Diktat Teknik Digital: Sistem Bilangan dan Representasi Data

Chapter · April 2018

CITATIONS
0

READS
3,884

1 author:

Freddy Kurniawan
School of Technology Adisutjipto, Indonesia, Yogyakarta
24 PUBLICATIONS **17** CITATIONS

SEE PROFILE

Bab 2

Sistem Bilangan dan Representasi Data

Setelah mempelajari bab ini, diharapkan mahasiswa dapat:

- Memahami perbedaan sistem bilangan desimal, biner, oktal, heksadesimal dan BCD
- Melakukan konversi antar beberapa sistem bilangan
- Merepresentasikan suatu nilai pecahan dalam bilangan biner, oktal dan heksadesimal
- Memahami konsep bilangan komplemen satu dan dua
- Merepresentasikan suatu nilai positif dan negatif dalam bilangan biner, oktal dan heksadesimal
- Memilih sistem bilangan yang digunakan secara tepat sesuai dengan kegunaannya.
- Memahami sistem bilangan presisi tunggal dan presisi ganda.
- Melakukan operasi penjumlahan, pengurangan, perkalian dan pembagian dalam sistem bilangan biner.
- Melakukan operasi penjumlahan dalam sistem bilangan BCD.
- Melakukan operasi matematika dalam sistem bilangan heksadesimal.
- Memahami pengkodean sistem ASCII.

Perangkat digital dapat mengenali, mengolah dan menyimpan informasi yang didapat dari alam seperti: bunyi, cahaya, suhu dan tekanan. Informasi tersebut disimpan dalam suatu data numeris yang berupa angka-angka yang diformat dalam sistem bilangan tertentu agar mudah dimengerti sistem. Terdapat beberapa sistem bilangan yang biasa digunakan dalam bidang teknik, yaitu: sistem bilangan biner, oktal, heksadesimal dan tentu saja desimal.

Sistem bilangan desimal biasa digunakan untuk menyatakan kuantitas dalam keidupan sehari-hari. Sementara itu tiga sistem bilangan lain jarang digunakan dalam kehidupan sehari-hari kecuali untuk keperluan khusus.

Sistem bilangan biner merupakan salah satu sistem bilangan yang biasa dipakai dalam sistem digital. Karena dalam sistem digital dikenal logika 0 dan 1, maka sistem bilangan biner menjadi begitu penting dalam sistem digital. Hampir semua perangkat digital menggunakan sistem bilangan ini untuk menyatakan kuantitas. Jika kita memasukkan bilangan pada perangkat digital, misalnya kalkulator atau komputer, maka kuantitas yang kita masukan dalam format desimal akan dikonversi ke biner. Demikian pula sebaliknya, data biner hasil olahan sistem akan dikonversi ke desimal terlebih dahulu sebelum diinformasikan ke pengguna.

Sistem oktal dan heksadesimal juga banyak digunakan dalam sistem digital. Kedua sistem bilangan tersebut terutama digunakan membentuk tampilan yang lebih efisien. Data yang disajikan dalam format oktal atau heksadesimal akan lebih mudah dibaca daripada data yang ditampilkan dalam format biner.

Data yang ditulis dalam format desimal dapat dikonversi ke format biner, oktal dan heksadesimal. Demikian pula sebaliknya. Bahkan proses konversi dari biner ke oktal atau heksadesimal dan sebaliknya lebih mudah dan cepat daripada konversi dari biner ke desimal.

Pada perangkat digital, keempat sistem bilangan tersebut mungkin tidak akan digunakan sekaligus. Beberapa bagian di bab ini mungkin tidak akan digunakan secara langsung dalam sistem digital. Namun suatu saat jika kita membahas tentang mikroprosesor atau mikrokontroler, mungkin beberapa subbab tersebut perlu dibahas ulang secara lebih mendalam.

Secara umum bilangan dapat dibagi menjadi beberapa katagori. Dari segi penggunaan koma desimal (*point*), bilangan dapat dibagi menjadi dua, yaitu: bilangan bulat (*integer number/fixed-point number*) dan bilangan pecahan (*floating-point number*). Dari segi pemakaian tanda, bilangan dapat dibagi menjadi bilangan tak bertanda (*unsigned number*) dan bilangan bertanda (*signed number*). Pada bab ini, beberapa sistem bilangan tersebut akan dibahas sesuai dengan penggunaannya. Dengan mempelajari beberapa sistem bilangan dalam beberapa katagori diharapkan kita dapat memanipulasi suatu data digital menggunakan sebuah sistem bilangan yang kita anggap paling mudah. Di akhir bab ini akan dijelaskan salah satu bakuan konversi informasi sesuatu yang dimengerti oleh manusia, di antaranya berupa huruf dan kata, ke data yang dimengerti oleh kebanyakan sistem digital. Bakuan dari ASCII tersebut kini diterapkan di hamoir semua sistem komputer komersial.

2.1 Sistem Bilangan Desimal

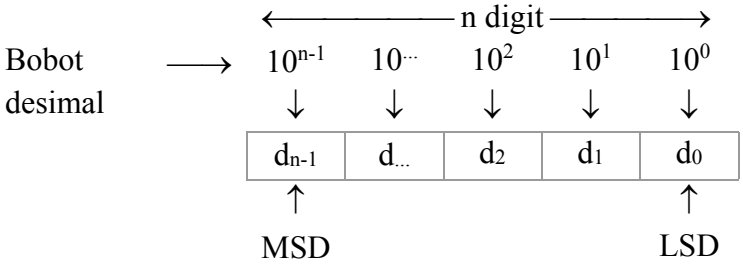
Sistem bilangan ini merupakan sistem bilangan yang paling mudah bagi kita. Sistem bilangan desimal disusun dari 10 angka atau lambang. Dengan menggunakan lambang-lambang tersebut sebagai digit pada sebuah bilangan, kita dapat mengekspresikan suatu kuantitas. Kesepuluh lambang tersebut adalah:

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

Sistem bilangan desimal disebut juga sistem bilangan basis 10 atau radiks 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*.

Untuk membedakan dengan sistem bilangan yang lain, suatu bilangan desimal dapat menggunakan tambahan subskrip *des* atau *10* atau tambahan *D* di akhir suatu bilangan. Contoh: $357_{des} = 357_{10} = 357D$. Namun jika tidak dituliskan bersama dengan sistem bilangan yang lain, subskrip tersebut biasanya dihilangkan karena sistem bilangan desimal sudah menjadi bilangan yang biasa digunakan sehari-hari.

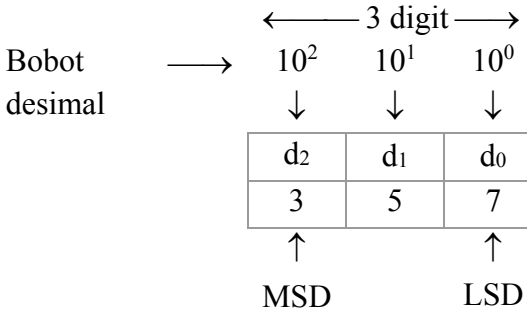
Sistem bilangan desimal merupakan sebuah sistem nilai-posisi. Pada sistem ini, nilai sebuah digit tergantung pada posisinya. Representasi bilangan bulat desimal n digit adalah sebagai berikut.



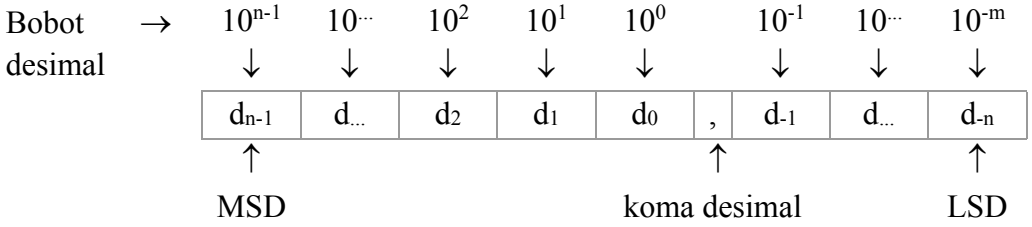
Digit ke-i yang dinyatakan dengan d_i mempunyai bobot desimal 10^i. Digit paling kiri atau d_{n-1} merupakan digit paling berarti karena mempunyai bobot desimal paling besar, yaitu 10^{n-1}. Digit ini disebut digit paling berarti atau MSD (*Most Significant Digit*). Sedangkan digit paling kanan atau d_0 merupakan digit paling tidak berarti karena mempunyai bobot desimal paling kecil, yaitu 10^0=1. Digit ini disebut digit paling tidak berarti atau LSD (*Least Significant Digit*). Suatu bilangan desimal Z dengan panjang n digit akan mempunyai nilai:

$$Z = \sum_{i=0}^{n-1} d_i . 10^i$$

Sebagai contoh adalah bilangan 357. 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 MSD. Sedangkan 7 mempunyai arti paling kecil di antara tiga digit yang ada bertindak sebagai LSD.



Untuk bilangan pecahan desimal, representasi nilainya menjadi sebagai berikut,

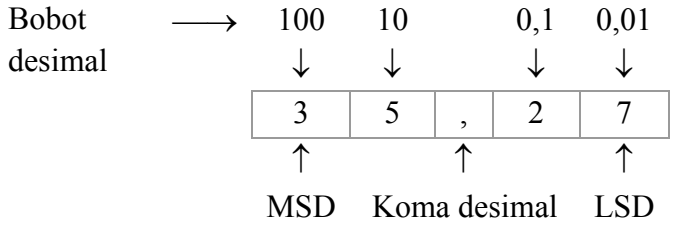


Sehingga suatu bilangan pecahan desimal Z akan mempunyai nilai:

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

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

Contoh lain adalah bilangan 35,27. Bilangan ini mempunyai arti 3 puluhan ditambah 5 satuan ditambah 2 per sepuluhan ditambah 7 per seratusan. Koma desimal memisahkan pangkat positif dari 10 dengan pangkat negatifnya.



35,27 = 3 X 10 + 5 X 1 + 2 X 0,1 + 7 X 0,01

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

2.2 Sistem Bilangan 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 termasuk 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 yaitu Pentium dapat menerima, memproses, menyimpan dan mengirim data atau instruksi dalam format 32 bit.

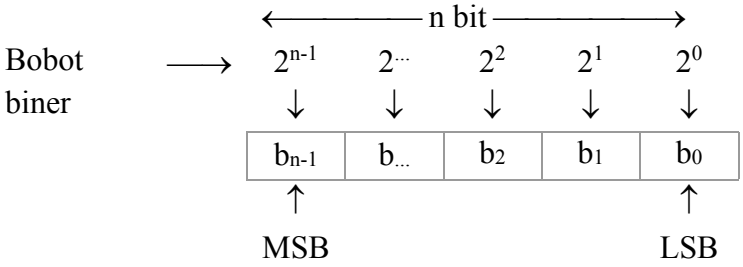
Jika komputer digunakan untuk memproses karakter, maka karakter tersebut harus diformat dalam bentuk kode alfanumerik. Format baku ASCII (*American Standard Code for Information Interchange*) menggunakan format data tujuh bit untuk mewakili semua karakter yang ada termasuk tanda baca dan penanda kontrol.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

Sistem bilangan biner merupakan sistem bilangan basis dua. Seperti telah disebutkan di atas, pada sistem bilangan ini hanya dikenal dua lambang, yaitu:

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

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 bulat biner n bit adalah sebagai berikut,

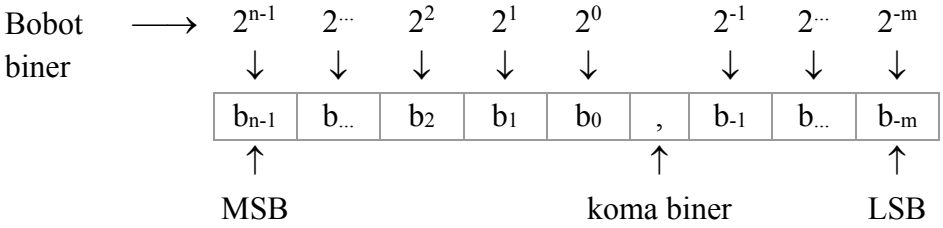


Bit paling kiri dari suatu bilangan biner bertindak sebagai bit paling berarti atau MSB (*Most Significant Bit*). Bit tersebut mempunyai bobot biner 2^{n-1}. Sedangkan bit paling kanan bertindak sebagai bit paling tidak berarti atau LSB (*Least Significant Bit*). Bit tersebut mempunyai bobot biner 2^0=1. Suatu bilangan biner Z dengan panjang n bit akan mempunyai nilai:

$$Z = \sum_{i=0}^{n-1} b_i . 2^i$$

Pada suatu bilangan biner, dengan format bit yang cukup panjang kadang menyulitkan pembacaan. Untuk itu, kadang bit-bit tersebut dikelompokkan empat-empat atau delapan-delapan. Sebagai contoh, bilangan 01010011 dapat ditulis 0101 0011. Empat bit sebelah kiri dinamakan bit-bit tinggi atau nibble tinggi dan empat bit sebelah kanan dinamakan bit-bit rendah atau nibble rendah. Kelompok paling kiri dapat saja mempunyai panjang bit lebih kecil. Sebagai contoh, bilangan 1010011 dapat ditulis 101 0011. Nibble tinggi adalah 101 dan nibble rendah adalah 0011.

Untuk bilangan pecahan biner, representasi nilainya menjadi sebagai berikut,



Sehingga suatu bilangan pecahan biner Z akan mempunyai nilai:

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

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 bobot binernya. Dapat dipastikan bahwa semua bilangan biner, baik bulat maupun pecahan, dapat dikonversi ke bilangan desimal tanpa harus dilakukan pembulatan.

Contoh 5

Konversikan $111{,}0011_{bin}$ ke desimal!

Jawab:

Bobot biner →	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}	2^{-4}
	↓	↓	↓		↓	↓	↓	↓
	1	1	1	,	0	0	1	1

- $111{,}0011_{bin}$= 1 X **4** + 1 X **2** + 1 X **1** + 0 X **0,5** + 0 X **0,25** + 1 X **0,125** + 1 X **0,625**

  = 4+2+1+0+0+0,125+0,0625

  = $7{,}1875_{des}$

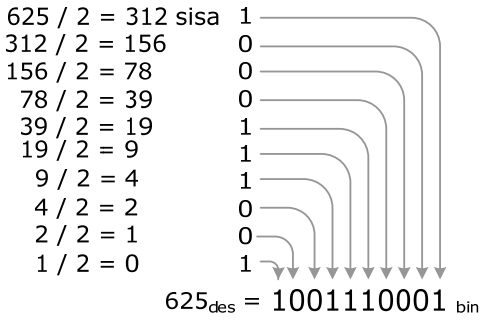
b) Konversi Bilangan Bulat Desimal ke Biner

Konversi bilangan bulat desimal ke biner dilakukan dengan membagi secara berulang-ulang suatu bilangan desimal dengan 2. Sisa setiap pembagian merupakan bit yang didapat. Pembagian diulang terus hingga sisa = 0.

Contoh 6

Konversikan 625_{des} ke biner!

Jawab:



c) Konversi Bilangan Pecahan Desimal ke Biner

Bilangan real desimal dapat dapat pula dikonversi ke bilangan real biner. Konversi dilakukan dengan dua tahap. Tahap pertama, konversikan bagian bulat. Ini dapat dilakukan seperti cara di atas. Tahap kedua, konversikan bagian pecahan. Konversi ini 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 untuk bagian pecahan. Bagian bulat dari setiap hasil perkalian merupakan bit yang didapat. Lihat Contoh 2.4. Perhatikan bahwa urutan mendapatkan hasil untuk konversi bagian pecahan berkebalikan dengan urutan hasil untuk konversi bagian bulat.

Contoh 7

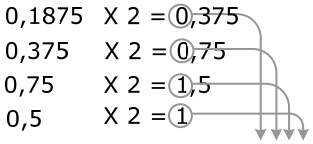
Konversikan $625{,}1875_{des}$!

Jawab:

- Konversi 625_{des} ke biner

$625_{des} = 1001110001_{bin}$ (lihat hasil dari Contoh 6)

- Konversi $0{,}1875_{des}$ ke biner



$0{,}1875_{des} = 0{,}0011_{bin}$

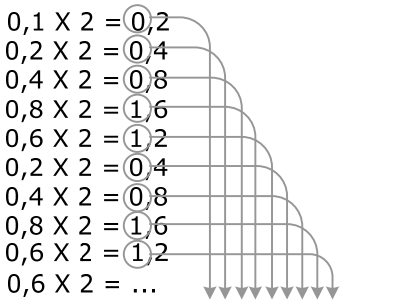
Jadi $625{,}1875_{des} = 1001110001{,}0011_{bin}$

Pada konversi bilangan desimal ke biner, tidak semua bilangan pecahan desimal dapat dikonversi ke pecahan biner. Pecahan desimal yang dapat dikonversi ke pecahan biner harus dapat dinyatakan dengan $\frac{x}{2^y}$ dengan x dan y adalah bilangan bulat positif. Contohnya: $\frac{1}{2}$ =0,5, $\frac{3}{2^3}$ = 0,375 dan $\frac{7}{2^5}$ = 0,21875. Bilangan selainnya dapat dikonversi ke biner namun dengan pembulatan.

Contoh 8

Konversikan $0{,}1_{des}$ ke biner!

Jawab:



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

Jadi $0{,}1_{des} = 0{,}00011001100110011\ldots_{bin}$.

Bilangan $0{,}1_{des}$ tidak dapat dikonversi ke biner dengan tepat.

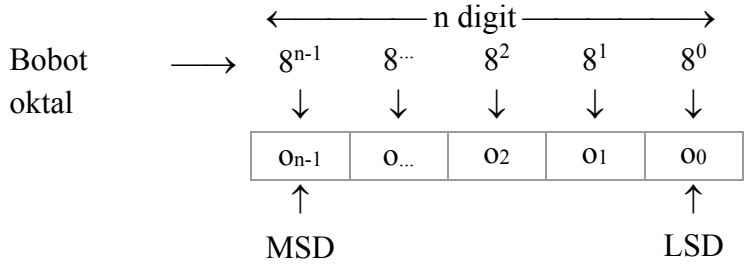
2.3 Sistem Bilangan Oktal

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

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

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

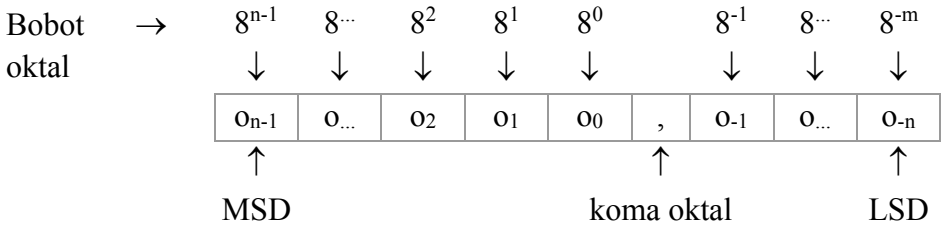
Representasi suatu bilangan bulat oktal n digit hampir sama dengan representasi bilangan bulat desimal dan bulat biner n digit.



Sehingga suatu bilangan bulat oktal Z dengan panjang n digit akan mempunyai nilai:

$$Z = \sum_{i=0}^{n-1} o_i . 8^i$$

Untuk bilangan pecahan oktal, representasi nilainya menjadi sebagai berikut,



Sehingga suatu bilangan pecahan oktal Z akan mempunyai nilai:

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

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

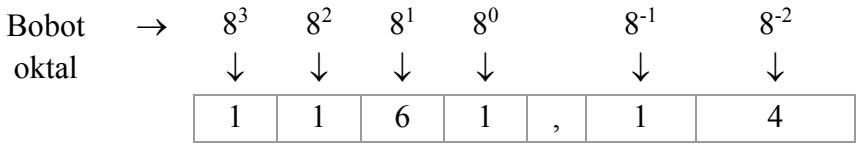
a) Konversi Bilangan Oktal ke Desimal

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

Contoh 9

Konversikan $1161,14_{okt}$ ke desimal!

Jawab:



- 1161_{okt} = 1 X **512** + 1 X **64** + 6 X **8** + 1 X **1** + 1 X **0,125** + 4 X **0,015625**
  = 512+64+48+1+0,125+0,0625
  = $625,1875_{des}$

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

Contoh 10

Konversikan $625{,}1875_{des}$ ke oktal!

Jawab:

Seperti halnya konversi bilangan real desimal ke biner, konversi bilangan real desimal ke oktal harus dilakukan dalam dua tahap. Tahap pertama konversi bagian bulatnya dan tahap kedua bagian pecahannya.

- Konversi 625_{des} ke oktal

```
625 / 8 = 78 sisa 1
78 / 8  = 9       6
9 / 8   = 1       1
1 / 8   = 0       1
```

$625_{des} = 1161_{okt}$

- Konversi $0{,}1875_{des}$ ke oktal

```
0,1875  X 8 = 1,5
0,5     X 8 = 4
```

$0{,}1875_{des} = 0{,}14_{okt}$

Jadi $625{,}1875_{des} = 1161{,}14_{okt}$

Pada konversi bilangan pecahan desimal ke oktal, tidak semua bilangan pecahan desimal dapat dikonversi ke pecahan oktal. Pecahan desimal yang dapat dikonversi ke pecahan biner harus dapat dinyatakan dengan $\frac{x}{8^y}$ dengan x dan y adalah bilangan bulat positif. Contohnya: $\frac{4}{8^1}$ =0,5, $\frac{3}{8^1}$ = 0,375 dan $\frac{14}{8^2}$ = 0,21875. Jika suatu bilangan pecahan desimal dapat dikonversi ke bilangan biner dengan tepat, maka bilangan itu pasti juga dapat dikonversi ke bilangan oktal dengan tepat. Bilangan selainnya dapat dikonversi ke oktal namun dengan pembulatan. Silakan dicoba konversikan $0{,}1_{des}$ ke oktal! Hasilnya tidak akan tepat karena pada konversi ke biner di Contoh 8 juga mendapatkan hasil tidak tepat.

c) Konversi Bilangan Oktal ke Biner

Konversi bilangan oktal ke biner lebih mudah dibandingkan dengan konversi bilangan oktal ke desimal. Konversi dilakukan per digit oktal. Satu digit oktal dikonversi ke 3 bit biner. Tabel 2.1 di halaman 28 dapat digunakan untuk membantu proses pengonversian ini.

Contoh 11

Konversikan $1161{,}14_{okt}$ ke biner!

Jawab:

oktal ⟶	1	1	6	1	,	1	4
	↓	↓	↓	↓		↓	↓
biner ⟶	001	001	110	001	,	001	100

Jadi $1161{,}14_{okt} = 1001110001{,}0011_{bin}$.

Seperti halnya pada bilangan desimal, angka nol paling kiri untuk bagian bulat dapat dihapus tanpa mengubah nilai, demikian pula angka nol paling kanan untuk bagian pecahan.

d) Konversi Bilangan Biner ke Oktal

Konversi bilangan biner ke oktal juga lebih mudah dilakukan dibandingkan konversi bilangan desimal ke oktal. Konversi dilakukan per tiga bit biner. Tiga bit biner menjadi satu digit 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 12

Konversikan $1001110001{,}0011_{bin}$ ke oktal!

Jawab:

biner ⟶	001	001	110	001	,	001	100
	↓	↓	↓	↓		↓	↓
oktal ⟶	1	1	6	1	,	1	4

Jadi $1001110001{,}011100_{bin} = 1161{,}14_{okt}$.

Seperti halnya pada bilangan desimal, sutau bilangan biner dapat ditambah angka nol di sebelah kiri MSB untuk bagian bulat dan angka nol di sebelah kanan LSB untuk bagian pecahan tanpa mengubah nilai.

2.4 Sistem Bilangan Heksadesimal

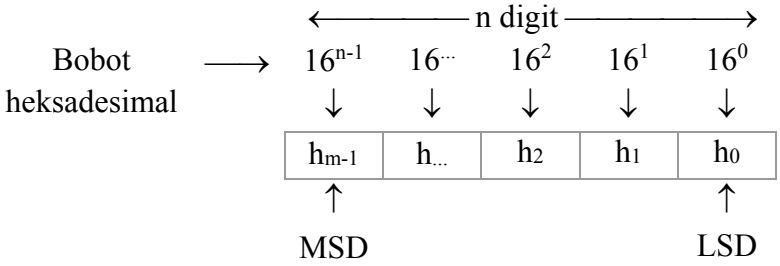
Sistem bilangan heksadesimal merupakan sistem bilangan basis enam belas. Meskipun pada sistem digital dan komputer operasi secara fisik dikerjakan secara biner, namun untuk representasi data banyak digunakan format bilangan heksadesimal. Hal ini disebabkan format heksadesimal lebih praktis, mudah dibaca 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:

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

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

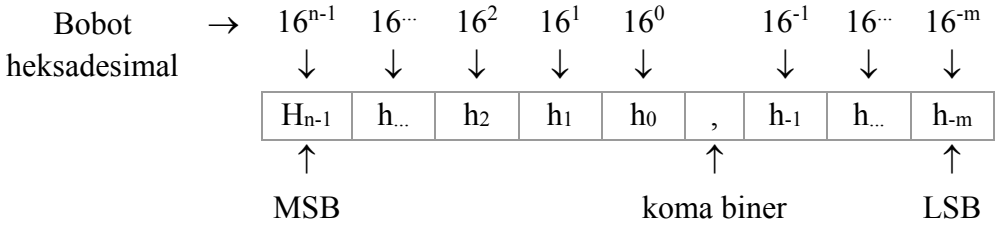
Representasi suatu bilangan bulat heksadesimal hampir sama dengan representasi bilangan bulat desimal dan bilangan bulat biner.



Sehingga suatu bilangan heksadesimal Z dengan panjang n digit akan mempunyai nilai:

$$Z = \sum_{i=0}^{n-1} h_i . 16^i$$

Untuk bilangan pecahan heksadesimal, representasi nilainya menjadi sebagai berikut,



Sehingga suatu bilangan pecahan heksadesimal Z akan mempunyai nilai:

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

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

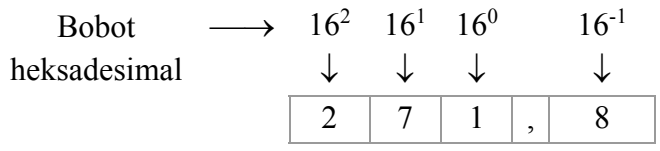
a) Konversi Bilangan Heksadesimal ke Desimal

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

Contoh 13

Konversikan $271,3_{heks}$ ke desimal!

Jawab:



$271,8_{heks}$ = 2 X **256** + 7 X **16** + 1 X **1** + 3 X **0,0625**

= 512+112+1+0,125+0,1875

= $625,1875_{des}$

b) Konversi Bilangan Desimal ke Heksadesimal

Konversi bilangan bulat desimal ke heksadesimal dilakukan dengan membagi secara berulang-ulang suatu bilangan desimal dengan 16. Sisa setiap pembagian merupakan digit heksadesimal yang didapat. Sedangkan konversi bilangan pecahan desimal ke heksadesimal dilakukan dengan cara hampir sama dengan konversi bilangan pecahan desimal ke biner, yaitu dengan mengalikan suatu bilangan pecahan desimal dengan 16. Bagian pecahan dari hasil perkalian ini dikalikan dengan 16. Langkah ini diulang hingga didapat hasil akhir 0 untuk bagian pecahannya. Bagian bulat dari setiap hasil perkalian merupakan digit yang didapat.

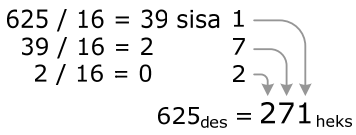
Contoh 14

Konversikan $625{,}1875_{des}$ ke heksadesimal!

Jawab:

Seperti halnya konversi bilangan pecahan desimal ke biner, konversi bilangan real desimal ke heksadesimal harus dilakukan dalam dua tahap. Tahap pertama adalah konversi bagian bulatnya dan tahap kedua bagian pecahannya.

- Konversi 625_{des} ke heksadesimal



- Konversi $0{,}1875_{des}$ ke heksadesimal

  $0{,}1875 \times 16 = 7$

  Jadi $625{,}1875_{des} = 271{,}7_{heks}$

Pada konversi bilangan desimal ke heksadesimal, tidak semua bilangan pecahan desimal dapat dikonversi ke pecahan heksadesimal. Pecahan desimal yang dapat dikonversi ke pecahan biner harus dapat dinyatakan dengan $\frac{n}{16^m}$ dengan n dan m adalah bilangan bulat. Contohnya: $\frac{8}{16}$=0,5, $\frac{6}{16}$= 0,375 dan $\frac{56}{16^2}$= 0,21875. Jika suatu bilangan pecahan desimal dapat dikonversi ke bilangan biner dan oktal dengan tepat, maka bilangan itu pasti juga dapat dikonversi ke bilangan heksadesimal dengan tepat. Demikian pula sebaliknya. Bilangan selainnya dapat dikonversi ke heksadesimal namun dengan pembulatan. Silakan dicoba konversikan $0{,}1_{des}$ ke heksadesimal!

c) 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 biner. Tabel 2.1 di halaman 28 dapat digunakan untuk membantu proses pengonversian ini.

Contoh 15

Konversikan $271,7_{heks}$ ke biner!

Jawab:

heksadesimal ⟶	2	7	1	,	7
	↓	↓	↓		↓
biner ⟶	0010	0111	0001	,	0111

Jadi $271,7_{heks} = 1001110001,0111_{bin}$

d) Konversi Bilangan Biner ke Heksadesimal

Konversi bilangan biner ke heksadesimal lebih mudah dibandingkan konversi bilangan desimal ke heksadesimal. Empat bit biner menjadi satu digit 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 16

Konversikan $1001110001,0111_{bin}$ ke heksadesimal!

Jawab:

biner ⟶	0010	0111	0001	,	0111
	↓	↓	↓		↓
heksadesimal ⟶	2	7	1	,	7

Jadi $1001110001,0111_{bin} = 271,7_{heks}$.

2.5 Sistem Bilangan 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. Sehingga setiap digit desimal dikodekan ke 4 bit biner. Sistem bilangan BCD biasanya digunakan untuk keperluan penampil 7-segmen (*seven-segment display*) dan ini akan dibahas di Bab 9.

Contoh 17

Konversikan 271 ke BCD!

Jawab:

desimal ⟶	2	7	1
	↓	↓	↓
BCD ⟶	0010	0111	0001

Jadi $271_{des} = 0010\ 0111\ 0001_{BCD}$.

Berikut tabel konversi bilangan desimal dari 0 sampai 15 ke bilangan biner, oktal, heksadesimal dan BCD. Setelah selesai mempelajari subbab ini, seharusnya kita dapat membuat tabel tersebut untuk bilangan di atas 15!

Tabel 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
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

Meskipun pada hakekatnya bit-bit bilangan BCD sama dengan bit-bit bilangan biner, perlu diperhatikan di sini, pada sistem bilangan BCD, 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. Keempat nilai tersebut tentu saja tidak ada pada Tabel 2.1. Dalam perancangan perangkat konversi BCD ke desimal, yang akan dibicarakan dalam Bab 7, keempat bilangan biner tersebut masuk dalam keadaan diabaikan (*don't care*).

2.6 Sistem Bilangan Biner Tak Bertanda dan Bertanda

Seperti telah diuraikan di muka, bilangan dapat dikatagorikan menjadi bilangan takbertanda dan bertanda. Demikian pula untuk sistem bilangan biner. Terdapat dua jenis bilangan biner, yaitu bilangan biner tak bertanda dan bilangan biner bertanda. Pada sistem bilangan biner tak bertanda, hanya dikenal bilangan biner posisif dan tidak diijinkan adanya bilangan biner negatif. Di sini semua bit digunakan untuk merepresentasikan suatu nilai.

Sebagai contoh:

- Bilangan biner 4 bit 1100.

b_3	b_2	b_1	b_0
1	1	0	0

Pada bilangan biner tak bertanda di atas, nilai bilangan dihitung dari b_3, b_2, b_1 hingga b_0. Sehingga nilai desimalnya,

$1100_{bin} = 1 \text{ X } 2^3 + 1 \text{ X } 2^2 + 0 \text{ X } 2^1 + 0 \text{ X } 2^0$

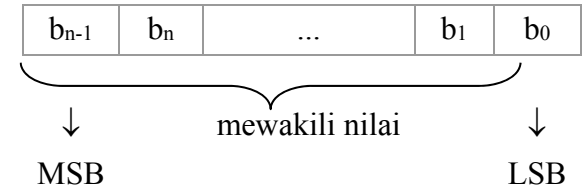
$= 12_{des}$

Pada bilangan biner bertanda, bit paling kiri, yaitu b_3, menyatakan tanda positif atau negatif suatu bilangan biner. Bit ini sering disebut bit tanda (*sign bit*). Nilai bilangan dihitung dari b_2, b_1 hingga b_0.

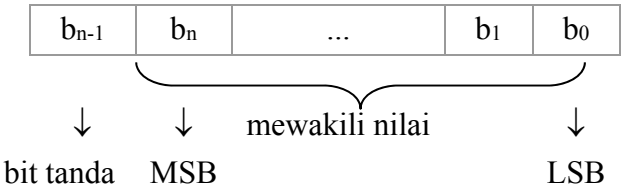
$0100_{bin} = + (1 \text{ X } 2^2 + 0 \text{ X } 2^1 + 0 \text{ X } 2^0)$

$= 4_{des}$

Pada sistem bilangan biner bertanda, karena bit paling kiri merupakan bit tanda, maka MSB terletak di sebelah kanan bit tanda. Sehingga pada bilangan biner bertanda n-bit, hanya n-1 bit saja yang mewakili nilai. Satu bit paling kiri merupakan bit tanda. Tanda positif biasanya diwakili oleh bit 0 dan tanda negatif diwakili oleh bit 1.



(a) bilangan tak bertanda



(b) bilangan bertanda

Gambar 2.1 Format bilangan biner tek bertanda dan bertanda

Misalnya, suatu memori dapat menampung 7 bit bilangan biner. Memori tersebut mengunakan sistem bilangan biner bertanda. Maka dari ketujuh bit yang ada, bit paling kiri, yaitu b_6, digunakan sebagai bit tanda, sedangkan bit-bit yang lain, yaitu bit b_5 ... b_0 mewakili suatu nilai. Sehingga untuk membedakan nilai $+52_{des}$ dan -52_{des} dapat dilihat bit paling kirinya.

- Bilangan biner 0110100

b_6	b_5	b_4	b_3	b_2	b_1	b_0	
0	1	1	0	1	0	0	$= +52_{des}$
↑							
Bit tanda (+)	**52_{des}**						

Bilangan ini merupakan bilangan biner positif karena $A_6 = 0$, dengan nilai $+52_{des}$.

- Bilangan biner 1110100

b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	1	1	0	1	0	0	$= -52_{des}$
↑	⌣				⌣		
Bit tanda (-)	**52_{des}**						

Bilangan ini adalah negatif karena $A_6 = 1$ dengan nilai -52.

Pada metode di atas, pengubahan bilangan positif ke negatuf dapat dilakukan dengan menjadikan bit paling kiri sama dengan 1. Sementara itu bit yang lain tidak berubah. Metode di atas tidak banyak digunakan kecuali untuk keperluan khusus, karena bilangan biner dengan format tersebut tidak dapat dikenai perhitungan matematis. Coba saja tambahkan -52_{des} dengan 52_{des} atau perhitungan yang lain dalam format biner! Hasilnya pasti salah. Untuk itu harus digunakan metode lain agar bilangan biner positif maupun negatif dapat dikenai perhitungan matematis.

a) Bilangan Biner Komplemen Satu

Ada metode lain untuk mengonversi bilangan positif ke negatif, yaitu dengan metode komplemen. Terdapat dua sistem komplemen, yaitu sistem bilangan biner komplemen satu dan sistem bilangan biner komplemen dua. Metode 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. Atau dengan kata lain, penegatifan dapat dilakukan dengan pembalikan bit 0 dan 1-nya.

Sebagai contoh, 0101101 merupakan bilangan biner 7 bit dengan nilai 45. Maka -45 sama dengan 1010010.

0	1	0	1	1	0	1	→ 45
↓	↓	↓	↓	↓	↓	↓	(dibalik 0 dan 1-nya)
1	0	1	0	0	1	0	→ -45

Jika P merupakan suatu bilangan positif, bilangan komplemen satu n bit –P juga dapat diperoleh dengan mengurangkan P dari 2^n–1. Atau, bilangan komplemen satunya menjadi $(2^n–1) – P$. Contohnya adalah jika $P = 45_{des} = 0101101_{bin}$

Dalam komplemen satu, $–P = (2^7 – 1) – 45$

$= 1111111 – 0101101$

$= 1010010$

Sistem bilangan komplemen satu jarang digunakan karena tidak memenuhi satu kaedah matematis, yaitu jika suatu bilangan dijumlahkan dengan negatifnya, maka akan dihasilkan bilangan nol. Lihat proses penjumlahan berikut.

	0	1	0	1	1	0	1
+	1	0	1	0	0	1	0
	1	1	1	1	1	1	1

Pada contoh tersebut, 0101101 + 1010010 = 1111111, sehingga $45 + (-45) \neq 0$.

Karena terdapat kesalahan untuk operasi matematis atas bilangan negatif, maka sistem komplemen satu hanya digunakan untuk mewakili bilangan positif atau bilangan

tak bertanda (*unsigned number*) saja. Untuk mewakili bilangan negatif dibentuklah sistem bilangan biner baru, yaitu sistem bilangan biner komplemen dua.

b) Bilangan Biner Komplemen Dua

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

negatif komplemen dua = negatif komplemen satu + 1

Sebagai contoh, 0101101 merupakan bilangan biner 7 bit dengan nilai 45. Maka -45 sama dengan 1010011.

```
  0 1 0 1 1 0 1    → 45
  ↓ ↓ ↓ ↓ ↓ ↓ ↓    (dibalik 0 dan 1-nya sebagaimana komplemen satu)
  1 0 1 0 0 1 0    → -45 dalam komplemen satu
+                1
-----------------
  1 0 1 0 0 1 1    → -45 dalam 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:

```
  1 0 1 0 0 1 1    → -45
—                1
-----------------
  1 0 1 0 0 1 0    → -45 dalam komplemen satu
  ↓ ↓ ↓ ↓ ↓ ↓ ↓    (dibalik 0 dan 1-nya)
  0 1 0 1 1 0 1    → 45
```

Jika P merupakan suatu bilangan positif, bilangan komplemen dua n bit –P juga dapat diperoleh dengan mengurangkan P dari 2^n. Atau, bilangan komplemen duanya menjadi $2^n – P$. Contohnya adalah jika $P = 45_{des} = 0101101_{bin}$

Dalam komplemen dua $–P = 2^7 – 45$

$= 1000000 – 010011$

$= 101101$

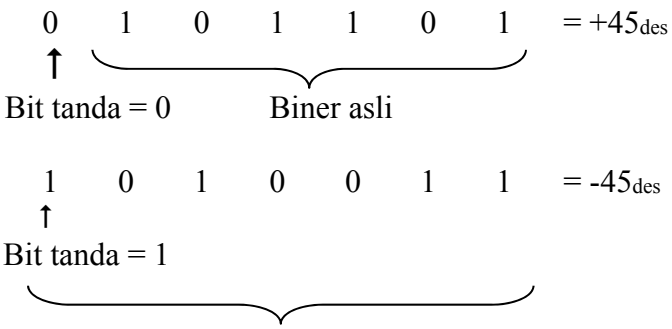
Sistem bilangan biner komplemen dua banyak digunakan dalam sistem digital dan komputer karena memenuhi kaedah matematis, yaitu jika suatu bilangan dijumlahkan dengan negatifnya, maka akan dihasilkan bilangan nol.

```
  0 1 0 1 1 0 1
+ 1 0 1 0 0 1 1
---------------
1 0 0 0 0 0 0 0    1*)
↳ bawaan 1 tidak digunakan
```

Pada penjumlahan tersebut, bit 1 paling depan merupakaan bit bawaan (*carry*) dan tidak digunakan. Jadi 0101101 + 1010011 = 0000000, sehingga 45 + (-)45 = 0.

*) Lebih jauh mengenai proses penjumlahan akan dijelaskan dalam subbab berikutnya.

Pada suatu bilangan biner komlemen 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-bit tersebut merupakan bentuk bilangan biner komplemen dua.



Contoh 18

- -7_{des} = ... bin
- 0101_{bin} = ... des
- 1011_{bin} = ... des

Jawab:

- $7_{des} = 0111_{bin}$

  Mencari -7_{des} adalah dengan membalik 0111 dan menambah dengan 1.

  -7_{des} = 1000
  1+
  1001
- Untuk bilangan 0101_{bin}, karena bit tanda sama dengan nol, maka bilangan tersebut merupakan bilangan positif, sehingga tiga bit di belekangnya merupakan bilangan biner asli. $0101_{bin} = 5_{des}$.
- Untuk bilangan 1011_{bin}, karena bit tanda sama dengan 1, maka bilangan tersebut merupakan bentuk bilangan komplemen dua. Langkah mengonversi 1011_{bin} ke desimal adalah dengan menguranginya dengan 1 dan membalik bit-bitnya.

  1011
  1 -
  1010
  ↓ dibalik 0 dan 1-nya

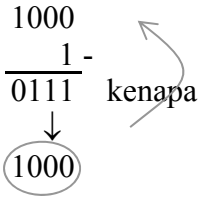
  $0101_{bin} = 5_{des}$.

  Karena $0101_{bin} = 5_{des}$, maka $1011_{bin} = -5_{des}$.

Contoh 19

Berikut merupakan bilangan biner 4 bit, 1000_{bin}. Konversikan ke desimal!

Jawab:



kenapa kembali ke 1000?

Bilangan 1000_{bin} tidak dapat dikonversi ke desimal dengan cara di atas!

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-1}, dengan n merupakan panjang bit yang digunakan. Contohnya:

- 10_{bin} (panjang 2 bit) = $-2^{2-1} = -2_{des}$.
- 1000_{bin} (panjang 4 bit) = $-2^{4-1} = -8_{des}$.
- 10000000_{bin} (panjang 8 bit) = $-2^{8-1} = -128_{des}$.

Bilangan biner n-bit dapat mewakili 2^n nilai. Namun perlu diperhatikan, jika digunakan sistem bilangan biner komplemen satu atau dua, maka banyaknya 2^n nilai tersebut diperuntukkan untuk mewakili nilai positif dan negatif. Batasan bilangan biner n-bit dapat dilihat pada Tabel 2.2.

Tabel 2.2 Batasan jangkauan bilangan biner n-bit

Bilangan biner n-bit	Minimal	Maksimal
Tak bertanda	0	2^n-1
Komplemen satu	$-2^{n-1}+1$	$2^{n-1}-1$
Komplemen dua	-2^{n-1}	$2^{n-1}-1$

Contoh 20

Berikan semua variasi nilai bilangan biner 4 bit untuk bilangan biner tak bertanda, bilangan biner komplemen satu dan bilangan biner komplemen dua! Bilangan desimal berapa saja yang dapat diwakili oleh bilangan biner tersebut?

Jawab:

Semua variasi bilangan biner 4 bit dan bilangan desimal yang diwakili dapat dilihat pada Tabel 2.3. Sesuai dengan Tabel 2.2, bilangan biner tak bertanda 4 bit dapat mempunyai 16 variasi nilai, yaitu dari 0 hingga $2^4-1 = 15$. Bilangan biner komplemen satu 4 bit dapat mempunyai 15 variasi nilai, yaitu dari $-2^{4-1}+1 = -7$ hingga $2^{4-1}-1 = 7$. Sedangkan bilangan biner komplemen dua 4 bit dapat mempunyai 16 variasi nilai, yaitu dari $-2^{4-1} = -8$ hingga $2^{4-1}-1 = 7$.

Tabel 2.3 Nilai yang dapat diwakili bilangan biner komplemen dua 4 bit

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

Dikarenakan sistem bilangan komplemen dua memenuhi kaedah matematis, maka banyak perangkat aplikasi yang membutuhkan operasi aritmatika dan logika menggunakan sistem bilangan ini. Pada perangkat tersebut biasanya terdapat sebuah untai khusus yang digunakan untuk melakukan perhitungan aritmatika dan logika. Untai tersebut dinamakan untai aritmatika dan logika (*Arithmatic and Logic Unit*, ALU). Misalnya di kalkulator, mikroprosesor dan mikrokontroler. Prinsip kerja ALU secara sederhana akan dibahas di Bab 5.

c) Format Panjang Bilangan Biner

Bilangan biner terutama komplemen dua 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. Penambahan panjang bilangan biner 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. Penambahan panjang bilangan biner 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

Ketentuan nomor dua di atas mungkin agak aneh. Namun silakan dibuktikan bahwa $1100_{bin} = 1\ 1100_{bin} = 11\ 1100_{bin} = 1111\ 1100_{bin} = 4_{des}$. Perlu diingat pada contoh di atas bahwa bit paling depan merupakan bit tanda, sehingga pada format 4 bit hanya ada 3 bit yang mewakili suatu nilai, dan pada format 8 bit hanya ada 7 bit yang mewakili nilai.

Contoh 21

Pada Gambar 1.3, sebuah isyarat analog dengan kisaran 0 hingga 1000 mV dikonversi ke isyarat digital dengan tegangan logika.

- Berapa bit yang dibutuhkan untuk menyatakan informasi setiap tegangan logika?
- Dengan tetap menggunakan empat tegangan logika, bagaimana jika digunakan format 4 bit untuk menyatakan posisi setiap tegangan logika?

Jawab:

- Karena hanya ada empat posisi, maka hanya dibutuhkan 2 bit untuk setiap tegangan logika. Keempat posisi tersebut dapat dinyatakan dengan 00, 01, 10 dan 11.
- Jika digunakan format 4 bit untuk menyatakan setiap tegangan logika, maka cukup tambahkan dua nol di depan karena semua nilainya adalah bilangan positif.

2.7 Sistem Bilangan Biner Baku

Dalam sistem komputer dan bahasa pemrograman, bilangan dapat menggunakan dua format, yaitu format normal dan format eksponen. Pada format normal, suatu nilai dinyatakan seperti apa adanya; sedangkan pada format eksponen, suatu nilai dinyatakan dalam bentuk eksponen. Suatu nilai yang dinyatakan dengan format normal $\pm D_1D_0,D_{-1}D_{-2}D_{-3}$ dapat diubah menjadi format eksponen $\pm D_0,D_{-1}D_{-2}D_{-3}$ X 10^E dan sebaliknya. Sebagai contoh, bilangan berformat normal -0,123456 dapat dinyatakan dengan format eksponen -1,23456 X 10^{-1}.

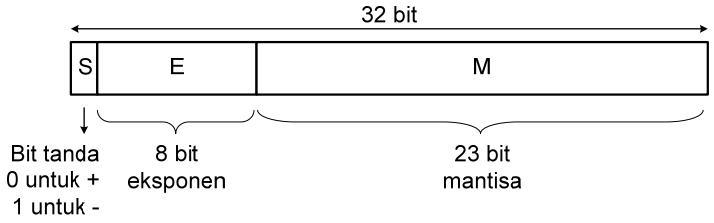
Bagian bulat suatu bilangan dinamakan digit, bagian pecahan suatu bilangan dinamakan mantisa, sedangkan pangkat 10-nya dinamakan eksponen. Pada contoh di atas ‘1’ merupakan digit, ‘23456’ merupakan mantisa, dan ‘-1’ merupakan eksponen.

Sistem bilangan biner pada pengolah data umumnya menggunakan sistem bilangan biner dengan format yang telah dibakukan oleh IEEE (*Institute of Electrical and Electronic Engineers*). Bilangan biner baku ini dapat mewakili bilangan pecahan positif maupun negatif. Dua format yang banyak digunakan dalam banyak aplikasi, terutama

untuk bahasa pemrograman Basic, C maupun Pascal, adalah bilangan biner presisi tunggal berformat 32 bit dan bilangan biner presisi ganda berformat 64 bit.

a) Format Presisi Tunggal

Pada format ini, sebuah nilai diwakili oleh sebuah bilangan biner 32 bit. Bit paling kiri merupakan bit tanda (*sign*, S), delapan bit sesudah bit tanda merupakan bit eksponen (E), dan 23 bit berikutnya menyatakan mantisa (M).



Gambar 2.2 Format biner presisi tunggal

Bit tanda 0 untuk menunjukkan bahwa nilai yang diwakili adalah bilangan positif, dan bit tanda 1 menunjukkan bahwa nilai yang diwakkili adalah bilangan negatif. Bit eksponen mewakili nilai eksponen dari dua ($2^{Eksponen}$). Nilai eksponen dapat positif maupun negatif. Dengan panjang 8 bit untuk eksponen dapat mewakili nilai eksponen dari -128 hingga 127. Di sini nilai E dikurangi 127 atau 10000000_{bin} untuk mendapatkan nilai eksponen, sehingga nilai E berkisar dari 0 hingga 255.

Eksponen = E - 127

Sebagai contoh, jika $E=1010000_{bin}$ (80_{des}), maka nilai eksponen adalah 80_{des}-127_{des}= -47_{des}. Nilai eksponen akan menjadi penting dan harus diketahui terlebih dahulu sebelum dilakukan proses penjumlahan dan pengurangan, karena syarat dua buah bilangan dapat dijumlahkan atau dikurangkan adalah adanya kesamaan posisi koma yang menjadi pemisah antara digit dan mantisa.

Nilai paling kecil E, yaitu 0, digunakan untuk mewakili nilai mendekati nol. Sedangkan nilai paling besar E, yaitu 255, mewakili nilai tak terhingga (∞). Sehingga nilai E normal adalah dari 1 hingga 254 untuk mewakili nilai eksponen -126 hingga 127.

Pada format presisi tunggal, mantisa dinyatakan dalam 23 bit. Sedangkan MSB dinormalisasi menjadi 1. Dan bit ini tidak perlu dituliskan. Sehingga untuk bilangan yang dinyatakan pada Gambar 2.2,

Nilai = $\pm 1,M \times 2^{E-127}$

Contoh 22

Berapa nilai bilangan yang diwakili oleh 1 10100000 11000000000000000000000?

Jawab:

Bit tanda bilangan tersebut adalah 1 yang berarti bilangan negatif, mantisa bilangan tersebut adalah 11000000000000000000000. Sedangkan eksponennya adalah 10100000-10000000 = 00100000. Sehingga nilai bilangan tersebut dalam biner adalah

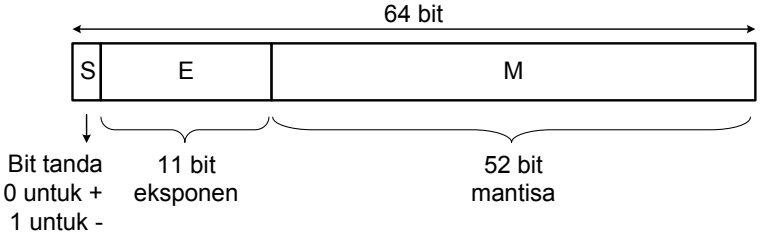
-1, $11000000000000000000000 \times 10^{00100000}$

Format presisi tunggal mempunyai 23 bit untuk mantisa, sehingga jika dikonversi ke desimal akan memiliki ketepatan hingga 7 digit. Sedangkan eksponen binernya akan

mempunyai jangkauan dari 2^{-126} sampai 2^{127} atau jika dikonversi ke desimal dapat mewakili nilai dari 10^{-38} hingga 10^{+38}.

b) Format Presisi Ganda

Format ini menggunakan 64 bit untuk menyatakan suatu nilai. Panjang eksponen diperbesar menjadi 11 bit dan mantisa menjadi 52 bit.



Gambar 2.3 Format biner presisi ganda

Jangkauan E adalah dari 0 hingga 2047. Untuk mendapatkan eksponen, E harus dikurangi 1023. Nilai E = 0 digunakan untuk menunjukkan nilai mendekati nol, dan nilai E = 2047 untuk menunjukkan nilai tak terhingga. Nilai E normal adalah 1 hingga 2046 yang mewakili nilai eksponen dari -1022 hingga 1023.

Mantisa dinyatakan oleh 52 bit. Seperti pada format baku presisi tunggal, MSB dinormalisasi menjadi 1. Dan bit ini tidak perlu dituliskan. Sehingga untuk bilangan yang dinyatakan pada Gambar 2.3,

$$\textbf{Nilai} = \pm 1{,}M \times 2^{E-1023}$$

Dengan 52 bit mantisa dapat mewakili sekitar 16 digit untuk mantisa desimal. Sedangkan jangkauan nilai eksponen biner dari 2^{-1022} sampai 2^{1023} dapat mewakili nilai desimal dari 10^{-308} hingga 10^{+308}.

2.8 Sistem Bilangan Oktal dan Heksadesimal Bertanda

Sebagaimana bilangan biner, pada sistem bilangan oktal dan heksadesimal pun dapat digunakan untuk mewakili bilangan negatif. Perhatikan Tabel 2.4! Seperti telah dijelaskan pada subbab 2.3.d), konversi dari bilangan biner ke oktal dilakukan dengan mengelompokkan bilangan biner tiga-tiga dari paling kanan. Jika kita ambil sebuah kelompok tiga-bit-biner, maka dengan format tiga bit biner, akan kita dapatkan konversi ke oktal seperti Tabel 2.4 di atas. Sehingga bilangan oktal 1 ~ 3 mewakili nilai oktal positif dan 4 ~ 7 mewakili nilai oktal negatif. Bilangan oktal bertanda yang mempunyai MSD 1 ~ 3 merupakan merupakan oktal positif dan bilangan oktal yang mempunyai MSD 4 ~ 7 merupakan merupakan oktal negatif. Sistem bilangan oktal bertanda yang digunakan adalah sistem oktal komplemen dua.

Tabel 2.4 Nilai yang dapat diwakili bilangan oktal komplemen dua satu digit

Desimal	Komplemen dua	
	biner	oktal
+3	011	3
+2	010	2
+1	001	1
0	000	0
-1	111	7
-2	110	6
-3	101	5
-4	100	4

Contoh 23

Ubahlah hasil di Contoh 18 menjadi oktal!

Jawab:

- $-7_{des} = 1001_{bin} = 111\ 001_{bin}$ (penambahan bit 1 di depan biner negatif tidak mengubah nilai)

111	001
↓	↓
7	1

  Dengan demikian $-7_{des} = 71_{okt}$
- $0101_{bin} = 5_{okt}$.
- $1011_{bin} = 111\ 011_{des} = 73_{okt}$

Sistem bilangan heksadesimal bertanda juga menggunakan sistem heksadesimal komplemen dua. Lihat Tabel 2.5! Bilangan heksadesimal bertanda yang mempunyai MSD 1 ~ 7 merupakan merupakan heksadesimal positif dan bilangan heksadesimal yang mempunyai MSD 8 ~ F merupakan merupakan heksadesimal negatif.

Tabel 2.5 Nilai yang dapat diwakili bilangan heksadesimal komplemen dua satu digit

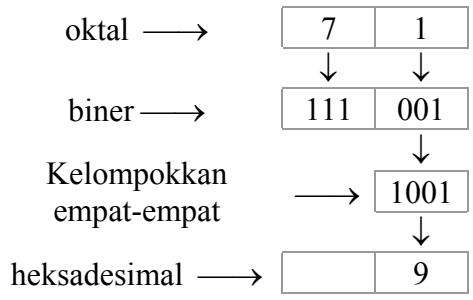
Desimal	Komplemen dua	
	biner	heksadesimal
+7	0111	7
+6	0110	6
+5	0101	5
+4	0100	4
+3	0011	3
+2	0010	2
+1	0001	1
0	0000	0
-1	1111	F
-2	1110	E
-3	1101	D
-4	1100	C
-5	1011	B
-6	1010	A
-7	1001	9
-8	1000	8

Contoh 24

Ubahlah hasil di Contoh 23 menjadi heksadesimal!

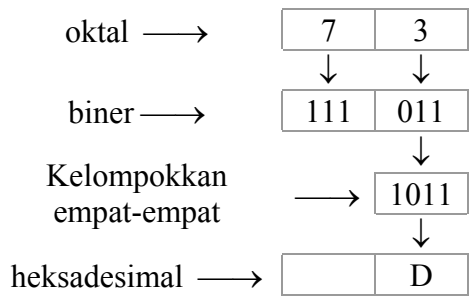
Jawab:

- 71_{okt} = ... heks



Dengan demikian $71_{okt} = 9_{heks}$

- $5_{okt} = 5_{heks}$
- 73_{okt} = ... heks



Perhatikan bahwa penambahan bilangan 7 di depan bilangan oktal negatif dan penambahan bilangan F di depan bilangan heksadesimal negatif tidak akan mengubah nilai. Cermati Contoh 25 berikut.

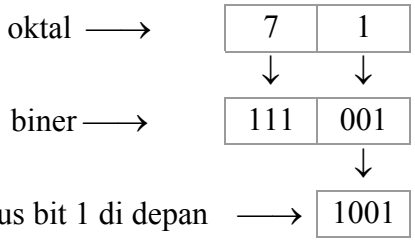
Contoh 25

Konversikan:

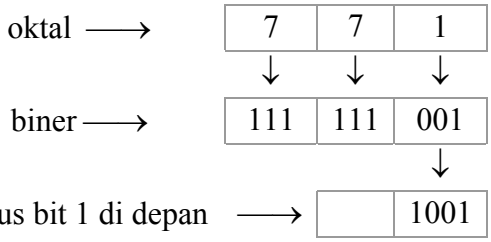
- 71_{okt} dan 771_{okt} ke biner.

Jawab:

- 71_{okt} = ... bin



- 771_{okt} = ... bin



Ternyata $771_{okt} = 71_{okt} = 1001_{bin.}$

2.9 Aritmatika Biner

7. Pada bilangan biner, baik bulat maupun pecahan, dapat dikenakan operasi aritmatika. Demikian pula untuk bilangan biner positif dan negatif. Di subbab ini akan dijelaskan bagaimana operasi aritmatika dapat dilakukan pada bilangan biner. Subbab ini akan diakhiri bahasan penjumlahan pada bilangan BCD.

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). Hal ini berlaku untuk penjumlahan bilangan bulat biner maupun pecahan. Pada operasi penjumlahan, hasil penjumahan akan mempunyai panjang bit minimal sama dengan bilangan yang dijumlahkan. Contoh:

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

Pada penjumlahan bilangan biner komplemen dua terdapat beberapa kasus yaitu:

Penjumlahan bilangan positif

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

```
  01001  (9)
+ 00100  (4)
  01101 (13)
```

↳ 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)
1 00101  (5)
```

↳ bawaan 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)
  11011 (-5)
```

↳ bit tanda

Perlu diingat, 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)
1 10011(-13)
```

↳bawaan 1 tidak digunakan

Penjumlahan bilangan positif dan negatifnya

Penjumlahan ini akan menghasilkan nol. Contoh:

```
  01001  (9)
+ 10111 (-9)
1 00000  (0)
```

↳bawaan 1 tidak digunakan

Pada operasi penjumlahan bilangan biner komplemen dua n bit, harus diperhatikan bahwa hasil yang didapat harus masih dalam jangkauan nilai -2^{n-1} hingga $2^{n-1}-1$. Jika hasil di luar batas tersebut, maka hasil perhitungan sudah di luar batas dan dapat menjadikan hasil menjadi salah. Sebagai contoh adalah proses penjumlahan bilangan 4-bit berikut,

```
  0100  (4)
+ 0101  (5)
  1001  (5)      → hasil salah
```

↳bit tanda salah

Pada sistem bilangan desimal, 4 + 5 = 9. Namun karena 9 di luar jangkauan bilangan biner komplemen dua 4 bit (lihat kembali Bab 2), maka penjumlahan di atas akan memberikan hasil yang salah. Kesalahan terjadi karena hasil operasi terletak di luar jangkauan suatu sistem bilangan biner 4-bit. Kesalahan ini sering disebut *overflow*. Bagaimana jika penjumlahan di atas dilakukan untuk format biner 5-bit? Tentu saja hasilnya akan benar karena hasil penjumlahan masih dalam jangkauan biner 5-bit. Silakan dicoba!

Contoh 26

Jumlahkan dua bilangan biner tak bertanda 10010 dan 10001!

Jawab:

```
  10010
  10001 +
 100011
```

Contoh 27

Ulangi Contoh 26, namun gunakan format bilangan komplemen dua 5 bit!

Jawab:

```
  10010
  10001 +
1 00011
```

Pada penjumlahan dua bilangan negatif tersebut, terjadi *overflow*. Hasilnya salah karena merupakan bilangan positif.

Terdapat beberapa IC yang dapat digunakan untuk operasi penjumlahan. Salah satu IC yang dapat digunakan untuk praktek penjumlahan adalah IC ALU (*Arithmatic and Logic Unit*). IC tersebut dapat melakukan operasi aritmatika (penjumlahan dan pengurangan) dan logika. Pada sebuah ALU, terdapat bit khusus yang merupakan indikasi adanya *overflow*. Jika bit tersebut sama dengan 1, maka hasil perhitungan *overflow*; dan jika bit tersebut sama dengan nol, maka hasil perhitungan masih dalam jangkauan perhitungan ALU. Lebih jauh tentang bit ini akan dijelaskan di Bab 5.

b) Pengurangan

Pengurangan pada sistem bilangan biner dilakukan dengan mengurangkan setiap bit dimulai dari LSB sebagaimana pengurangan pada bilangan desimal. Perhatikan contoh berikut.

```
  01101 (13)          01001  (9)
- 01001  (9)        - 00100  (4)
  00100  (4)          00101  (5)
```

Pengurangan pada sistem bilangan biner dapat pula dilakukan dengan manambahkan dengan komplemen duanya. Contoh:

$$9 - 4 = 9 + (-4)$$
$$01001_{bin} - 00100_{bin} = 01001_{bin} + 11100_{bin}$$

```
  01001  (9)
+ 11100 (-4)
-----------
1 00101  (5)
```

↳tidak digunakan

c) 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 biner positif.

Perkalian dengan 2^n

Perkalian dengan dua merupakan operasi aritmatika perkalian paling sederhana. Ingat, dalam sistem bilangan biner, dua atau 10_{bin} merupakan angka terkecil pertama di atas satu. Pada perangkat keras digital, seperti pada sebuah mikroprosesor, perkalian dengan suatu bilangan dapat dibentuk oleh operasi perkalian dengan dua yang dikerjakan berulang-ulang. Hasil perkalian dengan dua dapat diperoleh dengan menggeser bilangan yang dikalikan ke kiri satu bit. Pada mikroprosesor, perkalian dengan dua dapat dibentuk dengan operasi geser kiri satu bit. Bit paling kanan diisi nol. Contoh:

```
  1001  (9)
x   10  (2)
-----------
 10010 (18)
```

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

```
  1001  (9)          1001  (9)          1001    (9)
x  100 (2²)       x 1000 (2³)       x 10000000 (2⁷)
------------      ------------      ------------------
100100 (36)       1001000 (72)      10010000000 (1152)
```

Perkalian bilangan positif dengan bilangan positif

Dengan dasar perkalian dengan dua, dapat dibentuk perkalian dengan bilangan lain. Contoh:

```
  1001  (9)
x 1010 (10)
-----------
  0000
 1001
 0000
1001
-------
1011010 (90)
```

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

```
   1001   (9)
 x 1010  (10)
   0000   (hasil kali 1)
  +1001   (hasil kali 2)
   1001   (subtotal 1)
 +0000    (hasil kali 3)
  0100    (subtotal 2)
+1001     (hasil kali 4)
 1011010 (90)
```

Pada operasi 1001 X 1010, pertama akan dilakukan 1001 X 0. Hasilnya adalah 0000, sebut saja hasil ini adalah hasil kali 1. Kemudian dilakukan perkalian 1001 X 10, hasilnya adalah bilangan 1001 itu sendiri yang digeser ke kiri satu bit. Sebut saja hasil ini sebagai 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 adalah 0000 yang digeser kiri satu bit terhadap sub total 3. Hasilnya disebut hasil kali 3. Subtotal 1 dan hasil kali 3 dijumlahkan dengan format bit mengikuti hasil kali 3 menjadi subtotal 2. Kemudian 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 bilangan biner n bit dikalikan bilangan biner m bit, maka hasil kalinya mempunyai panjang bit maksimal n+m bit.

Contoh 28

Ulangi Contoh 27 namun untuk operasi perkalian!

Jawab:

```
     10010
     10001 ×
     10010
10010     +
100110010
```

Perkalian bilangan positif dengan bilangan negatif atau sebaliknya

Perkalian bilangan positif dengan bilangan negatif atau sebaliknya dilakukkan dengan mengalikan dengan bilangan positifnya. Bit tanda hasil perkalian dapat disesuaikan setelah hasil perkalian didapat. Sebagai contoh, 9 X (-10) akan memberikan hasil yang sama dengan 9 X 10, kemudian 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)
                    ↳bit tanda
```

Demikian pula untuk (-9) X 10 akan memberkan hasil sama dengan 9 X 10, kemudian hasilnya dinegasikan. Pada proses perkalian ini harus diperhatikan panjang bit hasil perkalian. Sesuaikan panjang bit hasil akhir perkalian menggunakan aturan pengaturan format panjang bit biner yang telah diuraikan pada subbab 2.6.c.

Perkalian bilangan negatif dengan positif dapat dilakukan secara langsung. Sebagai contoh (-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 (hasil kali 1). 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 agar mempunyai panjang bit sama dengan hasil kali 4. Seperti diuraikan pada subbab 2.6(c), penambahan bit 1 di depan bilangan biner negatif tidak mengubah arti. Kemudian, dilakukan perkalian 10111 dengan 00000, hasilnya adalah 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. Dengan cara di atas, kita dapat menghitung (-9) X 10 namun tidak dapat 10 X (-9).

Perkalian bilangan negatif dengan bilangan negatif

Pada kasus ini akan dihasilkan bilangan positif. Cara mengerjakannya adalah mengonversikan kedua bilangan menjadi positif, baru dikalikan. 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 kecuali jika format panjang bit biner sudah ditentukan.

Contoh 29

Ulangi Contoh 28 namun untuk operasi perkalian bilangan komplemen dua 5 bit!

Jawab:

Dengan mengubah kedua bilangan tersebut menjadi positif, maka

```
10010                 01110
10001 ×  menjadi      01111 ×
                      01110
                     01110
                    01110
                   01110    +
                  11010010
```

Dengan disesuaikan menjadi format 10 bit maka didapat hasil 00011010010

d) Pembagian

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

Contoh:

```
   0011                1001
11|1001          1010|1011011
  011              1010
  0011               10
    11                0
     0              101
                      0
                   1011
                   1010
                      1 → sisa pembagian
```

Untuk pembagian dengan bilangan negatif, dapat dilakukan dengan menjadikan bilangan negatif ke positif terlebih dahulu. 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. Proses ini dapat dikatakan kebalikan dari proses perkalian dengan dua. Bit paling kanan akan hilang pada penggeseran ini.

Pada perangkat keras digital, seperti pada sebuah mikroprosesor, pembagian dengan 2^n dapat dibentuk oleh operasi pembagian dengan dua yang dikerjakan berulang-ulang. Contoh:

```
   1000 (8)
÷    10 (2)
   100 (4)
  1000 (8)
```

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

e) 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 bilangan biner. Contoh:

```
   4   →     0100
 + 5   →   + 0101
   9   →     1001

  33   →     0011 0011
+ 54   →   + 0101 0100
  87   →     1000 0111
```

Penjumlahan Bilangan 10 atau Lebih

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

```
  7  →      0111
+ 6  →    + 0110
 13  →      1101   → bukan format bilangan BCD
```

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

```
  7  →      0111
+ 6  →    + 0110
 13          1101   → bukan format bilangan BCD
           + 0110   → tambahkan dengan 6
      0001 0011   → 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
```

Perhatikan hasil penjumlahan sebelum ditambah dengan 6. Nibble 1000 merupakan bilangan BCD (sama dengan 8_{des}), sehingga tidak perlu ditambah 6. Sementara itu nibble 1011 bukan merupakan bilangan BCD, sehingga perlu ditambah 6. Hasil penjumlahan dengan 6 tersebut memberikan bawaan ke nibble 0110 sehingga nibble paling kiri menjadi 0111 (sama dengan 7_{des}).

Contoh 30

Ulangi Contoh 29 namun gunakan format BCD!

jawab:

```
10010bin          18des      0001 1000BCD
10001bin  +   ⇨   17des   ⇨  0001 0111BCD  +
                             0010 1111BCD
                                  0110BCD  +
                             0011 0101BCD
```

2.10 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. Perlu diperhatikan di sini bahwa pada bilangan heksadesimal komplemen dua, jika MSD≥8 maka bilangan tersebut adalah negatif. Contoh:

```
  85
+ 42
  C7
```

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 dua dalam biner
  ↘   ↓   ↙
  A   C   9           → komplemen dua dalam heksadesimal
```

Ada cara cepat untuk mendapatkan komplemen dua dari suatu bilangan heksadesimal, yaitu dengan mengurangkan 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

Operasi penjumlahan merupakan operasi aritmatika dasar yang harus dapat dilakukan oleh sebuah perangkat pemroses perhitungan seperti mikroprosesor. Dari operasi ini dapat dikembangkan operasi aritmatika yang lain, seperti pengurangan, perkalian dan pembagian. Dan pada hakekatanya, sebuah prosesor melakukan operasi perkalian dan operasi yang lain hanya dengan melakukan operasi penjumlahan berulang-ulang. Pada sebuah perangkat keras digital, operasi penjumlahan dapat dilakukan oleh sebuah untai penjumlah. Terdapat dua macam untai penjumlah, yaitu penjumlah paruh (*half adder*) dan penjumlah penuh (*full adder*). Penjelasan tentang untai penjumlah akan dibahas di Bab 5.

2.11 ASCII

Pada sistem digital terutama sistem komputer, berbagai macam informasi yang ada dibentuk dari serangkaian huruf, angka, tanda baca dan karakter kontrol. Informasi tersebut harus dapat ‘dimengerti’ oleh sistem digital. Terdapat standar internasional untuk

mengkonversi barbagai macam informasi tersebut ke format biner yang dapat dimengerti sistem digital. Salah satunya adalah format dari *American Standard Code for Infomation Interchange* (ASCII). Standar ini mengonversi berbagai informasi ke dalam 7-bit data biner sebagaimana Tabel 2.6

Tabel 2.6 Karakter ASCII

Bit-bit rendah	Bit-bit tinggi							
	000	**001**	**010**	**011**	**100**	**101**	**110**	**111**
0000	NUL	DLE	SPACE	0	@	P	`	p
0001	SOH	DC1	!	1	A	Q	a	q
0010	STX	DC2	“	2	B	R	b	r
0011	ETX	DC3	#	3	C	S	c	s
0100	EOT	DC4	$	4	D	T	d	t
0101	ENQ	NAK	%	5	E	U	e	u
0110	ACK	SYN	&	6	F	V	f	v
0111	BEL	ETB	‘	7	G	W	g	w
1000	BS	CAN	(	8	H	X	h	x
1001	HT	EM	)	9	I	Y	i	y
1010	LF	SUB	*	:	J	Z	j	z
1011	VT	ESC	+	;	K	[	k	{
1100	FF	FS	,	<	L	\	l	|
1101	CR	GS	-	=	M	]	m	}
1110	SO	RS	.	>	N	^	n	~
1111	SI	US	/	?	O	_	o	DEL

Karakter kontrol meliputi:

NUL	Null/Idle	SI	Shift In
SOH	Start of header	DLE	Data link escape
STX	Start of text	DCI-DC4	Device control
ETX	End of text	NAK	Negative acknowledgement
EOT	End of transmission	SYN	Synchronous idle
ENQ	Enquiry	ETB	End of transmission block
ACQ	Acknowledgement	CAN	Cancel (error in data)
BEL	Audible signal	EM	End of medium
BS	Back space	SUB	Special sequence
HT	Horizontal tab	ESC	Escape
LF	Line feed	FS	File separator
VT	Vertical tab	GS	Group separator
FF	Form feed	RS	Record separator
CR	Cariage return	US	Unit separator
SO	Shift out	DEL	Delete/Idle

Bakuan tersebut banyak dipakai pada berbagai sistem komputer. Karakter kontrol biasanya digunakan untuk keperluan pendukung transmisi data; sedangkan angka, huruf dan

tanda baca digunakan untuk konversi data dari dan ke pengguna. Sebagai contoh, karakter A akan disimpan dalam media penyimpanan menjadi 100 0001, agar mengeluarkan suara *beep* maka komputer diberi data BEL yaitu 000 0111, dan agar *printer* menarik kertas maka dikirimlah kode FF (*form feed*) yaitu 000 1100.

Pada kode ASCII, angka dan huruf ditempatkan sedemikian rupa sehingga dapat dilakukan proses pengurutan secara sederhana. Huruf A, B, C hingga Z ditempatkan berurutan dari 100 0001 hingga 101 1010. Huruf kecil ditempatkan setelah huruf besar. Dan angka ditempatkan sebelum huruf besar.

Kode ASCII hanya merupakan konversi dari lambang yang dimengerti manusia ke kode yang dimengerti konputer. Kode ini tidak dimaksudkan untuk perhitungan matematis. Sebagai contoh, jika diberikan perintah ke komputer untuk menjumlahkan bilangan desimal 1 dan 2, bukan berarti kode untuk 1 yaitu 011 0001 dijumlahkan dengan kode untuk 2 yaitu 011 0010. Jika untuk perhitungan digunakan format data biner 4 bit, kode '1' = 011 0001 akan diterjemahkan ke 0001 dan '2' = 011 0010 akan diterjemahkan ke 0010. Baru kedua data dijumlahkan dan hasilnya kembali diformat dalam ASCII.

Karakter		Kode ASCII		Nilai dalam biner
'1'	⇨	011 0001	⇨	0001
'2'	⇨	011 0010	⇨	0010
				+
'3'	⇦	011 0011	⇦	0011

2.12 Istilah Penting

ASCII
basis
bawaan
bilangan bertanda
bin
Binary Coded Decimal
biner
bit
bit bawaan
bit paling berarti
bit paling tidak berarti
bit tanda
bit-bit rendah
bit-bit tinggi
bobot
byte
des
desimal
digit paling berarti
digit paling tidak berarti
eksponen
fixed-point number
floating-point number
heks
heksadesimal
IEEE
integer number
komplemen dua
komplemen satu
Least Significant Bit (LSB)
Least Significant Digit (LSD)
mantisa
Most Significant Bit (MSB)
Most Significant Digit (MSD)
nibble
nibble rendah
nibble tinggi
nilai posisi
okt
oktal
overflow
presisi ganda
presisi tunggal
radiks
signed number
unsigned number
word

Soal-soal Latihan

1. Ubahlah setiap bilangan berikut menjadi bilangan biner, oktal, heksadesimal dan BCD.
   a. 4
   b. 15
   c. 36
2. Ubahlah setiap bilangan biner tak bertanda berikut menjadi bilangan desimal.
   a. 00111
   b. 10001
   c. 10110
3. Ubahlah bilangan biner pada Soal 2 menjadi bilangan biner 8 bit.
4. Jika setiap bilangan biner pada Soal 2 menggunakan sistem bilangan biner komplemen dua, ubahlah menjadi bilangan desimal.
5. Jika setiap bilangan biner pada Soal 2 menggunakan sistem bilangan biner komplemen dua, ubahlah menjadi bilangan biner 8 bit.
6. Ubahlah bilangan biner pada Soal 2 menjadi bilangan oktal dan heksadesimal tanpa terlebih dahulu mengubahnya menjadi bilangan desimal.
7. Ubahlah setiap bilangan oktal berikut menjadi bilangan desimal.
   a. 74
   b. 105
   c. 372
8. Ubahlah bilangan oktal pada Soal 7 menjadi bilangan biner tanpa terlebih dahulu mengubahnya menjadi bilangan desimal.
9. Ubahlah setiap bilangan heksadesimal berikut menjadi bilangan desimal.
   a. 7
   b. 1E
   c. A7F
10. Ubahlah bilangan heksadesimal pada Soal 9 menjadi bilangan biner tanpa terlebih dahulu mengubahnya menjadi bilangan desimal.
11. Berapa bit yang dibutuhkan untuk melakukan penghitungan hingga 511?
12. Berapa nilai maksimal untuk bilangan biner 1, 2, 4, 8, 16 dan 32 bit?
13. Berapa bit yang dibutuhkan untuk merepresentasikan bilangan dari 0 hingga 999 menggunakan kode biner dan BCD?
14. Dengan format 8 bit, berapa nilai terkecil dan terbesar jika menggunakan sistem bilangan biner tak bertanda dan komplemen dua?
15. Suatu nilai tegangan akan diwakili oleh bilangan biner. Berapa nilai tegangan yang dapat diwakili jika digunakan bilangan biner:
   a. 8 bit
   b. 10 bit

16. Berapa digit yang dibutuhkan untuk merepresentasikan bilangan dari 0 hingga 999 menggunakan kode oktal dan heksadesimal?
17. Sebuah kartu grafis dikatakan dapat menampilkan warna sesuai aslinya jika untuk menampilkan warna merah, hijau dan biru digunakan format data masing-masing 8 bit.
    a. Hitunglah berapa variasi setiap warna merah, hijau dan biru yang dapat ditampilkan?
    b. Hitunglah berapa variasi warna yang dapat ditampilkan secara keseluruhan?
18. Konversikan bilangan komplemen dua 1011_{bin} ke oktal:
    a. 2 digit
    b. 4 digit
    c. 8 digit
19. Ubahlah bilangan heksadesimal komplemen dua berikut menjadi biner 4 bit.
    a. D
    b. FD
    c. FFFD
20. Kerjakan operasi matematis berikut.
    a. 10111+00101
    b. 00100+00111
    c. 01110+10011
    d. 10001+01111
21. Jika setiap bilangan biner pada soal nomor 20 menggunakan sistem bilangan biner komplemen dua, ulangi operasi matematis tersebut.
22. Ulangi Soal nomor 20 namun untuk operasi perkalian.
23. Ulangi Soal nomor 20 namun untuk operasi perkalian menggunakan sistem bilangan biner komplemen dua.
24. Ulangi Soal nomor 20 dan berikan hasilnya dalam format BCD.
25. Bagaimana cara mengubah suatu bilangan biner negatif komplemen dua ke bilangan desimal?
26. Ubahlah bilangan biner berikut menjadi bilangan desimal:
    a. 1001,1001
    b. 10011011001,10110
27. Kerjakan operasi perkalian bilangan biner berikut.
    a. 111 × 101
    b. 1001 × 1011
    c. 101,101 × 110,010
    d. 0,1101 × 0,1011
28. Kerjakan operasi pembagian bilangan biner berikut.
    a. 1100 ÷ 100
    b. 111111 ÷ 1001
    c. 10111 ÷ 100
    d. 10110,1101 ÷1,1
29. Ceklah kebenaran hasil perhitungan Soal 28.d dengan mengerjakan ulang dalam bilangan desimal!

Soal Khusus

Dalam sistem audio digital, isyarat audio biasanya dicuplik (disampling) 44.000 kali per detik. Hasil pencuplikan merupakan data biner yang disimpan di permukaan CD audio. Setiap nilai biner di sebuah data yang tersimpan mewakili sebuah nilai tegangan tertentu.

a. Jika data biner yang tersimpan harus dapat mewakili 256 variasi tegangan isyarat audio, berapa bit panjang format data yang harus digunakan?
b. Jika data biner yang tersimpan adalah 16-bit, berapa bit data yang harus tersimpan untuk merekam suara 1 detik?
c. Suatu CD dapat menyimpan data 700 juta byte. Berapa detik isyarat audio yang dapat disimpan dalam CD tersebut jika digunakan format data 16-bit?