1
SISTEM BILANGAN

Banyak sistem bilangan yang dapat dan telah dipakai dalam melaksanakan perhitungan. Tetapi ada sistem bilangan yang sudah jarang dipakai ataupun tidak dipakai lagi sama sekali dan ada pula sistem bilangan yang hanya dipakai pada hal-hal tertentu saja. Sistem bilangan limaan (quinary) dipergunakan oleh orang Eskimo dan orang Indian di Amerika Utara zaman dahulu. Sistem bilangan Romawi yang sangat umum dipakai pada zaman kuno, kini pemakaiannya terbatas pada pemberian nomor urut seperti I untuk pertama, II untuk kedua, V untuk kelima dan seterusnya; kadang-kadang dipakai juga untuk penulisan tahun seperti MDCCCIV untuk menyatakan tahun 1804. Sistem bilangan dua belasan (duodecimal) sampai kini masih banyak dipakai seperti 1 kaki = 12 Inci, 1 lusin = 12 buah dan sebagainya. Namun yang paling umum dipakai kini adalah sistem bilangan puluhan (decimal) yang kita pakai dalam kehidupan sehari-hari.

Karena komponen-komponen komputer digital yang merupakan sistem digital bersifat saklar (switch), sistem bilangan yang paling sesuai untuk komputer digital adalah sistem bilangan biner (binary). Keserdehanaan pengubahan bilangan biner ke bilangan oktal atau heksadesimal dan sebaliknya, membuat bilangan oktal dan heksadesimal juga banyak dipakai dalam dunia komputer, terutama dalam hubungan pengkodean. Bilangan Biner, Oktal dan Heksadesimal akan dibahas dalam bab ini didahului dengan pembahasan singkat tentang bilangan desimal sebagai pengantar.

1.1 Sistem Bilangan Puluhan

Sistem bilangan puluhan atau desimal (decimal system) adalah sistem bilangan yang kita pergunakan sehari-hari. Sistem bilangan ini disusun oleh sepuluh simbol angka yang mempunyai nilai yang berbeda satu sama lain dan karena itu dikatakan bahwa dasar/basis atau akar (base, radix) dari pada sistem bilangan ini adalah sepuluh. Kesepuluh angka dasar tersebut, sebagaimana telah kita ketahui, adalah: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9. Nilai yang terkandung dalam setiap simbol angka secara terpisah (berdiri sendiri) disebut nilai mutlak (absolute value). Jelaslah bahwa harga maksimum yang dapat dinyatakan oleh hanya satu angka adalah 9. Harga-harga yang lebih besar dapat dinyatakan hanya dengan memakai lebih dari satu angka secara bersama-sama. Nilai yang dikandung oleh setiap angka di dalam suatu bilangan demikian ditentukan oleh

letak angka itu di dalam deretan di samping oleh nilai mutlaknya. Cara penulisan ini disebut sebagai sistem nilai (berdasarkan) letak/posisi (positional value system). Angka yang berada paling kanan dari suatu bilangan bulat tanpa bagian pecahan disebut berada pada letak ke 0 dan yang di kirinya adalah ke 1, ke 2 dan seterusnya sampai dengan ke (n-1) jika bilangan itu terdiri dari n angka. Nilai letak dari pada angka paling kanan, yaitu kedudukan ke 0, adalah terkecil, yaitu $10^0 = 1$. Nilai letak ke 1 adalah 10^1, nilai letak ke 2 adalah $10^2 = 100$, dan seterusnya nilai letak ke n-1 adalah 10^{n-1}.

Untuk bilangan yang mengandung bagian pecahan, bagian bulat dan pecahannya dipisahkan oleh tanda koma (tanda titik di Inggris, Amerika, dan lain-lain). Angka di kanan tanda koma puluhan (decimal point) disebut pada kedudukan negatif, yaitu letak ke -1, ke -2 dan seterusnya dan nilai letaknya adalah 10^{-1}, 10^{-2}, dan seterusnya 10^{-m} untuk kedudukan ke (-m) di kanan koma puluhan. Nilai yang diberikan oleh suatu angka pada suatu bilangan adalah hasil kali dari pada nilai mutlak dan nilai letaknya. Jadi, nilai yang diberikan oleh angka 5 pada bilangan 1253,476 adalah $5x10^1 = 50$ dan yang diberikan oleh angka 7 adalah $7x10^{-2} = 0,07$.

Secara umum, suatu bilangan puluhan yang terdiri atas n angka di kiri tanda koma puluhan dan m angka di kanan tanda koma puluhan, yang dapat dinyatakan dalam bentuk:

$N = a_{n-1}\, a_{n-2} \ldots a_1\, a_0, a_{-1}\, a_{-2} \ldots a_{-m}$,

mempunyai harga yang dapat dinyatakan dalam bentuk:

$$N = a_{n-1}\, 10^{n-1} + a_{n-2}\, 10^{n-2} + \ldots + a_1\, 10^1 + a_0\, 10^0 + a_{-1}\, 10^{-1} + a_{-2}\, 10^{-2} + \ldots + a_{-m}\, 10^{-m} \quad (1.1)$$

1.2 Biner, Oktal dan Heksadesimal

Secara umum, semua sistem digital bekerja dengan sistem bilangan biner (binary) sehingga sistem binerlah yang paling penting dalam sistem digital. Selain sistem bilangan biner, sistem yang paling umum dipakai dalam pengkodean instruksi untuk komputer digital adalah sistem bilangan oktal dan hekadesimal.

Harga dalam desimal (puluhan) yang dinyatakan oleh suatu bilangan biner, oktal, heksadesimal atau yang lain-lain yang bukan desimal dapat dihitung dengan memakai rumus:

$$a_{n-1}a_{n-2}\ldots a_1a_0\, a_{-1}a_{-2}\ldots a_{-m} = a_{n-1}\, R^{n-1} + a_{n-2}\, R^{n-2} + \ldots + a_1\, R^1 + a_0\, R^0 + a_{-1}\, R^{-1} + \ldots + a_{-m}\, R^{-m} \quad (1.2)$$

dengan: a_{n-1} = angka yang paling kiri,

R = Angka dasar dari pada sistem bilangan
n = cacah angka yang menunjukan bilangan bulat
m = cacah angka yang menunjukkan bilangan pecahan

Persamaan (1.2), yang merupakan bentuk umum dari pada persamaan (1.1), berlaku untuk semua sistem bilangan yang berdasarkan letak yang tegas. Untuk semua sistem bilangan seperti bilangan Romawi, misalnya, persamaan ini tentunya tak dapat dipergunakan.

1.2.1 Bilangan Biner

Sistem bilangan biner mempunyai hanya dua macam simbol angka, yaitu 0 dan 1, dan karena itu dasar dari sistem bilangan ini adalah dua. Harga yang ditunjukkan oleh bilangan biner dalam puluhan dapat dihitung dengan memakai persamaan (1.2) di atas dengan memasukkan R= 2 ke dalamnya. Sebagai contoh, harga bilangan biner 101,01 adalah :

$1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = 5{,}25$

Dapat disadari bahwa bila kita bekerja dengan lebih dari satu bilangan, maka kita akan mengalami kebingungan bila kita tidak memakai suatu tanda yang menyatakan dasar setiap bilangan. Untuk mencegah hal ini, pada setiap bilangan dicantumkan dasar bilangannya, seperti $(101)_2$ atau $\underline{101}_2$ untuk menyatakan bilangan 101 dalam biner. Jadi, contoh diatas dapat dituliskan sebagai :

$(101{,}01)_2 = (5{,}25)_{10}$

Untuk uraian selanjutnya, kita akan memakai cara penulisan ini bilamana diperlukan. Bilamana dasar dari pada bilangan sudah jelas dari uraian ataupun bila kita hanya membicarakan satu sistem bilangan, tentunya kita tidak perlu dan tak akan memberikan tanda tersebut. Didalam praktek pemrograman komputer, sering tanda tersebut hanya diberikan kepada bilangan yang bukan puluhan.

1.2.2 Bilangan Oktal dan Heksadesimal

Bilangan Oktal mempunyai delapan macam simbol angka, yaitu: 0, 1, 2, 3, 4, 5, 6, 7, dan karena itu, dasar daripada bilangan ini adalah delapan. Harga desimal yang dinyatakan oleh suatu bilangan oktal diperoleh dengan memasukkan R= 8 kedalam pers. (1.2) di depan. Sebagai contoh,

$(235{,}1)_8 = 2 \times 8^2 + 3 \times 8^1 + 5 \times 8^0 + 1 \times 8^{-1} = (157{,}125)_{10}$.

Sistem bilangan Heksadesimal terdiri atas 16 simbol angka sehingga bilangan dasarnya adalah 16. Sepuluh dari simbol tersebut diambil dari ke-sepuluh simbol angka pada sistem bilangan puluhan dan enam angka yang lain diambil dari huruf dalam abjad A - F. Jadi, ke-16 simbol heksadesimal adalah: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F. Huruf-huruf A, B, C, D, C dan F secara berturut-turut bernilai 10, 11, 12, 13, 14, 15.

Harga desimal yang dinyatakan oleh bilangan heksadesimal juga dapat dihitung dengan memasukkan harga R = 16 kedalam pers. (1.2) di depan. Sebagai contoh,

$$(3C5,A)_{16} = 3 \times 16^2 + 12 \times 16^1 + 5 \times 16^0 + 10 \times 16^{-1}$$
$$= (965,0625)_{10}$$

Yang membuat sistem bilangan oktal dan heksadesimal banyak dipakai dalam sistem digital adalah mudahnya pengubahan dari biner ke oktal dan heksadesimal, dan sebaliknya, seperti akan dibicarakan dalam sub-bab berikut ini.

1.3 Konversi Bilangan

Konversi bilangan desimal ke sistem biner diperlukan dalam menerjemahkan keinginan manusia kedalam kode-kode yang dikenal oleh sistem digital, terutama komputer digital. Konversi dari biner ke desimal diperlukan untuk menterjemahkan kode hasil pengolahan sistem digital ke informasi yang dikenal oleh manusia. Pengubahan (konversi) dari biner ke oktal dan heksadesimal dan sebaliknya merupakan pengantara konversi dari/ke biner ke/dari desimal. Konversi ini banyak dilakukan karena disamping cacah angka biner yang disebut juga "bit", singkatan dari "**bi**nary digi**t**", jauh lebih besar dibandingkan dengan angka-angka pada sistem oktal dan heksadesimal, juga karena konversi itu sangat mudah.

Konversi dari biner, oktal dan heksadesimal ke sistem bilangan desimal, seperti telah dijelaskan di bagian depan, dapat dilakukan dengan memakai persamaan (1.2). Konversi sebaliknya akan diterangkan dalam sub-sub bab berikut ini.

1.3.1 Konversi Desimal-Biner

Kalau kita perhatikan konversi dari biner ke desimal dengan memakai pers.(1.2), maka dapat dilihat bahwa untuk bagian bulat (di kiri tanda koma) kita peroleh dengan melakukan perkalian dengan 2 setiap kita bergerak ke kiri. Untuk bagian pecahan, kita melakukan pembagian dengan 2 setiap kita bergerak ke kanan. Untuk melakukan konversi dari desimal ke biner kita melakukan sebaliknya, yaitu untuk bagian bulat bilangan desimal kita bagi dengan 2 secara

berturut-turut dan sisa pembagian pertama sampai yang terakhir merupakan angka-angka biner paling kanan ke paling kiri. Untuk bagian pecahan, bilangan desimal dikalikan 2 secara berturut-turut dan angka di kiri koma desimal hasil setiap perkalian merupakan angka biner yang dicari, berturut-turut dari kiri ke kanan. Contoh berikut ini memperjelas proses itu.

Contoh 1.

Tentukanlah bilangan biner yang berharga sama dengan bilangan desimal 118.

Pembagian secara berturut-turut akan menghasilkan:

118 : 2 = 59 sisa 0	7 : 2 = 3 sisa 1
59 : 2 = 29 sisa 1	3 : 2 = 1 sisa 1
29 : 2 = 14 sisa 1	1 : 2 = 0 sisa 1
14 : 2 = 7 sisa 0	0 : 2 = 0 sisa 0

Jadi, $(118)_{10} = (01110110)_2$

Perhatikan bahwa walaupun pembagian diteruskan, hasil berikutnya akan tetap 0 dan sisanya juga tetap 0. Ini benar karena penambahan angka 0 di kiri bilangan tidak mengubah harganya.

Contoh 2.

Tentukanlah bilangan biner yang berharga sama dengan bilangan desimal 0,8125.

Pengalian secara berturut-turut akan menghasilkan :

0.8125 x 2 = 1,625	0,500 x 2 = 1,000
0,625 x 2 = 1,250	0,000 x 2 = 0,000
0,250 x 2 = 0,500	

Jadi, $(0,8125)_{10} = (0,11010)_2$

Perhatikan bahwa angka-angka biner yang dicari adalah angka yang di kiri tanda koma, dan yang paling kiri dalam bilangan biner adalah angka di kiri koma hasil perkalian pertama. Juga perhatikan bahwa walaupun pengalian diteruskan hasil perkalian akan tetap 0 dan ini benar karena penambahan angka 0 ke kanan tidak akan mengubah harganya.

Contoh 3.

Ubahlah bilangan desimal 457,65 ke bilangan biner.

Untuk melakukan konversi ini, dilakukan pembagian untuk bagian bulatnya dan pengalian untuk bagian pecahannya seperti yang dilakukan pada kedua contoh sebelumnya, dengan hasil sebagai berikut ini:

457 : 2 = 228 sisa 1	0,65 x 2 = 1,3
228 : 2 = 114 sisa 0	0,30 x 2 = 0,6
114 : 2 = 57 sisa 0	0,60 x 2 = 1,2
57 : 2 = 28 sisa 1	0,20 x 2 = 0,4
28 : 2 = 4 sisa 0	0,40 x 2 = 0,8
14 : 2 = 7 sisa 0	0,80 x 2 = 1,6
7 : 2 = 3 sisa 1	0,60 x 2 = 1,2
3 : 2 = 1 sisa 1	0,20 x 2 = 0,4
1 : 2 = 0 sisa 1	0,40 x 2 = 0,8
	0,80 x 2 = 1,6

Jadi, $(457,65)_{10} = (111001001,1010011001 .....)_2$

Dari contoh terakhir ini dapat dilihat bahwa untuk bagian pecahan, pengalian dengan 2 akan berulang-ulang menghasilkan deretan 1,6; 1,2; 0,4; 0,8 yang berarti bahwa deretan angka biner 11001100 akan berulang terus. Ini berarti bahwa ada bilangan pecahan puluhan yang tak dapat disajikan dalam biner dengan ketelitian 100 %. Kesalahan atau ralat konversi itu semakin kecil bila cacah angka biner (bit) yang dipergunakan lebih besar. Bagaimanapun juga, cacah bit dalam setiap sistem digital sudah tertentu sehingga ketelitian pengkodean untuk setiap sistem digital sudah tertentu pula.

1.3.2 Konversi Biner-Oktal-Heksadesimal

Kemudahan konversi biner-oktal-heksadesimal secara timbal balik terletak pada kenyataan bahwa 3 bit tepat dapat menyatakan angka terbesar dalam oktal, yaitu 7, dan 4 bit tepat dapat menyatakan angka terbesar dalam heksadesimal, yaitu $F=(15)_{10}$. Ini berarti bahwa untuk mengubah bilangan biner ke oktal, bilangan biner dapat dikelompokkan atas 3 bit setiap kelompok dan untuk mengubah biner ke heksadesimal, bilangan biner dikelompokkan atas 4 bit setiap kelompok. Pengelompokan harus dimulai dari kanan bergerak ke kiri. Sebagai contoh, untuk memperoleh setara dalam oktal dan heksadesimal, bilangan biner 1011001111 dapat dikelompokkan sebagai berikut:

1 011 001 111	10 1100 1111
$(1\ 3\ 1\ 7)_8$	$(2\ C\ F)_{16}$

Konversi sebaliknya, dari oktal dan heksadesimal ke biner juga dapat

dilakukan dengan mudah dengan menggantikan setiap angka dalam oktal dan heksadesimal dengan setaranya dalam biner.

Contoh 1.

$(3456)_8 = (011\ 100\ 101\ 110)_2$
$(72E)_{16} = (0111\ 0010\ 1110)_2$

Dari contoh ini dapat dilihat bahwa konversi dari oktal ke heksadesimal dan sebaliknya akan lebih mudah dilakukan dengan mengubahnya terlebih dahulu ke biner.

Contoh 2.

$(3257)_8 = (011\ 010\ 101\ 111)_2$
$(0110\ 1010\ 1111)_2 = (6AF)_{16}$

Perhatikan bahwa bilangan biner dalam konversi oktal biner dan konversi biner-heksadesimal hanyalah berbeda dalam pengelompokannya saja.

1.3.3 Konversi Desimal-Oktal dan Heksadesimal

Konversi desimal ke oktal dan desimal ke heksadesimal dapat dilakukan dengan melakukan pembagian berulang-ulang untuk bagian bulat dan perkalian berulang-ulang untuk bagian pecahan seperti yang dilakukan pada konversi desimal-biner di bagian depan. Sebenarnya cara ini berlaku untuk semua dasar sistem bilangan.

Contoh : Untuk $(205{,}05)_{10}$

Oktal:	Heksadesimal:
205 : 8 = 25 sisa 5	205 : 16 = 12 sisa 13 = D
25 : 8 = 3 sisa 1	12 : 16 = 0 sisa 12 = C
3 : 8 = 0 sisa 3	
0,05 x 8 = 0,4	0,05 x 16 = 0,8
0,40 x 8 = 3,2	0,80 x 16 = 12,8 (12 = C)
0,20 x 8 = 1,6	0,80 x 16 = 12,8
0,60 x 8 = 4,8	
0,80 x 8 = 6,4	
0,40 x 8 = 3,2	
0,20 x 8 = 1,6	

Jadi, $(205{,}05)_{10} = (315{,}031463146...)_8 = (CD{,}0CCCC..)_{16}$

1.4 Komplemen

Dalam sistem digital, semua perhitungan aljabar, baik perjumlahan, pengurangan, perkalian maupun pembagian, dilaksanakan dengan penjumlahan. Perkalian dan pembagian dilaksanakan dengan melakukan penjumlahan diselingi penggeseran. Pelaksanaan pengurangan dengan penjumlahan dilakukan dengan menambahkan harga negatif bilangan pengurang. Ini dapat dilihat dari persamaan:

$$X - Y = X + (- Y)$$

Dalam pelaksanaanya, semua bilangan negatif dinyatakan dalam harga komplemennya. Untuk setiap sistem bilangan dengan dasar R, dibedakan 2 jenis komplemen, yaitu komplemen R dan komplemen R-1. Jadi, untuk sistem bilangan desimal dengan R= 10 ada komplemen 10 dan ada komplemen 9; untuk oktal ada komplemen 8 dan komplemen 7; untuk heksadesimal ada komplemen 16 dan komplemen 15, dan seterusnya.

Komplemen suatu bilangan N dalam sistem bilangan dengan dasar R didefinisikan sebagai berikut :

Komplemen R dari N : $(N)_{c,R} = R^n - N , \quad N \neq 0$ (1.3)
$= 0 , \quad N = 0$

Komplemen R-1 dari N : $(N)_{c,R-1} = R^n - R^{-m} - N$ (1.4)

dengan: n = cacah angka pada bagian bulat,
m = cacah angka pada bagian pecahan.

Contoh 1.

Tentukan komplemen R dari pada bilangan-bilangan berikut:

a. $(345)_{10}$ b. $(327,15)_{10}$ c. $(10110)_2$
d. $(1101,01)_2$ e. $(320)_{16}$ f. $(A53,2)_{16}$

Penyelesaian :

a. Komplemen 10: $(345)_{c,10} = 10^3 - 345 = 1000 - 345 = 655$
Komplemen 9: $(345)_{c,9} = 10^3 - 10^0 - 345 = 654$

b. Komplemen 10: $(327,15)_{c,10} = 10^3 - 327,15$
$= 1000,00 - 327,15 = 672,85$
Komplemen 9: $(327,15)_{c,9} = 10^3 - 10^{-2} - 327,15$
$= 1000,00 - 0,01 - 327,15 = 672,84$

c. Komplemen 2: $(10110)_{c,2} = 2^5 - (10110)_2$

$= (100000)_2 - (10110)_2 = (01010)_2$

Komplemen 1: $(10110)_{c,1} = 2^5 - 2^0 - (10110)_2$
$= (100000)_2 - (00001)_2 - (10110)_2$
$= (01001)_2$

d. Komplemen 2: $(1101,01)_{c,2} = 2^4 - (1101,01)_2 = (10000,00)_2$
$\underline{-\ (\ 1101,01)_2}$
$(\ 0010,11)_2$

Komplemen 1: $(1101,01)_{c,1} = 2^5 - 2^0 - (1101,01)_2 = (10000,00)_2$
$\underline{-\ (\quad 0,01)_2}$
$(\ 1111,11)_2$
$\underline{-\ (\ 1101,01)_2}$
$(\ 0010,10)_2$

e. Komplemen 16: $(320)_{c,16} = 16^3 - (320)_{16}$
$= (1000)_{16} - (320)_{16} = (CE0)_{16}$

Komplemen 15: $(320)_{c,16} = 16^3 - 16^0 - (320)_{16}$
$= (1000)_{16} - (001)_{16} - (320)_{16} = (CDF)_{16}$

f. Komplemen 16: $(A53,2)_{c,16} = 16^3 - (A53,2)_{16}$
$= (1000,0)_{16} - (320,0)_{16} = (5AC,E)_{16}$

Komplemen 15: $(A53,2)_{c,15} = 16^3 - 16^{-1} - (A53,2)_{16}$
$= (1000,0)_{16}$
$\underline{-\ (0000,1)_{16}}$
$(\ FFF,F)_{16}$
$\underline{-\ (\ A53,2)_{16}}$
$(5AC,D)_{16}$

Dari definisi dan contoh-contoh di atas dapat dilihat bahwa komplemen R-1 dari suatu bilangan dapat diperoleh dengan mengurangi angka terbesar dengan setiap angka dalam bilangan yang bersangkutan, sedangkan komplemen R dapat diperoleh dengan menambahkan 1 ke angka paling kanan dalam komplemen R-1

Contoh 2.

Dari contoh 1 di atas dapat dilihat bahwa:

$(345)_{c,9} = \quad 654 \qquad (9\text{-}3= 6,\quad 9\text{-}4= 5,\quad 9\text{-}5= 4)$
$\underline{\quad 1}$
$655 \quad = (345)c,10$

$$(327,15)c,9 = 672,84$$
$$\underline{\qquad\quad 1}$$
$$672,85 = (327,15)c,10$$

$$(10110)c,1 = (01001)2$$
$$\underline{\qquad\quad 1}$$
$$(01010)2 = (10110)c,2$$

dan seterusnya.

Sebenarnya, komplemen bilangan biner dapat diperoleh dengan sangat mudah. Komplemen 1 diperoleh dengan menggantikan setiap angka 0 menjadi 1 dan angka 1 menjadi 0. Komplemen 2 dapat diperoleh dengan menambahkan 1 kepada komplemen 1 atau kalau kita bergerak dari kanan ke kiri, biarkanlah semua angka 0 dan angka 1 paling kanan tak berubah dan semua angka yang di kiri angka 1 ini diubah dari 0 menjadi 1 dan dari 1 menjadi 0.

Contoh 3.

(a) Untuk bilangan biner 10100100
komplemen 1 adalah : 01011011
komplemen 2 adalah : 01011100

Perhatikanlah bahwa untuk komplemen 1, masing-masing bit dikomplemenkan, 0 menjadi 1 dan 1 menjadi 0, sedangkan untuk komplemen 2 kedua bit 0 di kanan dan bit 1 paling kanan tidak diubah sedangkan bit di kiri bit 1 paling kanan ini dikomplemenkan masing-masing bitnya. Hal ini juga berlaku walaupun bilangan biner itu mempunyai bagian pecahan, seperti pada contoh (b) berikut ini.

(b) Untuk bilangan biner 10100,101

Komplemen 2 adalah: 01011,011
komplemen 1 adalah : 01011,010

1.5 Pengurangan Dengan Komplemen

Di bagian depan telah diterangkan bahwa tujuan pemakaian komplemen adalah untuk melaksanakan pengurangan dengan penjumlahan. Hal ini dapat dilakukan dalam setiap sistem bilangan. Karena pengurangan dalam sistem bilangan desimal dapat dilakukan dengan mudah kalau memakai alat-alat tulis, pengurangan dengan komplemen tidak memberikan keuntungan. Tetapi, dalam sistem elektronik digital cara pengurangan dengan komplemen ini sangat

penting, dan semua sistem digital memakai cara ini. Ini penting karena pengubahan bilangan biner menjadi komplemennya dapat dilakukan dengan mudah dan karena peranti keras (hard ware) untuk penjumlahan dan pengurangan dapat menggunakan komponen yang sama sehingga harga akan lebih murah.

1.5.1 Pengurangan dengan komplemen R

Pengurangan dengan komplemen R (komplemen 10 dalam sistem bilangan desimal, komplemen 2 dalam biner) untuk dua bilangan dapat dilakukan sebagai berikut:

Sebutlah yang dikurangi sebagai M dan pengurang sebagai N. Untuk menghitung M - N, nyatakan N sebagai komplemen R-nya dan tambahkan ke M. Bila ada "end carry" (penambahan angka di kiri) pada penjumlahan itu, maka angka tambahan tidak dipakai (dibuang saja). Bila tidak ada "end carry" ini berarti bahwa hasil pengurangan (yang dilakukan dengan penjumlahan) itu adalah negatif. Untuk hal terakhir ini, harga hasil sebenarnya adalah negatif dari pada komplemen hasil penjumlahan itu.

Contoh 1. Pengurangan dengan komplemen 10 untuk desimal.

25643 - 13674: M = 25643 25643
N = 13674 komplemen 10 = 86326
+
end carry → 1 11969
end carry, dibuang : ← 11969

10023 - 13674: M = 10023 → 10023
N = 13674, komplemen 10 = 86326
+
96349

Karena tidak ada end carry, hasil ini dikomplemenkan, sehingga hasil sebenarnya adalah - 03651.

Contoh 2.

Pengurangan dengan komplemen 2 untuk biner.

100100 - 100010 : M= 100100, N= 100010
dan -N= $(100010)_{c,2}$ = 011110

Maka hasil pengurangan adalah :

```
                                     100100
                                     011110
                                   + ------
end carry ---------------------> [1] 000010
end carry, dibuang :      <-----┘    000010
```

100100 - 101100 :

Karena $(101100)_{c,2}$ = 010100, maka penjumlahan menghasilkan :

```
  100100
  010100
+ ------
  111000
```

Karena tidak ada end carry, harga sebenarnya adalah negatif dari 111000, yaitu: -001000.

1.5.2 Pengurangan dengan komplemen R-1

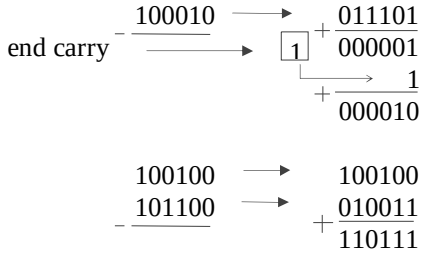
Seperti pada pengurangan dengan komplemen R, pada pengurangan dengan komplemen R-1 juga pengurang N dinyatakan dalam komplemennya, yaitu komplemen R-1. Harga komplemen ini ditambahkan ke bilangan yang dikurangi M. Perbedaan pelaksanaannya dengan pengurangan dengan komplemen R adalah penangganan end-carry. Kalau pada pengurangan dengan komplemen R end-carry itu dibuang, maka pada komplemen R-1 end carry itu ditambahkan ke angka yang paling kanan hasil penambahan. Penanganan carry seperti ini disebut "end carry-around carry".

Contoh 1. Desimal: 25643 - 13674: 10023 - 13674:

```
            25643 ---->  25643            10023 ----> 10023
          - 13674 ---->  86325          - 13674 ----> 86325
                       + -----                      + -----
end carry ----> [1]      11968                        96348
                 └---->      1                      (negatif)
                       + -----
                         11969     Komplemen-9 =    - 03651
```

Contoh 2.

Biner : 100100 ----> 100100



Negatif komplemen-1: - 001000

Dari uraian di atas dapat dilihat bahwa pengubahan suatu bilangan ke komplemen R-1 lebih mudah dibandingkan dengan pengubahan ke komplemen R. Tetapi dalam pelaksanaan penjumlahan, komplemen R-1 membutuhkan dua kali penjumlahan bila ada "end carry", sedangkan dalam komplemen R end carry diabaikan/ dibuang saja tanpa perlu dijumlahkan lagi. Disamping itu, dalam penyajian dengan komplemen R-1 ada dua harga 0, yaitu +0 dan -0, sedangkan dalam komplemen R hanya ada satu 0. Hal ini dapat ditunjukkan dengan pengurangan suatu bilangan dengan bilangan itu sendiri. Sebagai contoh, hasil 1011 - 1011 adalah:

Komplemen 2: 1011
0101
\+ ——
0000 (end carry dibuang)

Komplemen 1: 1011
0100
\+ —— (tak ada end carry, negatif)
1111

Dalam perhitungan Aljabar, adanya dua harga nol ini dapat membingungkan, terutama bila tanda dipakai untuk menentukan langkah proses selanjutnya. Namun demikian, karena mudahnya pengubahan ke komplemen 1 dalam biner, penyajian dalam komplemen 1 masih juga dipakai.

1.6 Pengurangan dalam Komputer Digital

Setiap satuan data dalam komputer digital disajikan/dinyatakan dengan sederetan angka-angka biner dengan panjang yang tertentu. Penyajian yang paling

umum adalah dengan panjang deretan yang merupakan kelipatan 4 atau 8 seperti 4, 8, 16, 36, atau 64 bit. Panjang deretan yang membentuk satu kesatuan data ini sering disebut "panjang kata" (word length).

Untuk data yang bersifat bilangan, setiap kata mempunyai bit tanda yang biasanya digunakan bit yang paling tinggi nilainya (Most Significant Bit, disingkat MSB), yaitu bit paling kiri. Untuk bilangan yang positif, umumnya bit tanda berharga 0, sedangkan untuk bilangan negatif bit tanda ini berharga 1. Bilangan negatif dapat disajikan dalam 3 cara, yaitu :

1. Dalam bentuk harga mutlak/magnitude dengan tanda (signed magnitude).
2. Dalam bentuk komplemen 1.
3. Dalam bentuk komplemen 2.

Dalam penyajian dalam bentuk harga mutlak dengan tanda, harga data yang sebenarnya dapat dilihat langsung dari bagian harga mutlaknya dan bit tanda. Operasi pengurangan dalam penyajian ini dilakukan seperti biasa dan tanda hasilnya ditentukan dengan membandingkan harga mutlak dari bilangan pengurang terhadap yang dikurangi. Jadi, bit tanda diperlakukan secara terpisah. Dibandingkan dua cara penyajian lainnya, penyajian ini lebih jarang dipakai dalam komputer kini.

Penyajian dalam komplemen tidak memperlakukan bit tanda terpisah dari bit-bit harga mutlak. Harga mutlak sebenarnya tergantung dari harga bit tanda. Setiap data bilangan negatif mempunyai bit tanda 1 dan untuk mengetahui harga mutlaknya, bilangan itu harus dikomplemenkan secara keseluruhan. Tetapi harga mutlak bilangan positif segera dapat dilihat dari penyajian biner bilangan itu. Sebagai contoh, untuk menyatakan bilangan desimal -45 dalam biner 8 bit, pertama harus dicari setara 45 dalam biner, baru dikomplemenkan. Harga biner 45 disajikan dalam 8 bit adalah 0010 1101. Maka -45 adalah 1101 0010 dalam komplemen 1 dan 1101 0011 dalam komplemen 2.

Karena panjang kata dalam setiap komputer sudah tertentu maka dalam melakukan pengurangan dalam komplemen, semua bit sebelah kiri yang berharga 0 pun harus ditunjukkan secara lengkap, tak boleh hanya memperhatikan bit-bit yang di sebelah kanan bit 1 paling kiri. Sebagai contoh, untuk mengurangkan 17 - 5 dalam biner, maka pengurangan harus dilakukan sebagai berikut :

$$
\begin{array}{rrl}
\text{Komplemen 1 : } 0001\ 0001 & \longrightarrow & 0001\ 0001 \\
0000\ 0101 & \longrightarrow & +\ \underline{1111\ 1010} \\
\text{end carry around} & \longrightarrow & \quad 0000\ 1011 \\
 & & +\ \underline{\qquad\quad 1} \\
 & & \quad 0000\ 1100
\end{array}
$$

Komplemen 2 : 0001 0001 → 0001 0001
0000 0101 → + 1111 1011
0000 1100
end carry dibuang → 0000 1100

Kalau seandainya kedua operannya tidak dinyatakan secara lengkap, maka akan diperoleh :

17 → 10001 dapat membawa kepada 10001
− 5 − 101 + 011
10100

Ini jelas salah. Kesalahan ini sebenarnya dapat segera dilihat bila diperhatikan bahwa hasil 10001-101 bertanda negatif (bit paling kiri berharga 1).

Dalam melihat harga sebenarnya daripada hasil pengurangan, perhatikan contoh berikut ini.

0000 0101	0000 0101	0000 0101
− 0101 1100	+ 1001 0011	+ 1001 0100
	1001 1000	1001 1001
Harga sebenarnya :	- (0110 0111)	- (0110 0111)
	(komplemen 1)	(komplemen 2)

1.7 Penyajian Data

Seperti diterangkan di bagian depan, setiap sinyal diskrit dapat dinya-takan sebagai kombinasi dari sejumlah angka biner (bit). Penyataan ini berarti pengubahan suatu bentuk informasi kebentuk yang lain dengan pengkodean yang terdiri atas sekelompok biner yang merupakan satu kesatuan. Pengelompokkan yang paling banyak dilakukan adalah pengelompokan atas kelipatan 4 bit. Kode yang terdiri atas 4 bit disebut "Nibble", kelompok yang terdiri atas 8 bit disebut "byte", dan kelompok terdiri dari 16 bit (2 byte) disebut "word". Word yang terdiri atas lebih dari 2 byte sering disebut long word. Berikut ini diuraikan secara singkat beberapa jenis kode yang sering dipergunakan dalam teknik digital.

1.7.1 Kode BCD

Seperti telah diterangkan dalam uraian mengenai sistem bilangan oktal dan heksadesimal di bagian depan, untuk menyatakan 1 angka desimal diperlukan 4

angka biner. Tetapi dengan 4 bit sebenarnya dapat dinya-takan 16 macam simbol yang berbeda sehingga kesepuluh simbol dalam bilangan desimal dapat dinyatakan dengan beberapa himpunan (set) kode yang berbeda. Perlu dibedakan dengan tegas antara pengkodean dan konversi. Kalau suatu bilangan dikonversikan ke bilangan lain maka kedua bilangan itu mempunyai harga/nilai. Sebagai contoh, kalau angka 8 desimal dikonversikan ke biner, maka satu-satunya pilihan adalah 1000. Tetapi kalau angka 8 ini dikodekan ke biner, ada bermacam-macam kode yang dapat dibentuk, walaupun hanya terdiri atas 4 bit. Dari bermacam-macam kode untuk angka-angka desimal, kode BCD (singkatan dari Binary Coded Decimal) merupakan kode yang paling sederhana karena kode itu sendiri merupakan konversi dari desimal ke biner.

Setiap bit dalam BCD diberi bobot menurut letaknya dalam urutan kode sesuai dengan rumus (1.2) di depan, yaitu 1, 2, 4, dan 8, berurut dari bit yang paling kanan. Jadi, untuk angka 9, yaitu 8 + 1, kode BCD-nya adalah: 1001; untuk angka 6 yaitu 4 + 2, kodenya adalah: 0110. Kode-kode 1010, 1100, 1011, 1100, 1101, 1110, dan 1111 tidak ada didalam BCD karena nilai kode-kode ini sudah lebih dari 9. Kode-kode BCD yang lengkap ditunjukkan pada Tabel 1.1.

Setiap angka desimal dikodekan dengan satu BCD yang empat bit. Karena itu, untuk menyatakan bilangan desimal ratusan diperlukan 3 kode BCD, jadi 12 bit. Sebagai contoh, bilangan 163 dikodekan dengan 0001 0110 0011.

Seperti yang ditunjukkan pada Tabel 1.1, bobot bit pada setiap posisi dapat dibuat berbeda-beda. Keuntungan kode BCD standar (8421) terletak pada kenyataan bahwa kode itu merupakan konversi langsung dari bit ke angka desimal. Dengan memberi bobot yang lain dapat diperoleh keuntungan berupa simetri atau sifat komplemen. Sebagai contoh, kode dengan bobot 2421 dan 84-2-1 mempunyai sifat mengkomplemenkan sendiri (self complementing). Perhatikan bahwa komplemen 3 adalah 6 dan dalam kode 84-2-1 ini ditunjukkan dengan 1010 (=6) yang merupakan komplemen dari 0101 (=3).

Tabel 1.1. Kode-kode untuk angka-angka desimal

Desi-mal	BCD* 8,4,2,1	Excess-3 (XS3)	Gray	8,4,-2,-1	2,4,2,1
0	0000	0011	0000	0000	0000
1	0001	0100	0001	0111	0001
2	0010	0101	0011	0110	0010
3	0011	0110	0010	0101	0011
4	0100	0111	0110	0100	0100
5	0101	1000	1110	1011	1011
6	0110	1001	1010	1010	1100

7	0111	1010	1011	1001	1101
8	1000	1011	1001	1000	1110
9	1001	1100	1000	1111	1111

* Kode dengan bobot 8421 dianggap sebagai kode BCD standar.

1.7.2 Kode Excess-3 (XS3)

Seperti dapat dilihat dari Tabel 1.1, kode Excess-3 (XS3) diperoleh dengan menambahkan 3 (=0011) kepada kode BCD standar, dan inilah alasan pemberian namanya. Tetapi dengan penambahan ini diperoleh sifat bahwa komplemen dalam kode XS3 juga menghasilakan komplemen dalam desimal. Sebagai contoh, komplemen 0100 (= 1 dalam desimal) adalah 1011 (= 8 dalam desimal) dan dalam desimal 1 adalah 8. Watak mengkom-plemenkan sendiri (self complementing) ini sangat berguna dalam komputer yang menggunakan kode BCD dalam perhitungannya sebab rangkaian elektronik komplemennya menjadi sederhana.

1.7.3 Kode Gray

Dalam kode Gray, setengah bagian atas, yaitu untuk kode desimal 5-9, merupakan bayangan cermin dari pada setengah bagian bawah, yaitu kode untuk desimal 0-4, kecuali untuk bit 3 (bit ke 4 dari kanan). Sifat ini disebut reflective. Di samping itu, seperti dapat dilihat pada Tabel 1.1 di depan, kode Gray juga mempunyai sifat bahwa kode untuk desimal yang berturutan berbeda hanya pada 1 bit. Sifat ini sangat penting dalam pengubahan sinyal-sinyal mekanis atau listrik ke bentuk digital. Sebagai contoh, kalau tegangan yang dikenakan pada suatu voltmeter digital berubah dari 3 volt ke 4 volt (dalam biner dari 0011 ke 0100), maka ada kemungkinan bit 2 (bit ke 3 dari kanan) akan berubah lebih dulu dari bit-bit yang lain sehingga akan memberikan penunjukan sementara 0111 (= 7) yang jelas salah. Dengan penggunaan kode Gray kesalahan seperti ini tidak akan terjadi.

1.7.4 Kode penunjuk kesalahan

Dalam hubungan antar satu komputer dengan yang lain, sering terjadi perbedaan antara sinyal yang dikirim dan sinyal yang diterima. Ini terjadi karena adanya gangguan (noise) yang timbul pada saluran komunikasinya. Untuk mengetahui adanya kesalahan itu sering ditambahkan satu bit tambahan kepada

kode sinyal aslinya. Bit tambahan ini disebut bit parity. Dengan penambahan bit parity ini, maka kesalahan satu bit dalam setiap kode yang merupakan kesatuan dapat diketahui/dideteksi. Bit parity biasanya ditambahkan pada saat pengiriman dan dibuang kembali di sisi penerimaan sebelum diproses. Perlu dicatat bahwa bit parity ini hanyalah menunjukkan adanya kesalahan, bukan membetulkan kesalahan itu.

Dalam pemakaian bit parity dikenal dua macam cara: parity genap (even) dan parity ganjil (odd). Dalam sistem parity ganjil, cacah bit 1 harus selalu ganjil. Bila dalam sistem ini diterima suatu kode dengan cacah bit 1 yang genap, ini berarti telah terjadi kesalahan dalam pengiriman. Dalam sistem parity genap cacah bit 1 dalam setiap unit kode harus tetap genap. Bila dalam sistem ini diterima diterima suatu satuan kode dengan cacah bit 1 yang ganjil, maka suatu kesalahan telah terjadi dalam transmisi. Sebagai contoh, untuk kode-kode BCD standar di depan, satu angka desimal akan dikirimkan sebagai satuan yang terdiri atas 5 bit setelah ditambahkan satu bit parity, biasanya pada posisi nilai tertinggi (di kiri). Untuk kode-kode desimal 5 dan 8, yang kode sebenarnya adalah 0101 dan 1000, dalam sistem parity ganjil akan dikirimkan sebagai 10101 dan 01000, sedangkan pada sistem parity genap kode-kode tersebut akan dikirimkan sebagai 00101 dan 11000.

1.7.5 Kode Alfanumerik

Dalam penggunaan komputer secara umum, walaupun kode yang diolah dalam komputer itu sendiri adalah angka-angka biner, tetapi selain angka-angka desimal juga diproses huruf-huruf dan tanda-tanda baca/tanda khusus lainnya. Untuk memroses data seperti ini tentunya diperlukan sistem kode yang lebih luas dari pada sistem-sistem kode yang telah diterangkan sebelumnya. Kode yang berlaku umum ini disebut kode "Alphanumeric" yang sering juga disingkat dengan nama "Alphameric". Dua jenis kode yang paling umum dipakai dalam dunia komputer sekarang ini adalah: ASCII (baca: eskii, singkatan dari: American Standard Code for Information Interchange) dan EBCDIC (baca: ebsidik, singkatan dari: Extended Binary Coded Decimal Interchange Code).

ASCII terdiri atas 7 bit yang dapat mengkodekan semua angka desimal, huruf abjad, baik huruf besar maupun kecil, tanda-tanda khusus dan tanda baca, dan beberapa kode kendali/kontrol yang umum dipakai dalam komunikasi data. Dalam praktek sekarang, walaupun aslinya 7 bit, kebanyakan ASCII menggunakan 8 bit dengan bit tambahan dipakai sebagai bit parity, kadang-kadang untuk membentuk aksara yang bukan aksara latin.

Sistem kode EBCDIC terdiri atas 8 bit, digunakan dalam komputer-komputer IBM tipe 360 dan 370 yang sangat terkenal itu.

Dalam perekaman data pada kartu tebuk (puch card), data alfanumerik dikodekan dengan menggunakan kode Hollerith standar yang terdiri atas 12 bit. Kartu tebuk standar terdiri atas 80 kolom yang terdiri atas 12 baris tebukan yang dibedakan atas 2 kelompok, yaitu baris 12, 11, dan 0 di bagian atas disebut sebagai ***zone*** dan baris 9, 8, 7,.., 1 di bagian bawah disebut baris ***numeric***.

Dalam **Tabel 1.2** ditunjukkan ketiga jenis kode Alfanumerik yang disebut di atas. Bilangan yang ditunjukkan dalam kolom kode Hollerith dalam tabel ini menunjukkan nomor baris yang ditebuk/dilubangi sedangkan posisi yang tidak ditunjukkan berarti kosong. Perhatikan bahwa kode-kode EBCDIC sangat erat hubungannya dengan kode Hollerith. Terutama dalam kode huruf, naiknya satu harga angka heksadesimal pertama pada kode EBCDIC setara dengan turunnya satu baris lubang pada kode Hollerith. Juga perhatikan bahwa dalam kode Hollerith, angka dinyatakan dengan 1 lubang, huruf dengan 2 lubang sedangkan tanda lain dari 1, 2, atau 3 lubang pada kolom yang sama. Dalam EBCDIC, untuk 4 bit paling kiri, angka dinyatakan dengan 1111 (F heksadesimal), huruf kapital dinyatakan dengan C s/d E dan untuk huruf kecil dinyatakan dengan angka heksadesimal 8 s/d A, sedang tanda lain dinyatakan dengan 01xx, dengan x dapat berarti 0 atau 1. Dalam ASCII, karakter dengan kode dibawah 20 heksadesimal digunakan sebagai kode kendali komunikasi, angka dikodekan dengan 30h - 39h, huruf kapital dikodekan 41h - 5Ah, huruf kecil 61h - 7Ah dan kode yang lainnya untuk tanda-tanda baca. Jelaslah bahwa kode ASCII lebih mudah untuk diingat.

Tabel 1.2. Kode Alfanumerik ASCII, EBCDIC, dan Hollerith

Tanda	ASCII	EBCDIC	Kartu
NUL	00	00	12,0,9,8,1
SOH	01	01	12, 9, 1
STX	02	02	12, 9, 2
ETX	03	03	12, 9, 3
BOT	04	37	9,7
ENQ	05	2D	0, 9,8,5

Tanda	ASCII	EBCDIC	Kartu
blank	20	40	no punch
!	21	5A	12,8,7
"	22	7F	8,7
#	23	7B	8,3
$	24	5B	11,8,3
%	25	6C	0,8,4

ACK	06	2E	0, 9,8,6
BEL	07	2F	0,9,8,7
BS	08	16	11,9,4
HT	09	05	11,9,5
LF	0A	25	0,9,5
VT	0B	0B	12,9,8,3
FF	0C	0C	12,9,8,4
CR	0D	0D	12,9,8,5
S0	0E	0E	12,9,8,6
S1	0F	0F	12,9,8,7
DLE	10	10	12,11,9,8,1
DC1	11	11	11,9,1
DC2	12	12	11,9,2
DC3	13	13	11,9,3
DC4	14	35	9,8,4
NAK	15	3D	9,8,5
SYN	16	32	9,2
ETB	17	26	0,9,6
CAN	18	18	11,9,8
EM	19	19	11,9,8,1
SUB	1A	3F	9,8,7
ESC	1B	24	0,9,7
FS	1C	1C	11,9,8,4
GS	1D	1D	11,9,8,5
RS	1E	1E	11,9,8,6
US	1F	1F	11,9,8,7

&	26	50	12
'	27	7D	8,5
(	28	4D	12,8,5
)	29	5D	11,8,5
*	2A	5C	11,8,4
+	2B	4E	12,8,6
,	2C	6B	0,8,3
-	2D	60	11
.	2E	4B	12,8,3
/	2F	61	0,1
0	30	F0	0
1	31	F1	1
2	32	F2	2
3	33	F3	3
4	34	F4	4
5	35	F5	5
6	36	F6	6
7	37	F7	7
8	38	F8	8
9	39	F9	9
:	3A	7A	8,2
;	3B	5E	11,8,6
<	3C	4C	12,8,4
=	3D	7E	8,6
>	3E	6E	0,8,6
?	3F	6F	0,8,7
@	40	7C	8,4

Tabel 1.2. Kode Alfanumerik (Lanjutan)

Tanda	ASCII	EBCDIC	Kartu
A	41	C1	12,1
B	42	C2	12,2
C	43	C3	12,3
D	44	C4	12,4
E	45	C5	12,5
F	46	C6	12,6

Tanda	ASCII	EBCDIC	Kartu
a	61	81	12,0,1
b	62	82	12,0,2
c	63	83	12,0,3
d	64	84	12,0,4
e	65	85	12,0,5
f	66	86	12,0,6

G	47	C7	12,7
H	48	C8	12,8
I	49	C9	12,9
J	4A	D1	11,1
K	4B	D2	11,2
L	4C	D3	11,3
M	4D	D4	11,4
N	4E	D5	11,5
O	4F	6	11,6
P	50	D7	11,7
Q	51	D8	11,8
R	52	D9	11,9
S	53	E2	0,2
T	54	E3	0,3
U	55	E4	0,4
V	56	E5	0,5
W	57	E6	0,6
X	58	E7	0,7
Y	59	E8	0,8
Z	5A	E9	0,9
[	5B	AD	12,8,2
\	5C	15	0,8,2
]	5D	DD	11,8,2
^	5E	5F	11,8,7
_	5F	6D	0,8,5
‘	60	14	8,1

g	67	87	12,0,7
h	68	88	12,0,8
i	69	89	12,0,9
j	6A	91	12,11,1
k	6B	92	12,11,2
l	6C	93	12,11,3
m	6D	94	12,11,4
n	6E	95	12,11,5
o	6F	96	12,11,6
p	70	97	12,11,7
q	71	98	12,11,8
r	72	99	12,11,9
s	73	A2	11,0,2
t	74	A3	11,0,3
u	75	A4	11,0,4
v	76	A5	11,0,5
w	77	A6	11,0,6
x	78	A7	11,0,7
y	79	A8	11,0,8
z	7A	A9	11,0,9
(	7B	8B	12,0
|	7C	4F	12,11
)	7D	9B	11,0
~	7E	4A	11,0,1
DEL	7F	07	12,9,7

*) ASCII dan EBCDIC ditulis dalam kode Hexadecimal

1.8 Soal Latihan

1. Nyatakanlah bilangan-bilangan desimal berikut dalam sistem bilangan:
   a. Biner, b. Oktal, c. Heksadesimal.

   5 11 38 1075 35001 0.35 3.625 4.33

2. Tentukanlah kompelemen 1 dan kompelemen 2 dari bilangan biner berikut:

   1010 1101 11010100 1001001

3. Tentukanlah kompelemen 9 dan kompelemen 10 dari bilangan desimal berikut:

   21 139 2400 9101

4. Tentukanlah kompelemen 7 dan kompelemen 8 dari bilangan Oktal berikut:

   21 137 320 161

5. Tentukanlah kompelemen 15 dan kompelemen 16 dari bilangan Heksadesimal:

   BAC B3F 120 1A1

6. Dengan panjang kata 8 bit dan bit paling kiri menyatakan tanda, 0= positif dan 1= negatif, nyatakanlah bilangan-bilangan desimal berikut dalam biner dengan menggunakan kompelemen 1 dan kompelemen 2:

   7 -11 -27

7. Dalam sistem yang menggunakan ukuran kata 16 bit, tentukanlah harga desimal dari bilangan-bilangan berikut:

   Biner : 0100 1101 1100 1000; 1011 0100 1010 0101

   Oktal : 73 ; 201 ; 172

   Heksadesimal: 6B ; A5 ; 7C

8. Dengan melakukan operasi penjumlahan, laksanakan pengurangan berikut:

   Desimal: 125 - 32; 15 - 72

   Biner : 1001 - 1000; 1001 - 1110

   (panjang kata 8 bit)

9. Nyatakanlah bilangan desimal berikut dalam kode-kode BCD, Gray dan Excess-3:

   51 125 0234

10. Tentukanlah hasil penjumlahan dalam kode BCD berikut:

    52 + 19 125 + 93 59 + 45

11. Tuliskanlah kode ASCII dan EBCDIC, baik secara biner maupun heksadesimal, larik : "Kodya Medan (SUMUT)".