah dibaca dan mempunyai kemungkinan timbul kesalahan lebih kecil. Penerapan format heksadesimal banyak digunakan pada penyajian lokasi memori, penyajian isi memori, kode instruksi dan kode yang merepresentasikan alfanumerik dan karakter nonnumerik.

Pada sistem bilangan ini terdapat enam belas lambang, yaitu:

$$H = \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F \}$$

Ciri suatu bilangan menggunakan sistem bilangan heksadesimal adalah adanya tambahan subskrip *heks* atau *16* atau tambahan huruf *H* di akhir suatu bilangan. Contoh:  $271_{heks} = 271_{16} = 271H$ .

Representasi suatu bilangan heksadesimal bulat adalah sebagai berikut,

$$(h_{m-1} \dots h_i \dots h_2 h_1 h_0) \text{ dengan } h_i \in H$$

Sehingga suatu bilangan heksadesimal m digit akan mempunyai nilai:

$$Z = \sum_{i=0}^{m-1} h_i \cdot 16^i$$

Untuk bilangan heksadesimal pecahan, representasi nilainya menjadi sebagai berikut,

$$(h_{m-1} \dots h_i \dots h_2 h_1, h_0 h_{-1} \dots h_n) \text{ dengan } h_i \in H$$

Sehingga suatu bilangan heksadesimal pecahan akan mempunyai nilai:

$$Z = \sum_{i=n}^{m-1} h_i \cdot 16^i$$

Persamaan tersebut dapat digunakan untuk mengonversi suatu bilangan oktal ke bilangan desimal.

a) Konversi Bilangan Heksadesimal ke Desimal

Konversi bilangan heksadesimal ke desimal dilakukan dengan menjumlahkan hasil perkalian semua digit heksadesimal dengan beratnya.

Contoh:

- $271_{heks} = 625_{des}$   
 $\begin{matrix} 2 & 7 & 1 \\ \swarrow & \downarrow & \searrow \\ 16^2 & 16^1 & 16^0 \end{matrix}$   
 $271_{heks} = 2 \times 16^2 + 7 \times 16^1 + 1 \times 16^0$   
 $= 512 + 112 + 1$   
 $= 625_{des}$
- $0,Chexs = 0,75$   
 $\begin{matrix} C & \\ \downarrow & \searrow \\ 16^0 & 16^{-1} \end{matrix}$   
 $0,Chexs = 0 \times 16^0 + 12 \times 16^{-1}$   
 $= 0 + 0,75$   
 $= 0,75_{des}$

b) Konversi Bilangan Desimal ke Heksadesimal

Konversi bilangan desimal bulat ke heksadesimal dilakukan dengan membagi secara berulang-ulang suatu bilangan desimal dengan 16. Sisa setiap pembagian merupakan digit heksadesimal yang didapat. Contoh:

- $625_{des} = 271_{heks}$

625 / 16 = 39 sisa 1

39 / 16 = 2      7

2 / 16 = 0      2

625<sub>des</sub> = 271<sub>heks</sub>

Konversi bilangan desimal pecahan ke heksadesimal dilakukan dengan cara hampir sama dengan konversi bilangan desimal pecahan ke biner, yaitu dengan mengalikan suatu bilangan desimal pecahan dengan 16. Bagian pecahan dari hasil perkalian ini dikalikan dengan 16. Langkah ini diulang hingga didapat hasil akhir 0. Bagian bulat dari setiap hasil perkalian merupakan digit yang didapat.

Contoh:

- 625,1875<sub>des</sub> = ... bin
- 0,75<sub>des</sub> = 0,75<sub>heks</sub>  
0,75 X 16 = 12
- 0,1<sub>des</sub> = 0,1<sub>heks</sub> ..... heks  
0,10 X 16 = 1,6  
0,60 X 16 = 9,6

Konversi Bilangan Heksadesimal ke Biner

Konversi bilangan heksadesimal ke biner lebih mudah dibandingkan konversi bilangan heksadesimal ke desimal. Satu digit heksadesimal dikonversi ke 4 bit. Tabel 2.1 dapat digunakan untuk membantu proses pengonversian ini.

Contoh:

271<sub>heks</sub> = 1001110001<sub>bin</sub>

2   7   1

↓   ↓   ↓

0010 0111 0001

0,19<sub>heks</sub> = 0,00011001<sub>bin</sub>

0   1   9

↓   ↓   ↓

0000 0001 1001

c) Konversi Bilangan Biner ke Heksadesimal

Konversi bilangan biner ke heksadesimal lebih mudah dibandingkan konversi bilangan desimal ke heksadesimal. Untuk bagian bulat, kelompokkan setiap empat bit biner dari paling kanan, kemudian konversikan setiap kelompok ke satu digit heksadesimal. Dan untuk bagian pecahan, kelompokkan setiap empat bit biner dari paling kiri, kemudian konversikan setiap kelompok ke satu digit heksadesimal. Proses ini merupakan kebalikan dari proses konversi bilangan heksadesimal ke biner.

Contoh:

1001110001<sub>bin</sub> = 271<sub>heks</sub>

10 0111 0001

↓   ↓   ↓

2   7   1

0,00011001<sub>bin</sub> = 0,19<sub>heks</sub>

0000 0001 1001

↓   ↓   ↓

0   1   9

2.5 BCD (Binary Coded Decimal)

Sistem bilangan BCD hampir sama dengan sistem bilangan biner. Pada sistem bilangan ini, setiap satu digit desimal diwakili secara tersendiri ke dalam bit-bit biner. Karena pada sistem bilangan desimal terdapat 10 digit, maka dibutuhkan 4 bit biner untuk mewakili setiap digit desimal. Setiap digit desimal dikodekan ke sistem bilangan biner tak bertanda. Sistem bilangan BCD biasanya digunakan untuk keperluan penampil tujuh segmen (*seven-segment*).

Contoh:

625<sub>des</sub> = 0110 0010 0101<sub>BCD</sub>

6   2   5

↓   ↓   ↓

0110 0010 0101

Konversi bilangan desimal dari 0 sampai 15 ke bilangan biner, oktal, heksadesimal dan BCD dapat dilihat pada tabel 2.1.

Table 2.1 Konversi antar sistem bilangan

Desimal	Biner	Oktal	Heksadesimal	BCD
0	0000	0	0	0000
1	0001	1	1	0001
2	0010	2	2	0010
3	0011	3	3	0011
4	0100	4	4	0100
5	0101	5	5	0101
6	0110	6	6	0110

11

7	0111	7	7	0111
8	1000	10	8	1000
9	1001	11	9	1001
10	1010	12	A	0001 0000
11	1011	13	B	0001 0001
12	1100	14	C	0001 0010
13	1101	15	D	0001 0011
14	1110	16	E	0001 0100
15	1111	17	F	0001 0101

Perlu diingat di sini, pada sistem bilangan BCD, pengkodean bilangan desimal menjadi bilangan biner format 4 bit, sehingga terdapat 6 nilai biner yang bukan merupakan format sistem bilangan BCD karena tidak mewakili nilai desimal. Keempat bilangan biner tersebut adalah 1010, 1011, 1100, 1101, 1110 dan 1111. Dalam praktek keempat bilangan biner tersebut masuk dalam kondisi yang diabaikan (*don't care*).

2.6 Sistem Bilangan Biner Tak Bertanda dan Bertanda

Terdapat dua sistem bilangan biner, yaitu bilangan biner tak bertanda dan bilangan biner bertanda. Pada sistem bilangan biner tak bertanda, hanya dikenal bilangan biner positif dan tidak diijinkan adanya bilangan biner negatif. Di sini semua bit digunakan untuk merepresentasikan suatu nilai.

Contoh:

- Bilangan biner 4 bit 1100.

A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>
1	1	0	0

Pada bilangan biner tak bertanda di atas, nilai bilangan dihitung dari A<sub>3</sub> ... A<sub>0</sub>. Sehingga,

1100<sub>bin</sub> = 1 X 2<sup>3</sup> + 1 X 2<sup>2</sup> + 0 X 2<sup>1</sup> + 0 X 2<sup>0</sup>  
= 12<sub>des</sub>

Pada bilangan biner bertanda, bit paling kiri menyatakan tanda, sehingga nilai bilangan dihitung dari A<sub>2</sub> ... A<sub>0</sub>.

0100<sub>bin</sub> = + (1 X 2<sup>2</sup> + 0 X 2<sup>1</sup> + 0 X 2<sup>0</sup>)  
= 4<sub>des</sub>

Pada sistem ini, bit paling kiri menyatakan tanda negatif atau positif nilai yang diwakilinya. Tanda positif diwakili oleh bit 0 dan tanda negatif diwakili oleh bit 1.

Sebagai contoh, suatu memori dapat menampung 6 bit bilangan biner. Memori tersebut menggunakan sistem bilangan biner bertanda. Maka dari keenam bit yang ada, bit paling kiri, yaitu A<sub>6</sub>, digunakan sebagai penanda bilangan dan dinamakan bit tanda (*sign bit*), sedangkan bit-bit yang lain, yaitu bit A<sub>5</sub> ... A<sub>0</sub> mewakili suatu nilai.

- Bilangan biner 0110100

A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	
0	1	1	0	1	0	0	= +52 <sub>des</sub>
#							
Bit tanda (+)							52 <sub>des</sub>

Bilangan ini merupakan bilangan biner positif karena A<sub>6</sub> = 0, dengan nilai 110100<sub>bin</sub> = +52<sub>des</sub>.

- Bilangan biner 1110100

A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>	
1	1	1	0	1	0	0	= -52 <sub>des</sub>
#							
Bit tanda (-)							52 <sub>des</sub>

Bilangan ini adalah negatif karena A<sub>6</sub> = 1. Nilai bilangan yang diwakili adalah 110100<sub>bin</sub> = 52<sub>des</sub>, sehingga bilangan yang diwakili adalah -52.

Pada sistem bilangan biner bertanda, karena bit paling kiri merupakan bit tanda maka MSB terletak di sebelah kanan bit tanda.

a) Bilangan Biner Komplemen Satu

Terdapat dua cara untuk mengubah suatu bilangan positif ke bilangan negatif, yaitu menggunakan sistem bilangan biner komplemen satu dan sistem bilangan biner komplemen dua. Cara pertama, merupakan cara yang paling mudah ditempuh. Dengan cara ini, untuk mengubah bilangan positif ke negatif cukup dilakukan dengan mengubah bit 0 ke 1 dan bit 1 ke 0 pada setiap bit suatu bilangan biner.



Sebagai contoh, 101101 merupakan bilangan biner dengan nilai 45. Maka -45 sama dengan 010010.

1 0 1 1 0 1 " bilangan biner asli  
↓ ↓ ↓ ↓ ↓ ↓  
0 1 0 0 1 0 " bilangan biner komplemen satu

Jika P merupakan suatu bilangan positif, bilangan komplemen satu n bit -P juga dapat diperoleh dengan mengurangkan P dari 2<sup>n</sup>-1. Atau, bilangan komplemen satunya menjadi (2<sup>n</sup>-1) -P. Contohnya adalah jika P = 45,

$$P = 45_{des} = 101101_{bin}$$
$$\text{Dalam komplemen satu } -P = (2^6 - 1) - 45$$
$$= 111111 - 101101$$
$$= 010010$$

-P (Sistem bilangan komplemen satu jarang digunakan karena tidak memenuhi satu kaedah matematis, yaitu jika suatu bilangan dijumlahkan dengan negatifnya, maka akan dihasilkan bilangan nol.

1 0 1 1 0 1  
+ 0 1 0 0 1 0  
-----  
1 1 1 1 1 1

Pada contoh tersebut, 101101 + 010010 = 111111, sehingga 45 + (-)45 ≠ 0.

b) Bilangan Biner Komplemen Dua

Pada sistem bilangan komplemen dua, penegasan suatu bilangan dilakukan dengan mengubah bit 0 ke 1 dan bit 1 ke 0 pada setiap bit suatu bilangan biner, kemudian menambahkannya dengan satu. Dengan kata lain, bilangan biner komplemen dua didapatkan dari bilangan biner komplemen satu ditambah satu.

Komplemen dua = komplemen satu + 1

Contoh, 101101 merupakan bilangan biner dengan nilai 45. Maka -45 sama dengan 010011.

1 0 1 1 0 1 " bilangan biner asli  
↓ ↓ ↓ ↓ ↓ ↓  
0 1 0 0 1 0 " bilangan biner komplemen satu  
+ 1  
-----  
0 1 0 0 1 1 " bilangan biner komplemen dua

Sebaliknya, pengubahan bilangan biner negatif menjadi bilangan biner positif dilakukan dengan mengurangi bilangan tersebut dengan satu kemudian mengubah bit 0 ke 1 dan bit 1 ke 0 pada setiap bitnya.

Contoh:

0 1 0 0 1 1 " bilangan biner komplemen dua  
- 1  
-----  
0 1 0 0 1 0 " bilangan biner komplemen satu  
↓ ↓ ↓ ↓ ↓ ↓  
1 0 1 1 0 1 " bilangan biner asli

Jika P merupakan suatu bilangan positif, bilangan komplemen dua n bit -P juga dapat diperoleh dengan mengurangkan P dari 2<sup>n</sup>. Atau, bilangan komplemen duanya menjadi 2<sup>n</sup> -P. Contohnya adalah jika P = 45,

$$P = 45_{des} = 101101_{bin}$$
$$\text{Dalam komplemen dua } -P = 2^6 - 45$$
$$= 100000 - 101101$$
$$= 010011$$

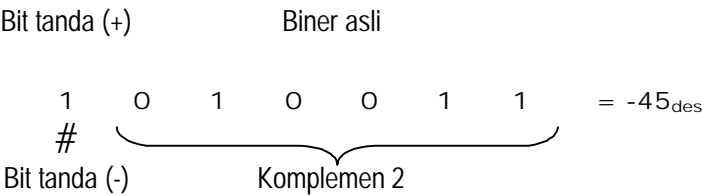
Sistem bilangan biner komplemen dua banyak digunakan dalam sistem digital dan komputer karena memenuhi kaedah matematis, yaitu jika suatu bilangan dijumlahkan dengan negatifnya, maka akan dihasilkan bilangan nol.

1 0 1 1 0 1  
+ 0 1 0 0 1 1  
-----  
1 0 0 0 0 0  
9 bawaan 1 tidak digunakan

Pada penjumlahan tersebut, bit 1 paling depan merupakan bit bawaan dan tidak digunakan. Jadi 101101 + 010011 = 000000, sehingga 45 + (-)45 = 0.

Pada suatu bilangan biner komplemen dua, harus diperhatikan bit tandanya. Jika bit tanda sama dengan 0, maka bit sesudahnya merupakan bentuk bilangan biner asli. Namun jika bit tanda sama dengan 1, maka bit sesudahnya merupakan bentuk bilangan biner komplemen duanya.

0 1 0 1 1 0 1 = +45<sub>des</sub>  
# { }



Terdapat kasus khusus pada sistem bilangan biner komplemen dua. Jika suatu bilangan biner mempunyai bit tanda sama dengan 1, namun bit di belakangnya 0 semua, maka nilai bilangan tersebut adalah  $-2^N$ , dengan N merupakan jumlah bit yang mewakili suatu nilai. Atau sama dengan  $-2^{M-1}$ , dengan M merupakan panjang bit bilangan biner tersebut.

Contoh:

- $10_{bin}$  (panjang 2 bit, 1 bit yang mewakili nilai) =  $-2^1 = -2_{des}$ .
- $1000_{bin}$  (panjang 4 bit, 3 bit yang mewakili nilai) =  $-2^3 = -8_{des}$ .
- $10000000_{bin}$  (panjang 8 bit, 7 bit yang mewakili nilai) =  $-2^7 = -128_{des}$ .

Sehingga jangkauan nilai sistem bilangan biner komplemen dua 4 bit sebagai berikut,

Bilangan desimal	Bilangan biner komplemen dua
+7 = $2^3 - 1$	0111
+6	0110
+5	0101
+4	0100
+3	0011
+2	0010
+1	0001
0	0000
-1	1111
-2	1110
-3	1101
-4	1100
-5	1011
-6	1010
-7	1001
-8 = $-2^3$	1000

c)    Format Bilangan Biner

Bilangan biner biasanya diformat dengan panjang bit tertentu. Panjang bit yang biasa digunakan adalah 2, 4, 8, 16 ... dan seterusnya, atau menurut aturan  $2^n$  dengan n bilangan bulat positif. Namun tetap dimungkinkan bilangan biner dengan format di luar ketentuan tersebut demi kepraktisan atau tujuan khusus.

Pengubahan format bilangan biner komplemen dua dari panjang n-bit menjadi m-bit dengan  $n < m$  mengikuti aturan berikut,

1. Pengubahan format bilangan biner komplemen dua positif dilakukan dengan menambahkan bit 0 di depannya.

Contoh:

4 =	0100	"	format 4 bit
	0000 0100	"	format 8 bit
	0000 0000 0000 0100	"	format 16 bit

2. Pengubahan format bilangan biner komplemen dua negatif dilakukan dengan menambahkan bit 1 di depannya.

Contoh:

-4 =	1100	"	format 4 bit
	1111 1100	"	format 8 bit
	1111 1111 1111 1100	"	format 16 bit

Perlu diingat pada contoh di atas bahwa bit paling depan merupakan bit tanda, sehingga pada format 4 bit hanya ada 3 bit yang merepresentasikan suatu nilai.

2.7    Aritmatika Biner

Operasi aritmatika untuk bilangan biner dilakukan dengan cara hampir sama dengan operasi aritmatika untuk bilangan desimal. Penjumlahan, pengurangan, perkalian dan pembagian dilakukan digit per digit. Kelebihan nilai suatu digit pada proses penjumlahan dan perkalian akan menjadi bawaan (*carry*) yang nantinya

ditambahkan pada digit di sebelah kirinya. Demikian pula sebaliknya kurangnya nilai suatu digit pada proses pengurangan akan 'dipinjam' oleh digit di sebelah kirinya.

a) Penjumlahan

Penjumlahan pada sistem bilangan biner dilakukan dengan cara hampir sama dengan penjumlahan pada sistem bilangan desimal. Bilangan dijumlahkan per bit dari paling kanan (LSB) hingga paling kiri (MSB) tanpa melibatkan tanda. Hal ini berlaku untuk penjumlahan bilangan biner bulat maupun pecahan. Pada operasi penjumlahan, hasil penjumlahan akan mempunyai panjang bit minimal sama dengan bilangan yang dijumlahkan.

Contoh:

1001 (9)	11,011(3,375)
+ 1111 (15)	+ 10,110(2,750)
<u>11000 (24)</u>	<u>110,001(6,125)</u>

Untuk penjumlahan bilangan biner komplemen dua, terdapat beberapa kasus yang harus diperhatikan.

- Penjumlahan bilangan positif

Penjumlahan dua atau lebih bilangan positif mengikuti aturan penjumlahan serupa pada sistem bilangan desimal.

Contoh:

01001 (9)
+ 00100 (4)
<u>01101 (14)</u>

9 bit tanda

- Penjumlahan bilangan positif besar dan bilangan negatif lebih kecil

Perlu diingat di sini untuk mengubah bilangan negatif ke bentuk komplemen dua terlebih dahulu.

Contoh:

01001 (9)
+ 11100 (-4)
<u>± 00101 (5)</u>

9 tidak digunakan

Pada penjumlahan tersebut terdapat bawaan 1. Pada penjumlahan bilangan positif besar dan bilangan negatif lebih kecil, bawaan selalu dihilangkan.

- Penjumlahan bilangan positif kecil dan bilangan negatif lebih besar

Penjumlahan ini akan menghasilkan bit tanda 1 yang berarti hasil merupakan bilangan negatif.

Contoh:

10111 (-9)
+ 00100 (4)
<u>11011 (-5)</u>

9 bit tanda

Perlu diperhatikan di sini, bahwa konversi nilai bilangan biner komplemen dua ke bilangan desimal tidak melibatkan bit tandanya.

- Penjumlahan dua bilangan negatif

Penjumlahan ini akan menghasilkan bit tanda 1 yang berarti hasil merupakan bilangan negatif.

Contoh:

10111 (-9)
+ 11100 (-4)
<u>± 10011 (-13)</u>

9 bawaan 1 tidak digunakan

- Penjumlahan bilangan positif dan negatifnya

Penjumlahan ini akan menghasilkan nol.

Contoh:

01001 (9)
+ 10111 (-9)
<u>± 00000 (0)</u>

9 bawaan 1 tidak digunakan

b) Pengurangan

Pengurangan pada sistem bilangan biner dilakukan dengan menambahkan dengan komplemen duanya.

Contoh:

$$\begin{aligned}
 9 - 4 &= 9 + (-4) \\
 01001_{\text{bin}} - 00100_{\text{bin}} &= 01001_{\text{bin}} + 11100_{\text{bin}} \\
 &\quad 01001 \quad (9) \\
 &+ \underline{11100 \quad (-4)} \\
 &\pm 00101 \quad (5)
 \end{aligned}$$
 9 tidak digunakan

c) Aritmatika Overflow

Bilangan biner dengan panjang bit tertentu mempunyai batasan nilai minimal dan maksimal yang dapat diwakili. Bilangan biner tak bertanda mempunyai nilai maksimal lebih besar daripada bilangan biner komplemen dua. Dan bilangan biner komplemen dua mempunyai batasan minimal lebih besar daripada bilangan biner tak bertanda.

Jika N adalah jumlah bit biner, maka jangkauan nilai sistem bilangan biner tak bertanda N bit adalah

$$0 \text{ sampai } 2^N - 1$$

dan jangkauan nilai sistem bilangan biner komplemen dua N bit adalah

$$-2^{N-1} \text{ sampai } 2^{N-1} - 1$$

Contohnya adalah bilangan biner 4 bit.

Bilangan desimal	Bilangan biner tak bertanda	Bilangan biner komplemen dua
0000	0	0
0001	1	+1
0010	2	+2
0011	3	+3
0100	4	+4
0101	5	+5
0110	6	+6
0111	7	+7
1000	8	-8
1001	9	-7
1010	10	-6
1011	11	-5
1100	12	-4
1101	13	-3
1110	14	-2
1111	15	-1

Dengan panjang 4 bit, jika dipakai sistem bilangan biner tak bertanda, maka jangkauan nilainya adalah 0 sampai 15; namun jika dipakai sistem bilangan biner komplemen dua, maka jangkauan nilainya adalah -8 sampai 7. Dengan demikian pada sistem bilangan biner komplemen dua, separuh kapasitas pada sistem bilangan biner tak bertanda dipakai untuk merepresentasikan bilangan negatif.

Hasil proses aritmatika yang berada di luar jangkauan suatu sistem bilangan biner akan memberikan hasil salah.

Contohnya pada sistem bilangan biner komplemen dua:

$$\begin{aligned}
 &\quad 0100 \quad (4) \\
 &+ \underline{0101 \quad (5)} \\
 &\quad 1001 \quad (5)
 \end{aligned}$$
 " hasil salah karena *overflow*  
 9 bit tanda salah

Pada sistem bilangan desimal, 4 + 5 = 9. Namun karena 9 di luar jangkauan bilangan biner komplemen dua 4 bit, maka penjumlahan di atas akan memberikan hasil salah jika dilakukan.

Kesalahan karena hasil operasi terletak di luar jangkauan suatu sistem bilangan dinamakan *overflow*. Kesalahan ini biasanya akan memberikan hasil berupa bilangan yang mempunyai format bit lebih panjang daripada format bit bilangan yang dijumlahkan.

d) Perkalian

Perkalian bilangan biner dilakukan hampir sama dengan perkalian pada bilangan desimal. Bilangan dikalikan per bit dari paling kanan (LSB) hingga paling kiri

(MSB). Perkalian bilangan biner dilakukan dengan mengalikan bilangan biner aslinya dan tanpa melibatkan bit tanda. Perkalian juga dapat dilakukan oleh suatu bilangan biner negatif dengan bilangan bener positif.

• Perkalian dengan 2

Perkalian dengan dua merupakan operasi aritmatika perkalian paling sederhana. Pada perangkat keras digital, seperti pada sebuah mikroprosesor, perkalian dengan suatu bilangan dapat dibentuk oleh operasi perkalian dengan dua yang dikerjakan berulang kali.

Hasil perkalian ini dapat diperoleh dengan menggeser bilangan yang dikalikan ke kiri satu bit. Bit paling kanan diisi nol.

Contoh:

$$\begin{array}{r} 1001 \quad (9) \\ \times \quad 10 \quad (2) \\ \hline 10010 \quad (18) \end{array}$$

Perkalian suatu bilangan dengan  $2^n$  dapat dilakukan dengan menggeser bilangan yang dikalikan ke kiri n bit.

Contoh:

$\begin{array}{r} 1001 \quad (9) \\ \times \quad 100 \quad (2^2) \\ \hline 100100 \quad (36) \end{array}$	$\begin{array}{r} 1001 \quad (9) \\ \times \quad 1000 \quad (2^3) \\ \hline 1001000 \quad (72) \end{array}$	$\begin{array}{r} 1001 \quad (9) \\ \times \quad 10000000 \quad (2^7) \\ \hline 10010000000 \quad (1152) \end{array}$
---	---	---

Dengan dasar perkalian dengan dua tersebut dapat dibentuk perkalian dengan bilangan lain.

Contoh:

$$\begin{array}{r} 1001 \quad (9) \\ \times \quad 1010 \quad (10) \\ \hline 0000 \\ 1001 \\ 0000 \\ 1001 \\ \hline 1011010 \quad (90) \end{array}$$

Perkalian tersebut dapat dilakukan oleh perangkat keras digital cukup dengan cara melakukan operasi geser kiri dan penjumlahan beberapa kali.

$\begin{array}{r} 1001 \quad (9) \\ \times \quad 1010 \quad (10) \\ \hline 0000 \quad (\text{hasil kali 1}) \\ +1001 \quad (\text{hasil kali 2}) \\ \hline 1001 \quad (\text{subtotal 1}) \\ +0000 \quad (\text{hasil kali 3}) \\ \hline 0100 \quad (\text{subtotal 2}) \\ +1001 \quad (\text{hasil kali 4}) \\ \hline 1011010 \quad (90) \end{array}$	$\begin{array}{c} \downarrow \\ \downarrow \\ \downarrow \\ \downarrow \end{array}$
--	---

Pada operasi 1001 X 1010, pertama akan dilakukan 1001 X 0. Hasilnya adalah 0000 (hasil kali 1). Kemudian dilakukan perkalian 1001 X 10, hasilnya 1001 yang digeser ke kiri satu bit (hasil kali 2). Hasil kali 1 dan 2 dijumlahkan dengan format bit mengikuti hasil kali 2 menjadi subtotal 1. Selanjutnya dilakukan perkalian 1001 X 000, hasilnya 0000 yang digeser kiri satu bit terhadap sub total 3 (hasil kali 3). Subtotal 1 dan hasil kali 3 dijumlahkan dengan format bit mengikuti hasil kali 3 menjadi subtotal 2. Terakhir dilakukan perkalian 1001 X 1000, hasilnya 1001 yang digeser kiri satu bit terhadap subtotal 2. Akhirnya dilakukan penjumlahan untuk bit paling kanan hasil kali 1, subtotal 1, dan semua bit subtotal 2 dan hasil kali 4.

Pada operasi perkalian bilangan biner dengan panjang bit sama, hasil akan mempunyai format panjang bit maksimal dua kali panjang bit pengalinya. Jika suatu bioangan biner n bit dikalikan bilangan biner m bit, maka hasil kalinya mempunyai panjang bit maksimal n+m bit.

Untuk perkalian dengan bilangan negatif, dapat dilakukan dengan mengalikannya dengan bilangan positifnya. Bit tanda hasil perkalian dapat disesuaikan setelah hasil perkalian didapat.

• Perkalian bilangan positif dengan bilangan negatif atau sebaliknya

Perkalian dapat dilakukan dengan bilangan biner aslinya. Sebagai contoh, 9 X (-10) akan meberikan hasil yang sama dengan 9 X 10, hasilnya dinegasikan. Pada kasus ini akan dihasilkan bilangan negatif.

Contoh:

01001 (9)

"

1001 (9)

x 10110 (-10)

"

x 1010 (10)

\$

0000

bit tanda

1001

0000

1001

1011010 (90)

\$

Setelah dinegasikan 1110100110 (-90)  
9 bit tanda

Perkalian juga dapat dilakukan dengan mengalikan (-9) X 10. Pengerjaan perkalian tersebut sebagai berikut,

10111 (-9)

x 01010 (10)

0000000 (hasil kali 1)

110111 (hasil kali 2)

110111 (subtotal 1)

00000 (hasil kali 3)

1111011 (subtotal 2)

110111 (hasil kali 4)

1110100 (subtotal 3)

000000 (hasil kali 5)

1110100110 (-90)

Pertama kali dilakukan 10111 X 0, hasilnya 00000. Kemudian dilakukan perkalian 10111 X 10, hasilnya 10111 yang digeser ke kiri satu bit. Karena hasil kali 10111 (-9) dan 10 ini harus negatif, maka tambahkan 1 di depan (hasil kali 2). Hasil kali 1 dan hasil kali 2 dijumlahkan menghasilkan subtotal 1. Kemudian dilakukan perkalian 10111 dan 000, hasilnya 00000 yang digeser ke kiri satu bit terhadap subtotal 1 (hasil kali 3). Subtotal 1 dan hasil kali 3 dijumlahkan menghasilkan subtotal 2. Kemudian dilakukan perkalian 10111 X 1000, hasilnya 10111 yang digeser 1 bit terhadap subtotal 2. Karena hasil kali 10111 dan 1000 ini harus negatif, maka tambahkan 1 di depan (hasil kali 4). Selanjutnya subtotal 2 dan hasil kali 4 dijumlahkan (subtotal 3). Penjumlahan ini dilakukan dengan terlebih dahulu menambahkan bit 11 di depan subtotal 2 (seperti diuraikan pada subbab 2.6(c), penambahan bit 1 di depan bilangan biner negatif tidak mengubah arti). Terakhir, dilakukan perkalian 10111 dengan 00000, hasilnya 00000 yang digeser ke kiri satu bit terhadap subtotal 3. Akhirnya dilakukan penjumlahan untuk bit paling kanan hasil kali 1, subtotal 1, subtotal 2, dan semua bit subtotal 3 dan hasil kali 5. Hasil akhir merupakan bilangan negatif, untuk itu tambahkan 1 di depan sehingga dihasilkan format bilangan biner 10 bit.

Sebagai catatan, suatu bilangan biner negatif dapat dikalikan dengan suatu bilangan biner positif dengan cara tersebut. Namun cara tersebut tidak berlaku untuk suatu bilangan biner positif yang dikalikan dengan suatu bilangan biner negatif.

- Perkalian bilangan negatif dengan bilangan negatif

Pada kasus ini akan dihasilkan bilangan positif.

Contoh:

10111 (-9)

"

1001 (9)

x 10110 (-10)

"

x 1010 (10)

\$

0000

bit tanda

1001

0000

1001

1011010 (90)

Meskipun hasil perkalian mempunyai panjang bit dua kali bilangan pengalinya, namun seperti pada sistem bilangan desimal, bila perlu bit 0 paling depan dapat dihilangkan.

18

e) Pembagian

Pembagian bilangan biner dilakukan hampir sama dengan pembagian pada bilangan desimal. Bilangan dibagi per bit dari paling kiri sesudah bit tanda (MSB) hingga paling kanan (LSB). Pembagian bilangan biner dilakukan dengan membagi bilangan biner aslinya dan tanpa melibatkan bit tanda.

Contoh:

0011

11

1001

011

0011

11

0

1001

1010

1011011

1010

10

0

101

0

1011

1010

1

" sisa pembagian

Untuk pembagian dengan bilangan negatif, dapat dilakukan dengan pembagian dengan bilangan positifnya. Bit tanda hasil pembagian dapat disesuaikan setelah hasil pembagian didapat.

Kasus khusus terjadi untuk pembagian dengan dua. Pembagian ini merupakan operasi aritmatika pembagian paling sederhana. Hasil pembagian ini dapat diperoleh dengan menggeser bilangan yang dikalikan ke kanan satu bit. Bit paling kanan akan hilang pada penggeseran ini.

Pada perangkat keras digital, seperti pada sebuah mikroprosesor, pembagian dengan 2<sup>n</sup> dapat dibentuk oleh operasi pembagian dengan dua yang dikerjakan berulang kali.

Contoh:

1000

÷ 10

100

(8)

(2)

(4)

1000

÷ 100

10

(8)

(2<sup>2</sup>)

(2)

1001

÷ 10

100

(9)

(2)

(4)

Jika bilangan yang dibagi merupakan bilangan ganjil, maka operasi penggeseran ke kanan satu hanya memberikan hasil bagian bulatnya (*integer*) saja.

f) Penjumlahan BCD

Pada sistem bilangan BCD, penjumlahan dan operasi aritmatika yang lain dilakukan dengan cara hampir sama dengan cara melakukan penjumlahan atau operasi yang lain pada sistem bilangan biner. Sedikit perbedaan di sini adalah penanganan hasil penjumlahan untuk digit desimal 10 ke atas.

Penjumlahan Bilangan Kurang dari 10

Penjumlahan bilangan dengan hasil di bawah 10 dilakukan seperti penjumlahan biasa.

Contoh:

4

+ 5

9

"

"

"

0100

+ 0101

1001

33

+ 54

87

"

"

"

0011 0011

+ 0101 0100

1000 0111

Penjumlahan Bilangan 10 atau Lebih

Penjumlahan bilangan dengan hasil 10 atau lebih akan memberikan hasil yang salah untuk bilangan BCD.

Contoh:

7

+ 6

9

"

"

"

0111

+ 0110

1101

" bukan format bilangan BCD

Untuk itu, penjumlahan dengan hasil bilangan 10 atau lebih harus ditambah dengan 6 untuk memberikan bawaan ke digit di depannya.

Contoh:

7

+ 6

13

"

"

"

0111

+ 0110

1101

" bukan format bilangan BCD

0001 0011

+ 0110

0001 0011

" tambahkan dengan 6

" hasil penjumlahan

Penambahan dengan 6 ini juga berlaku untuk bilangan BCD yang mempunyai digit lebih dari satu.

Contoh:

572	"	0101 0111 0010	
+ 146	"	+ 0001 0100 0110	
718		0110 1011 1000	
		+ 0110	" tambahkan dengan 6
		0111 0001 1000	" hasil penjumlahan

2.8 Aritmatika Heksadesimal

a) Penjumlahan

Penjumlahan dalam sistem bilangan heksadesimal mempunyai kemiripan dengan penjumlahan pada sistem bilangan desimal. Yang perlu diperhatikan di sini adalah penggunaan tambahan digit A hingga F pada sistem bilangan ini.

Contoh:

85	FA3
+ 42	+ C32
C7	1BD5

b) Pengurangan

Pengurangan pada sistem bilangan heksadesimal dapat dilakukan dengan menjumlahkan dengan komplemen duanya. Menentukan komplemen dua dari suatu bilangan heksadesimal dapat dilakukan dengan mengonversi bilangan tersebut ke bilangan biner terlebih dahulu.

Contoh:

5	3	7	
↙	↓	↘	
0101	0011	0111	
1010	1100	1001	" komplemen duanya
↘	↓	↙	
A	C	9	

Ada cara cepat untuk mendapatkan komplemen dua dari suatu bilangan heksadesimal, yaitu dengan mengurangi setiap digit heksadesimal dari F dan menambahkannya dengan 1. Contoh:

F	F	F
- 5	- 3	- 7
A	C	8
		+ 1
A	C	9

Bilangan heksadesimal komplemen dua banyak digunakan untuk menyajikan nilai terutama pada isi lokasi memori. Nilai suatu isi lokasi memori akan lebih mudah dibaca jika disajikan dalam format heksadesimal dibandingkan jika disajikan dalam format biner.

Contoh:

Alamat memori	Data tersimpan	Format heksadesimal
1000	0011 1011	3B
1001	1010 0101	A5

Ciri bilangan negatif pada sistem bilangan biner komplemen dua adalah MSB=1, sedangkan ciri bilangan negatif pada sistem bilangan heksadesimal komplemen dua adalah MSD≥8.

Soal-soal Latihan

- 2-1 Ubahlah setiap bilangan berikut menjadi bilangan biner, oktal, heksadesimal dan BCD.
- a. 4
  - b. 15
  - c. 36
  - d. 109
  - e. 1024
- 2-2 Ubahlah setiap bilangan biner tak bertanda berikut menjadi bilangan desimal.
- a. 00111
  - b. 10001



- c. 10110
  - d. 01001
  - e. 11110
- 2-3 Ubahlah bilangan biner pada Soal nomor 2-2 menjadi bilangan biner 8 bit.
- 2-4 Jika setiap bilangan biner pada soal nomor 2-2 menggunakan sistem bilangan biner komplemen dua, ubahlah menjadi bilangan desimal.
- 2-5 Jika setiap bilangan biner pada soal nomor 2-2 menggunakan sistem bilangan biner komplemen dua, ubahlah menjadi bilangan biner 8 bit.
- 2-6 Ubahlah bilangan biner pada Soal 2-2 menjadi bilangan oktal dan heksadesimal tanpa terlebih dahulu mengubahnya menjadi bilangan desimal.
- 2-7 Ubahlah setiap bilangan oktal berikut menjadi bilangan desimal.
- a. 74
  - b. 105
  - c. 372
  - d. 7623
  - e. 2643
- 2-8 Ubahlah bilangan oktal pada Soal 2-7 menjadi bilangan biner tanpa terlebih dahulu mengubahnya menjadi bilangan desimal.
- 2-9 Ubahlah setiap bilangan heksadesimal berikut menjadi bilangan desimal.
- a. 7
  - b. 1E
  - c. A7F
  - d. 1BC5
  - e. F01D1
- 2-10 Ubahlah bilangan oktal pada Soal 2-9 menjadi bilangan biner tanpa terlebih dahulu mengubahnya menjadi bilangan desimal.
- 2-11 Kerjakan operasi matematis berikut.
- a.  $10010+10001$
  - b.  $10111+00101$
  - c.  $00100+00111$
  - d.  $01110+10011$
  - e.  $10001+01111$
- 2-12 Jika setiap bilangan biner pada soal nomor 2-11 menggunakan sistem bilangan biner komplemen dua, ulangi operasi matematis tersebut.
- 2-13 Ulangi Soal nomor 2-11 namun untuk operasi perkalian.
- 2-14 Ulangi Soal nomor 2-11 namun untuk operasi perkalian menggunakan sistem bilangan biner komplemen dua.
- 2-15 Ulangi Soal nomor 2-11 dan berikan hasilnya dalam format BCD.
- 2-16 Bagaimana cara mengubah suatu bilangan biner negatif komplemen dua ke bilangan desimal?
- 2-17 Ubahlah bilangan biner berikut menjadi bilangan desimal:
- a. 1001,1001
  - b. 10011011001,10110
- 2-18 Berapa nilai maksimal untuk bilangan biner 1, 2, 4, 8, 16 dan 32 bit?
- 2-19 Berapa bit yang dibutuhkan untuk melakukan penghitungan hingga 511?
- 2-20 Dengan format 8 bit, berapa nilai terkecil dan terbesar jika menggunakan sistem bilangan biner tak bertanda dan komplemen dua?
- 2-21 Berapa bit yang dibutuhkan untuk merepresentasikan bilangan dari 0 hingga 999 menggunakan kode biner dan BCD?

- 2-22 Berapa digit yang dibutuhkan untuk merepresentasikan bilangan dari 0 hingga 999 menggunakan kode oktal dan heksadesimal?
- 2-23 Suatu nilai tegangan akan diwakili oleh bilangan biner. Berapa nilai tegangan yang dapat diwakili jika digunakan bilangan biner:
- a. 8 bit
  - b. 10 bit

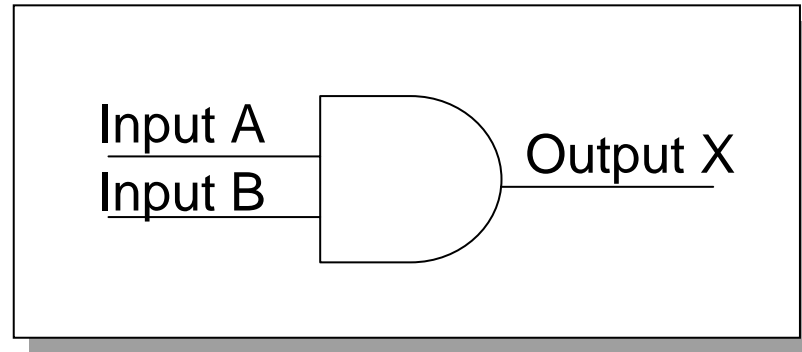
# GERBANG LOGIKA DASAR

Gerbang Logika → blok dasar untuk membentuk rangkaian elektronika digital

- Sebuah gerbang logika mempunyai satu terminal output dan satu atau lebih terminal input
- Output-outputnya bisa bernilai HIGH (1) atau LOW (0) tergantung dari level-level digital pada terminal inputnya.
- Ada 7 gerbang logika dasar : AND, OR, NOT, NAND, NOR, Ex-OR, Ex-NOR



## Gerbang AND

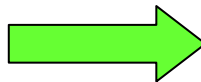


Simbol gerbang logika AND

### Operasi AND :

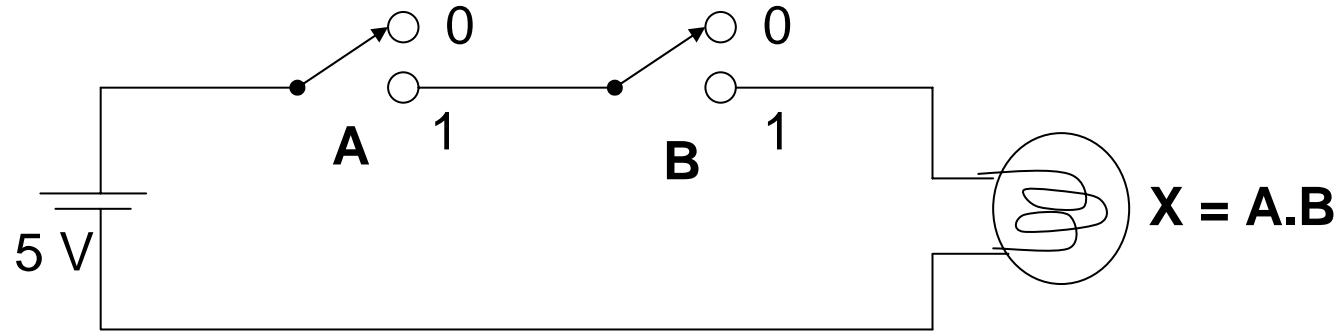
- Jika Input A AND B keduanya **HIGH**, maka output X akan **HIGH**
- Jika Input A atau B salah satu atau keduanya **LOW** maka output X akan **LOW**

Tabel Kebenaran  
gerbang AND – 2 input

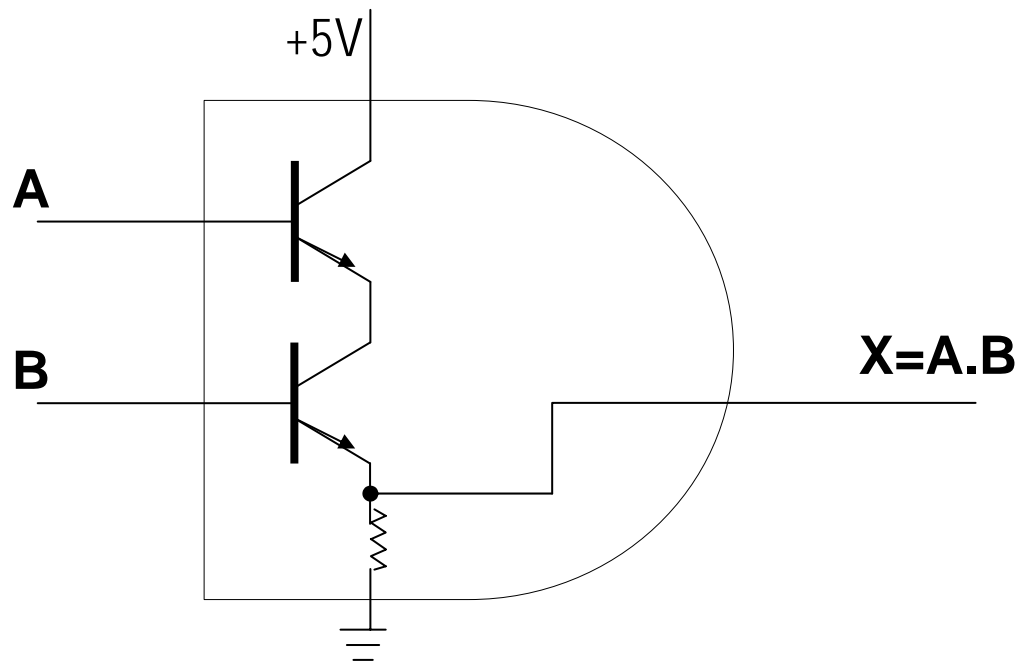


INPUT		Output X
A	B	
0	0	0
0	1	0
1	0	0
1	1	1

# Cara kerja Gerbang AND :

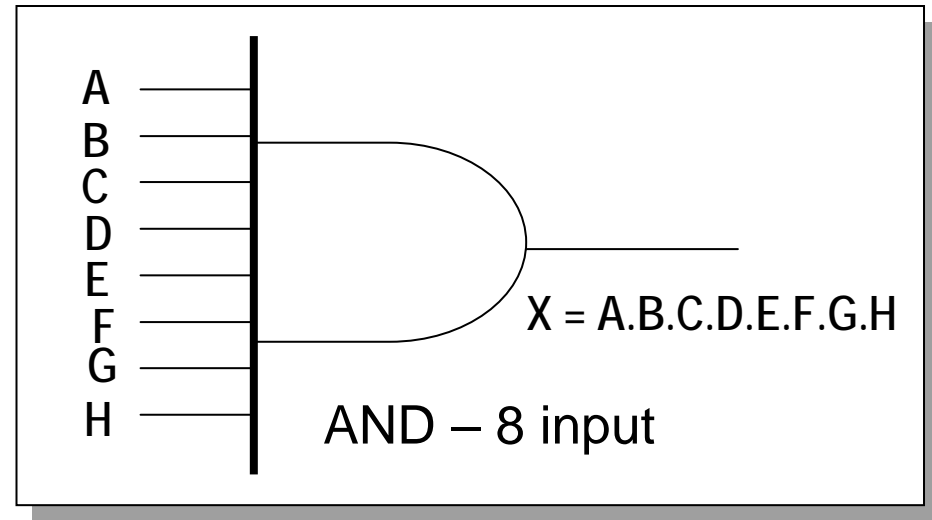
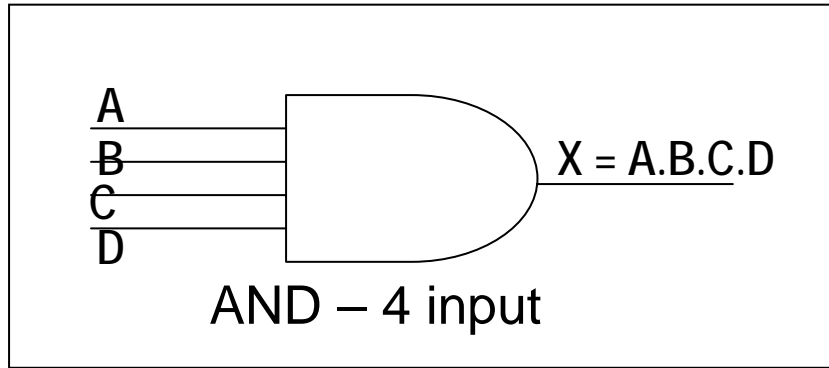


Analogi elektrik gerbang AND



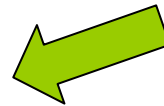
Gerbang AND dengan switch Transistor

# Gerbang AND dengan banyak Input

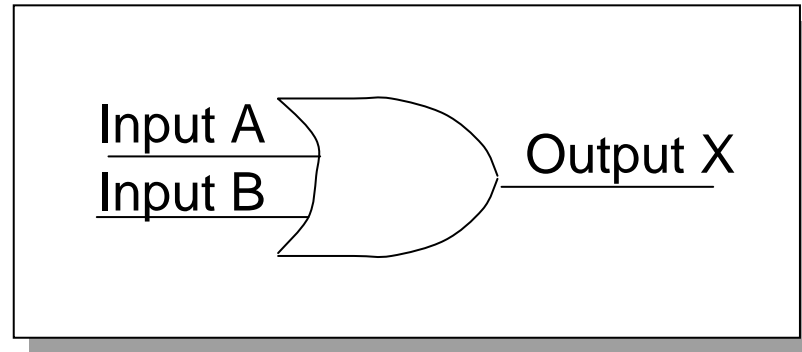


INPUT				Output
A	B	C	D	X
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Tabel Kebenaran AND-4 input



## Gerbang OR



Simbol gerbang logika OR

### Operasi OR :

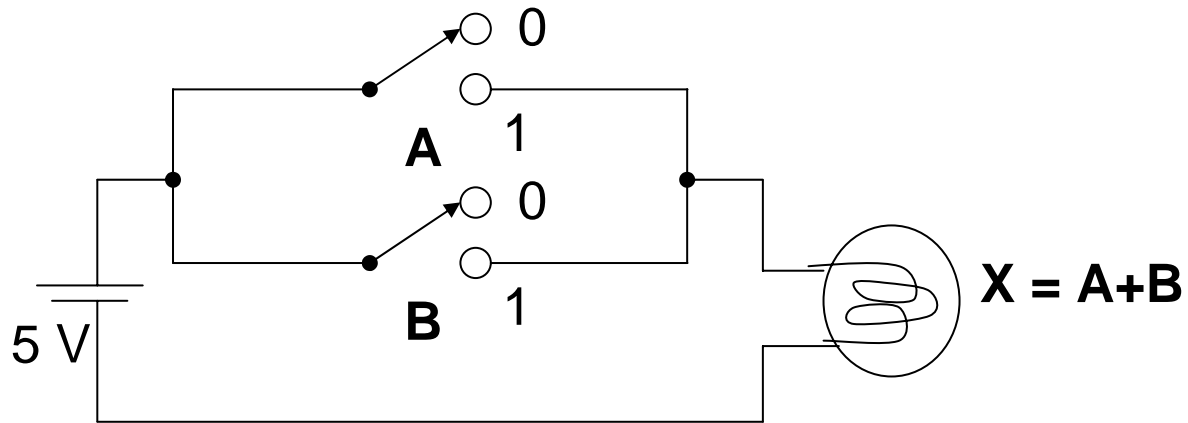
- Jika Input A OR B atau keduanya **HIGH**, maka output X akan **HIGH**
- Jika Input A dan B keduanya **LOW** maka output X akan **LOW**

Tabel Kebenaran  
gerbang OR – 2 input

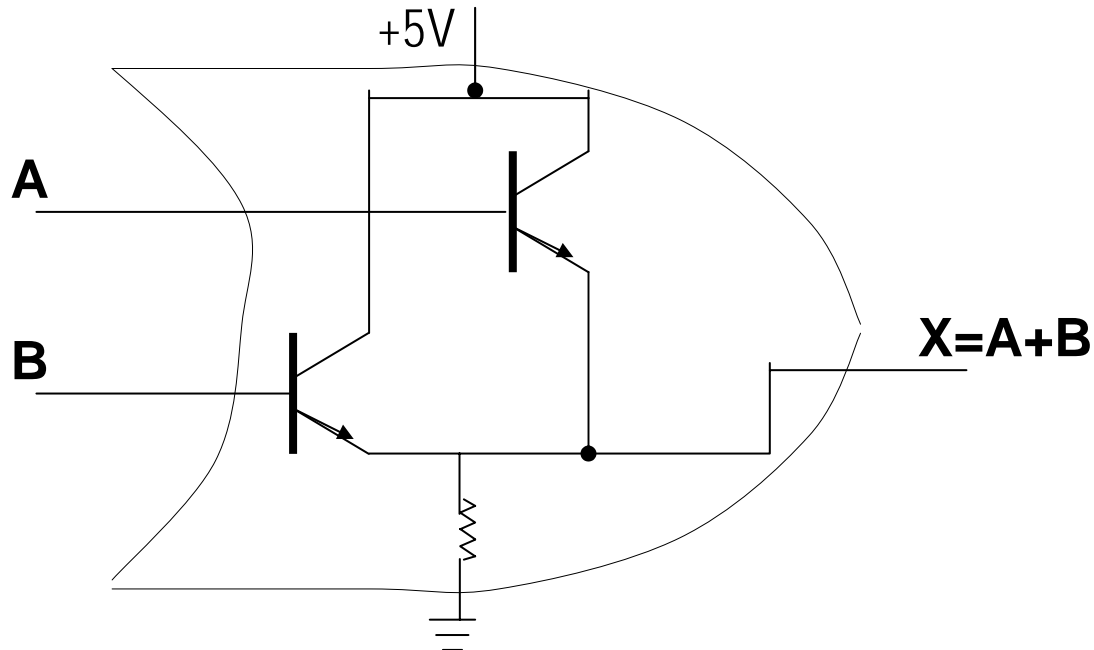


INPUT		Output
A	B	X
0	0	0
0	1	1
1	0	1
1	1	1

# Cara kerja Gerbang OR :



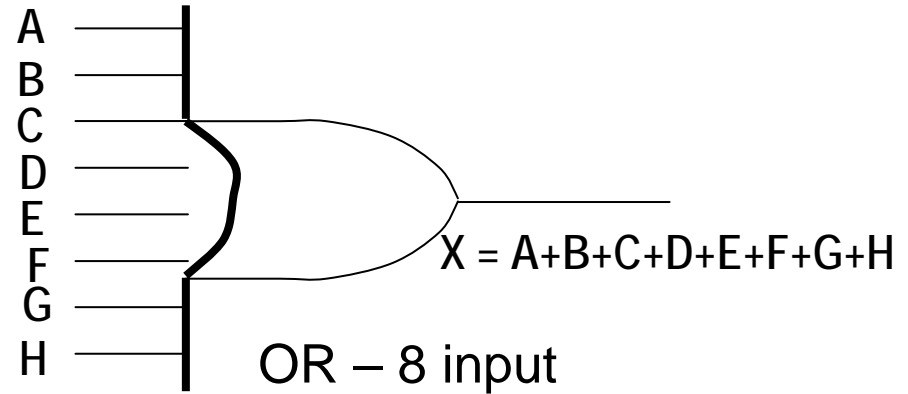
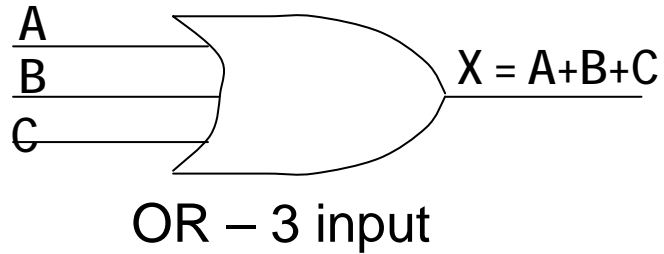
Analogi elektrik gerbang OR



Gerbang OR dengan switch Transistor



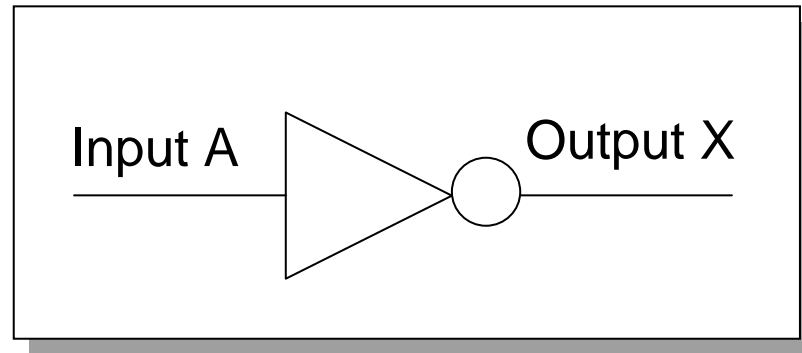
# Gerbang OR dengan banyak Input



Tabel Kebenaran OR-3 input

INPUT			Output
A	B	C	X
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1

# Gerbang NOT / INVERTER



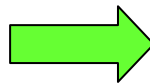
Simbol gerbang logika NOT

## Operasi NOT :

- Jika Input A **HIGH**, maka output X akan **LOW**
- Jika Input A **LOW**, maka output X akan **HIGH**

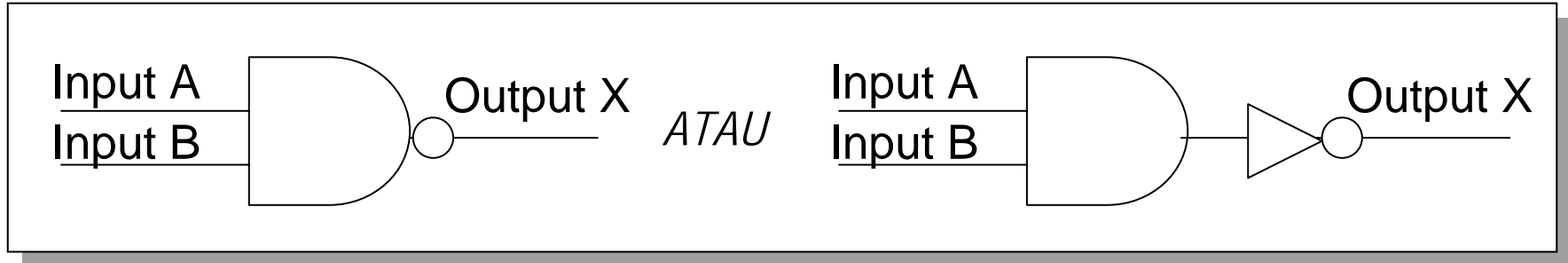
$$X = \overline{A}$$

Tabel Kebenaran  
gerbang NOT / INVERTER



INPUT A	Output X
0	1
1	0

## Gerbang NAND

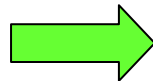


Simbol gerbang logika NAND

### Operasi NAND :

- Merupakan Inversi (kebalikan) dari operasi AND
- Jika Input A AND B keduanya **HIGH**, maka output X akan **LOW**
- Jika Input A atau B atau keduanya **LOW**, maka output X akan **HIGH**

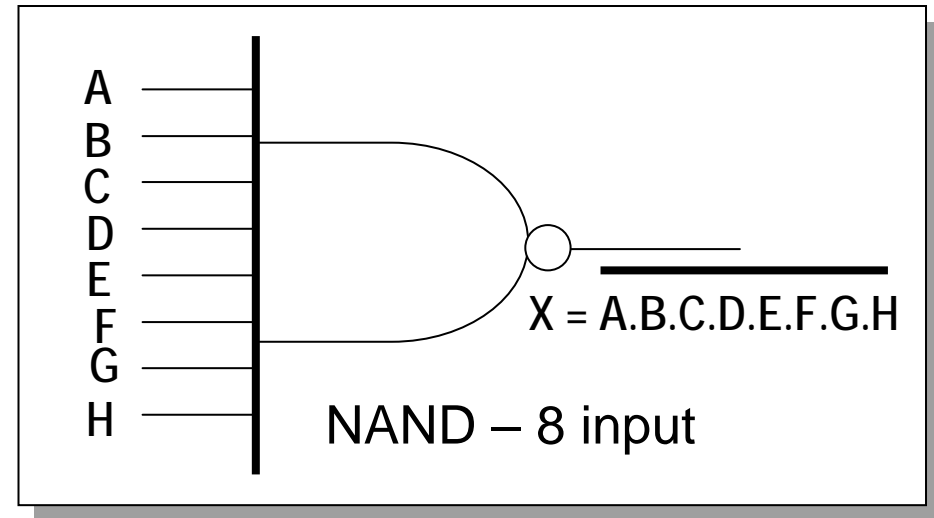
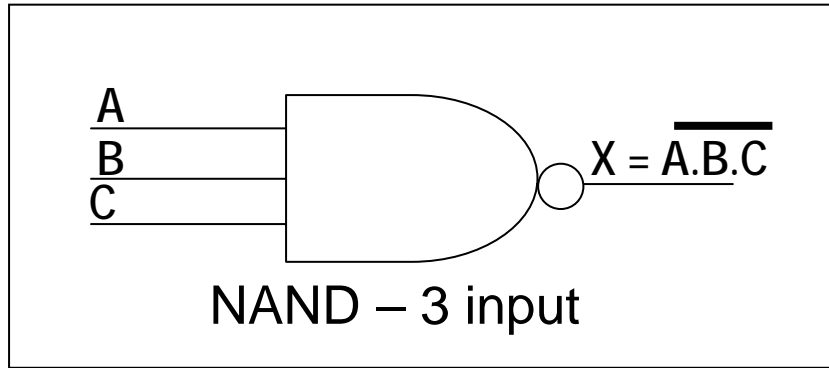
Tabel Kebenaran  
gerbang NAND



INPUT		Output
A	B	X
0	0	1
0	1	1
1	0	1
1	1	0

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

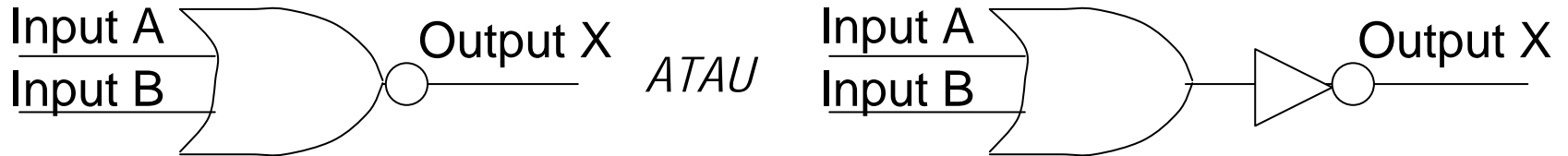
# Gerbang NAND dengan banyak Input



Tabel Kebenaran NAND-3 input

INPUT			Output X
A	B	C	
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

## Gerbang NOR

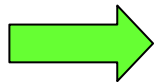


Simbol gerbang logika NOR

### Operasi NOR :

- Merupakan Inversi (kebalikan) dari operasi OR
- Jika Input A dan B keduanya **LOW**, maka output X akan **HIGH**
- Jika Input A OR B salah satu atau keduanya **HIGH**, maka output X akan **LOW**

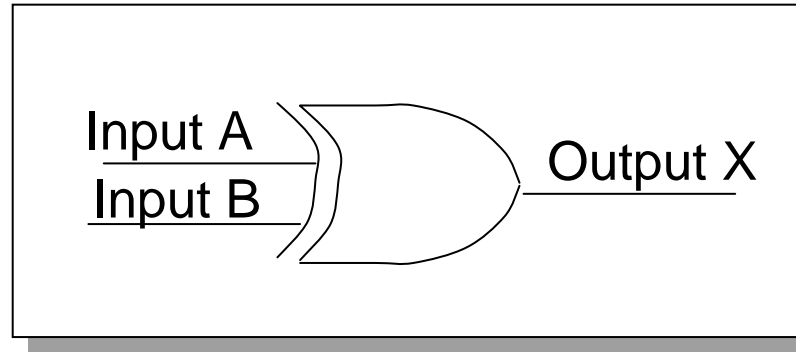
Tabel Kebenaran  
gerbang NOR



INPUT		Output X
A	B	
0	0	1
0	1	0
1	0	0
1	1	0

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

## Gerbang Ex-OR



Simbol gerbang logika Ex-OR

### Operasi Ex-OR :

- Ex-OR adalah kependekan dari Exclusive OR
- Jika salah satu dari kedua inputnya HIGH (bukan kedua-duanya), maka output X akan HIGH
- Jika kedua inputnya bernilai LOW semua atau HIGH semua, maka output X akan LOW

## Tabel Kebenaran Gerbang Ex-OR

INPUT		OUTPUT
A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

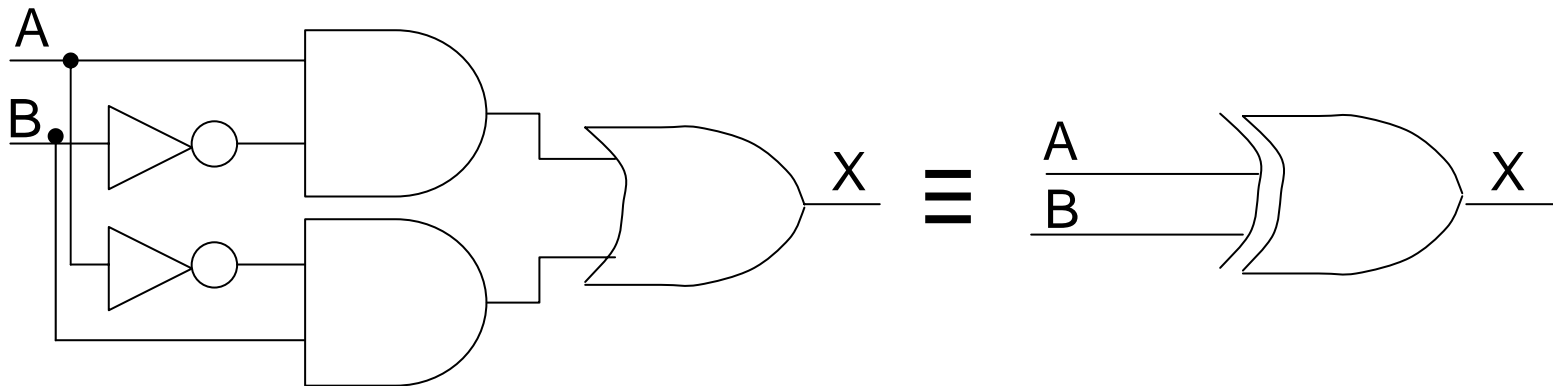
Persamaan Logika Ex-OR

$$X = A \oplus B$$

Berdasarkan Tabel Kebenaran di atas (yang bernilai output = 1), Ex-OR dapat disusun dari gerbang dasar : AND, OR dan NOT

Persamaan EX-OR (dari AND, OR dan NOT) :

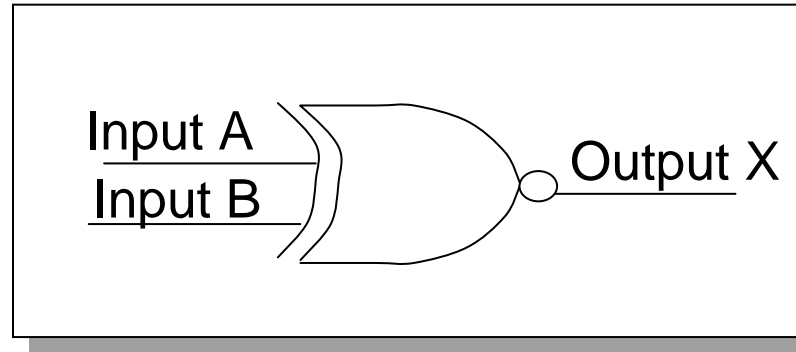
$$X = \overline{A}B + A\overline{B}$$



Gerbang Ex-OR dari AND, OR, NOT

Simbol logika Ex-OR

## Gerbang Ex-NOR



Simbol gerbang logika Ex-NOR

### Operasi Ex-NOR :

- Ex-NOR merupakan kebalikan dari Ex-OR
- Jika salah satu dari kedua inputnya HIGH (bukan kedua-duanya), maka output X akan LOW
- Jika kedua inputnya bernilai LOW semua atau HIGH semua, maka output X akan HIGH



## Tabel Kebenaran Gerbang Ex-NOR

INPUT		OUTPUT
A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

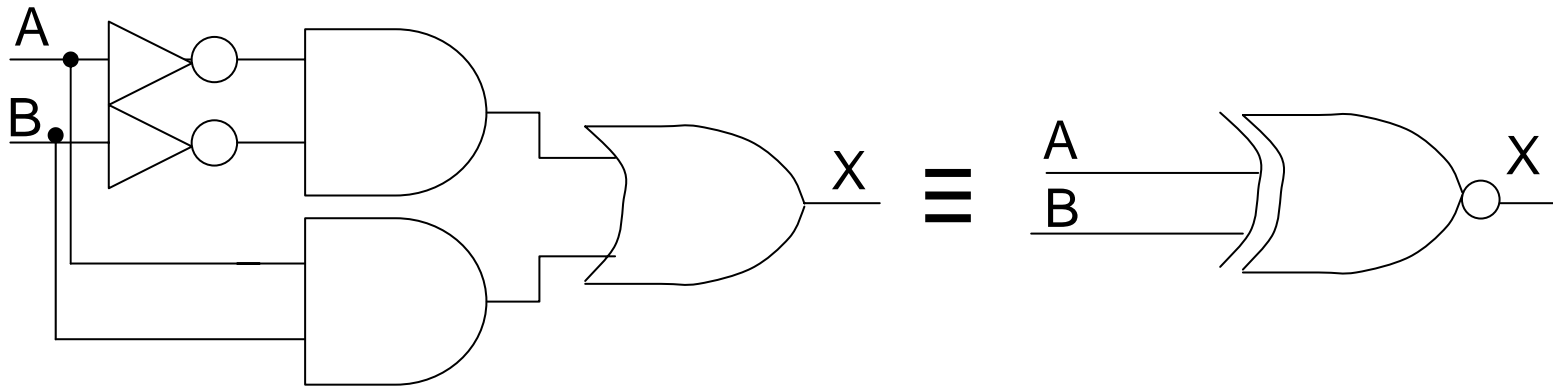
Persamaan Logika Ex-NOR

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

Berdasarkan Tabel Kebenaran di atas (yang bernilai output = 1), Ex-NOR dapat disusun dari gerbang dasar : AND, OR dan NOT

Persamaan EX-NOR (dari AND, OR dan NOT) :

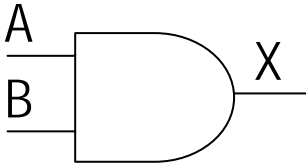
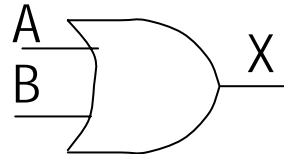
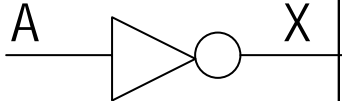
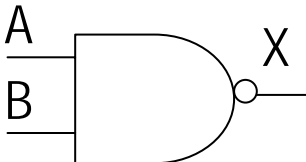
$$X = \overline{A} \overline{B} + AB$$



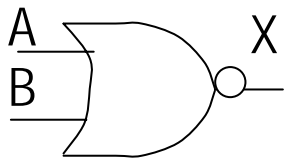
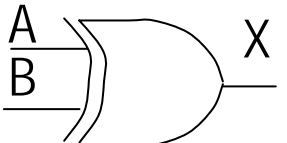
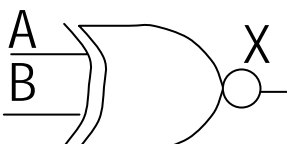
Gerbang Ex-NOR dari AND, OR, NOT

Simbol logika Ex-NOR

# RINGKASAN JENIS GERBANG LOGIKA

No	NAMA	TIPE IC	Simbol Logika	Persamaan	Tabel Kebenaran																		
1	AND	7408		$X=A.B$	<table><tr><th colspan="2">INPUT</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	INPUT		Output	A	B	X	0	0	0	0	1	0	1	0	0	1	1	1
INPUT		Output																					
A	B	X																					
0	0	0																					
0	1	0																					
1	0	0																					
1	1	1																					
2	OR	7432		$X=A+B$	<table><tr><th colspan="2">INPUT</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	INPUT		Output	A	B	X	0	0	0	0	1	1	1	0	1	1	1	1
INPUT		Output																					
A	B	X																					
0	0	0																					
0	1	1																					
1	0	1																					
1	1	1																					
3	NOT	7404		$X=\overline{A}$	<table><tr><th>INPUT</th><th>Output</th></tr><tr><th>A</th><th>X</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	INPUT	Output	A	X	0	1	1	0										
INPUT	Output																						
A	X																						
0	1																						
1	0																						
4	NAND	7400		$X=\overline{A.B}$	<table><tr><th colspan="2">INPUT</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	INPUT		Output	A	B	X	0	0	1	0	1	1	1	0	1	1	1	0
INPUT		Output																					
A	B	X																					
0	0	1																					
0	1	1																					
1	0	1																					
1	1	0																					

# RINGKASAN JENIS GERBANG LOGIKA.....*cont*

No	NAMA	TIPE IC	Simbol Logika	Persamaan	Tabel Kebenaran																		
5	NOR	7402		$X = \overline{A+B}$	<table><tr><th colspan="2">INPUT</th><th>Output</th></tr><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	INPUT		Output	A	B	X	0	0	1	0	1	0	1	0	0	1	1	0
INPUT		Output																					
A	B	X																					
0	0	1																					
0	1	0																					
1	0	0																					
1	1	0																					
6	Ex-OR	7486		$X = A \oplus B$	<table><tr><th colspan="2">INPUT</th><th>OUTPUT</th></tr><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table>	INPUT		OUTPUT	A	B	X	0	0	0	0	1	1	1	0	1	1	1	0
INPUT		OUTPUT																					
A	B	X																					
0	0	0																					
0	1	1																					
1	0	1																					
1	1	0																					
7	Ex-NOR			$X = \overline{A \oplus B}$	<table><tr><th colspan="2">INPUT</th><th>OUTPUT</th></tr><tr><th>A</th><th>B</th><th>X</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	INPUT		OUTPUT	A	B	X	0	0	1	0	1	0	1	0	0	1	1	1
INPUT		OUTPUT																					
A	B	X																					
0	0	1																					
0	1	0																					
1	0	0																					
1	1	1																					

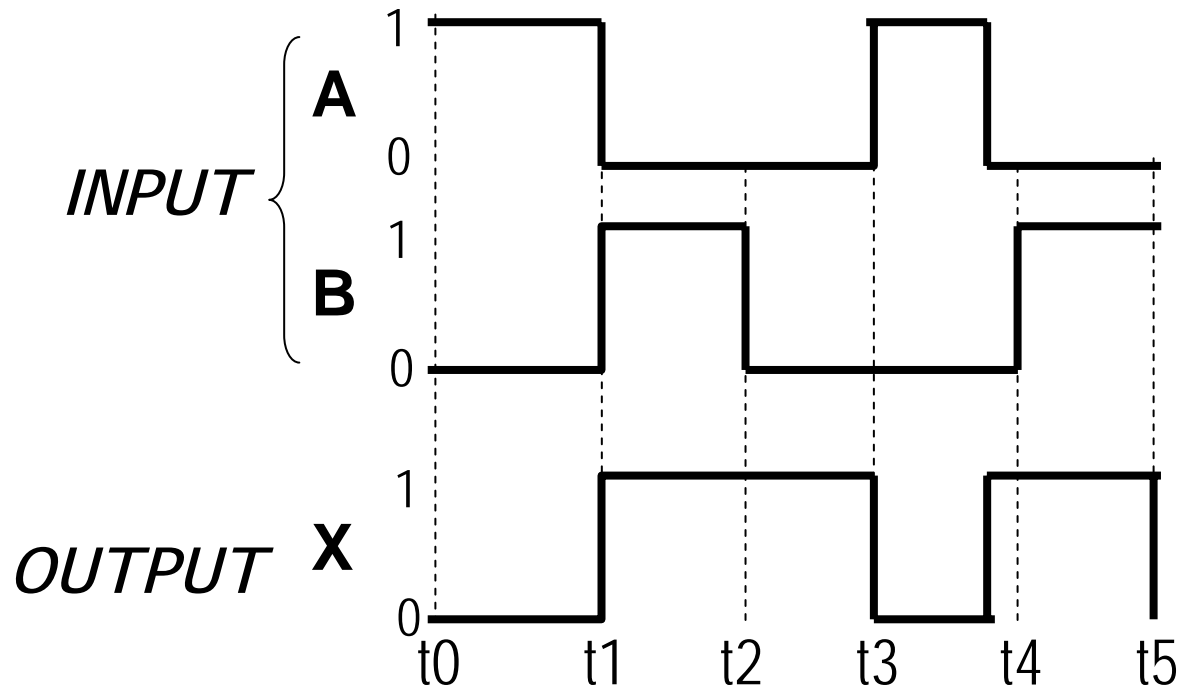
## ANALISA PE-WAKTU-AN

Cara penganalisaan response output terhadap kombinasi input-inputnya pada periode waktu tertentu,

Cara penganalisaan yang lain adalah dengan Tabel Kebenaran

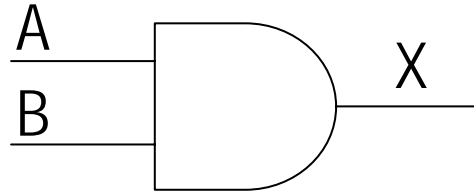
Peralatan yang digunakan disebut : *Timing Diagram* (Diagram pe-waktu-an).

Bentuk Timing Diagram :

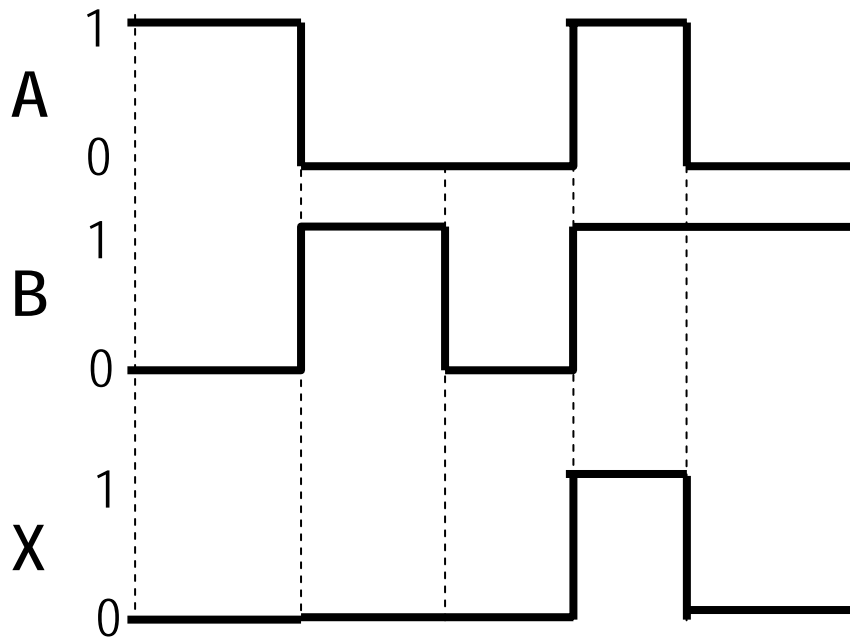


Contoh :

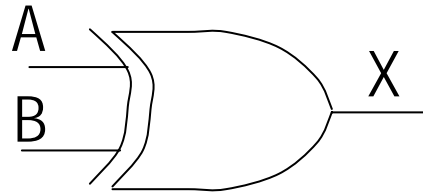
1. Buatlah timing diagram untuk mendapatkan output dari gerbang AND berikut ini :



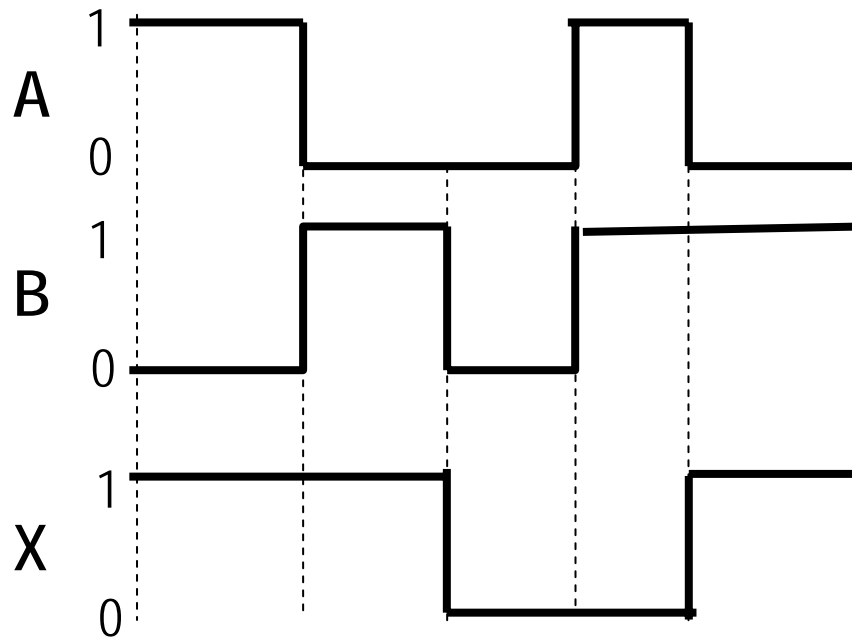
Jawab :



2. Buatlah timing diagram untuk mendapatkan output dari gerbang Ex-OR berikut ini :

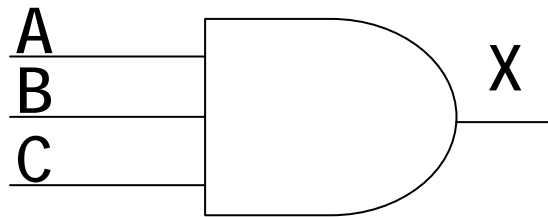


Jawab :

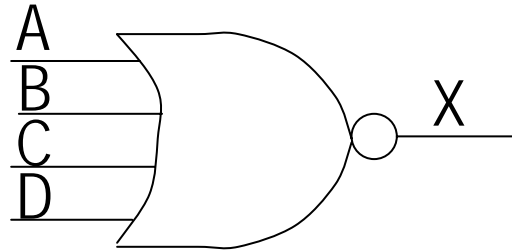


Soal Latihan :

1. Sebuah input data mempunyai urutan : 101110010. Gambarkan bentuk gelombang dari data input tersebut dalam representasi sinyal digital.
2. Sebutkan 3 jenis aplikasi yang menggunakan teknologi digital.
3. Buat Tabel Kebenaran untuk gerbang AND-3 input berikut ini :



4. Buat Tabel Kebenaran untuk gerbang NOR-4 input berikut ini :



5. Buat Timing Diagram untuk output X dari gerbang OR-3 input berikut ini :

