情報技術者試験受験テキスト

基礎理論

離散数学

令和元年８月発行

ＫＭＣ学習所

txt0101 離散数学目次

txt0101 離散数学

txt01011 数値の表現

txt010111 基数変換

① 基数とは

ⓐ 10進数

１桁の数字を表わすために、０、１、２、…、８、９の10個の数字を使用し、10を数えると、位取りを１桁上げる表現法である。

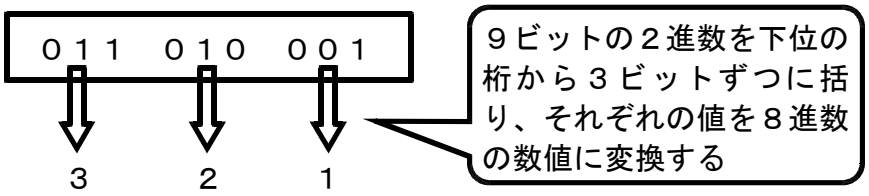
ⓑ ２進数

１桁の数字を表わすために、０、１の２個の数字を使用し、２を数えると、位取りを１桁上げる表現法である。

ⓒ ８進数

１桁の数字を表わすために、０、１、…、６、７の８個の数字を使用し、８を数えると、位取りを１桁上げる表現法である。

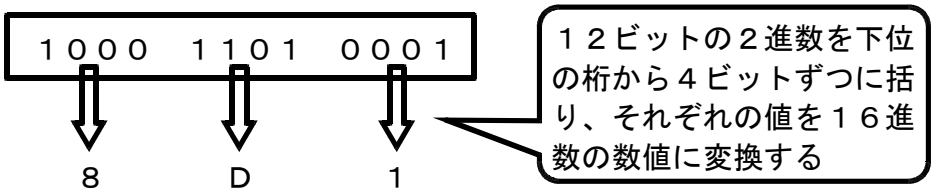
２進数で表現すると、１桁を３ビットで表すことができる。



ⓓ 16進数

１桁の数字を表わすために、０、１、…、８、９の10個の数字と、Ａ～Ｆの６個の英字を使用し、16を数えると、位取りを１桁上げる表現法である。

２進数で表現すると、１桁を４ビットで表すことができる。



ⓔ　ｎ進数

１桁の数字を表わすために、０、１、…、８、ｎ－１のｎ個の数字を使用し、ｎを数えると、位取りを１桁上げる表現法である。ただし、ｎが10を超える場合、１桁の数字を表すためにＡ、Ｂ、…、の英字を使用する。

例えば、ｎが17の場合、１桁の数字に０、１、２、…、８、９の10個の数字と、Ａ、Ｂ、…、Ｆ、Ｇの７個の英字を使用し、17を数えると、位取りを１桁上げる表現法となる。

②　基数変換の手順

ⓐ　10進数の整数を２進数に変換する手順

㋐　10進数を２で割り、商と余りを求める。

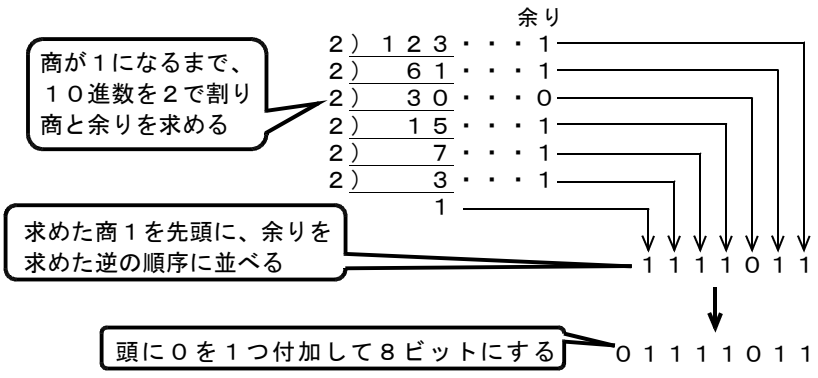
㋑　商が１になるまで、商と余りを求める演算を繰り返す。

㋒　最後の商１を先頭に、求めた余りを求めた順序とは逆に並べていく。

㋓　ビット数を調整するために先頭に０を必要個数付加する。

具体例

10進数の整数123を８ビットの２進数に変換する。



変換手順

㋐　整数123を２で割り、商61と余り１を求める。次に、商61を２で割り、商30と余り１を求め、更に、商30を２で割り、商15と余り０を求める。

㋑　以下順次、同様の操作を商が１になるまで繰り返して、最後に余り１を求める。

㋒　演算結果の余りを求めた順に並べ最後の桁に商１を付加して1101111の７ビットの２進数を

得る。

㋓　この７ビットの２進数の並びを反転すると、1111011となる。

㋔　ビット数を８にするために先頭に０を付加して、求める答え01111011を得る。

ⓑ　10進数の小数を２進数に変換する手順

㋐　10進数の小数に２を掛ける。

㋑　整数部分が１になれば１、０ならば０を小数１位から順次、各桁の値とする。

㋒　整数部が１の場合、小数部のみに対して㋐～㋑の操作を繰り返す。整数部が０の場合、そのまま小数部に㋐～㋑の操作を繰り返す。

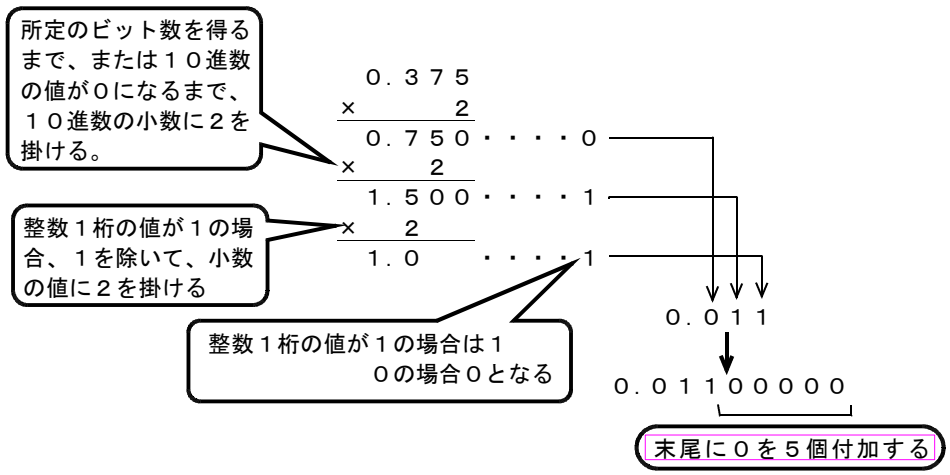
㋓　小数部が０になるか、所定の桁数になれば、㋐～㋒の操作を打ち切る。

㋔　小数部が０になり打ち切った場合、小数点以下の不足した桁数には末尾に０を付加する。

具体例

10進数の小数0.375を８ビットの２進数に変換する。

変換手順



㋐　小数0.375を２倍して、0.750を得る。整数部分の値が０であるから、小数１桁目の値は０となる。

㋑　次に、前に求めた結果0.750を２倍して、1.50を得る。整数部分の値が１であるから、小数２桁目の値は１となる。

㋒　更に、一つ前の演算結果1.50の整数部分の１を除いた小数0.5を２倍して、1.0をえる。整

数部分の値が１であるから、小数３桁目の値は１となる。

㊓　次に、一つ前の演算結果1.0の整数部分の１を除いた小数の値は０となったから演算を打ち切る。

㊔　変換後の２進数の値は0.011となる。

㊕　ビット数を８ビットに調整するために、小数の後方の桁に５ビット０を付加して、0.01100000の小数を得る。

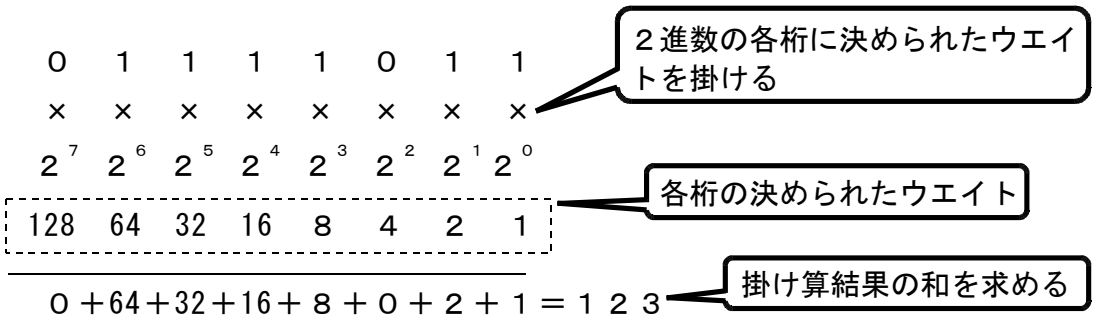
ⓒ　２進数整数を10進数に変換する手順

㋐　下位の桁から順次、2^0、2^1、2^2、2^3、…、のウェイトを掛ける。

㋑　掛け算の結果の和を求める。

具体例

２進数の整数01111011を10進数に変換する。



㋐　下位の桁から順次、2^0、2^1、2^2、2^3、…、のウェイトを掛ける。
上図に示す掛け算の結果、０、64、32、16、８、０、２、１を得る。

㋑　掛け算の結果の和を求める。
０＋64＋32＋16＋８＋０＋２＋１＝123

ⓓ　２進数小数を10進数に変換する手順

㋐　小数１位の桁から順次、2^{-1}、2^{-2}、2^{-3}、…、のウェイトを掛ける。

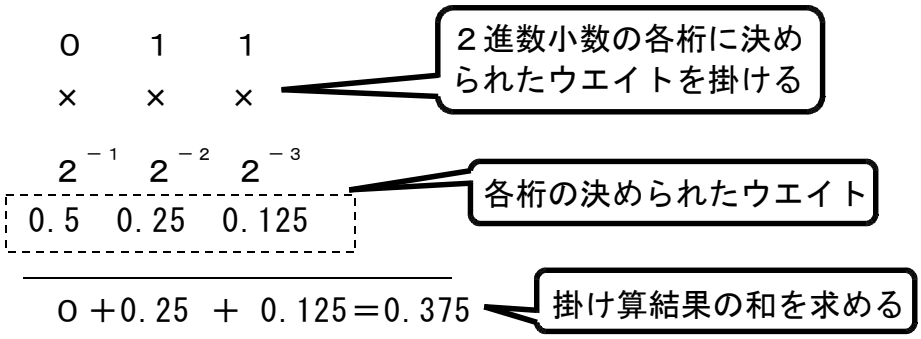
㋑　掛け算の結果の和を求める。

具体例

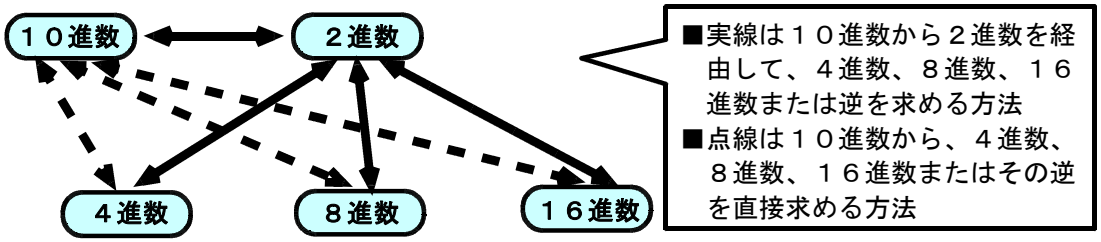
２進数の小数0.011を10進数に変換する例を次に示す。仮想小数点が左端にあると考える。

㋐　小数１位の桁から順次、2^{-1}、2^{-2}、2^{-3}、…、のウェイトを掛ける。
上図に示す掛け算の結果、０、0.25、0.125を得る。

㋑　掛け算の結果の和を求める。
０＋0.25＋0.125＝0.375



③　２進数、８進数、10進数、16進数間の基数変換



ⓐ　２進数から16進数への変換手順

㋐　小数点を起点にして、整数部分は上位桁に向かって４ビット単位に括る。小数部分は下位桁に向かって４ビット単位に括る。

㋑　４ビットの２進数を16進数に変換する。

ⓑ　16進数から２進数への変換手順

㋐　各桁の16進数を４ビットの２進数に変換する。

㋑　16進数の上位桁から順次４ビットの２進数を並べる。

㋒　所定のビット数に調整する。

ⓒ　２進数から８進数への変換手順

㋐　小数点を起点にして、整数部分は上位桁に向かって３ビット単位に括る。小数部分は下位桁に向かって３ビット単位に括る。

㋑　３ビットの２進数を８進数に変換する。

ⓓ　８進数から２進数への変換手順

㋐　各桁の８進数を３ビットの２進数に変換する。

㋑　８進数の上位桁から順次３ビットの２進数を並べる。

㋒　所定のビット数に調整する。

ⓔ　10進数から８進数を求める方法

10進数を８で割って余りを求め、商が８より小さくなると、最後に求めた商を先頭に求めた逆の順序に余りを並べる。

または、10進数を２進数に変換し、２進数を８進数に変換する。

ⓕ　８進数から10進数を求める方法

８進数の各桁の上位から、順次、…、8^{2}、8^{1}、8^{0}、8^{-1}、8^{-2}、…、のウエイトを掛けて、その結果の和を求める。

ⓖ　10進数から16進数を求める方法

10進数を16で割って、余りを求め、商が16より小さくなると、最後に求めた商を先頭に求めた逆の順序に余りを並べる。

または、10進数を２進数に変換し、２進数を16進数に変換する。

ⓗ　16進数から10進数を求める方法

16進数の各桁の上位から、順次、…、16^{2}、16^{1}、16^{0}、16^{-1}、16^{-2}、…、のウエイトを掛けて、その結果の和を求める。

ⓘ　10進数からｒ進数を求める方法

10進数をｒで割って、余りを求め、商がｒより小さくなると、最後に求めた商を先頭に求めた逆の順序に余りを並べる。

または、10進数を２進数に変換し、２進数をｒ進数に変換する。

ⓙ　ｒ進数から10進数を求める方法

ｒ進数の各桁の上位から、順次、…、r^2、r^1、r^0、r^{-1}、r^{-2}、…、のウエイトを掛けて、その結果の和を求める。

ⓚ　ｍ進数からｎ進数を求める方法

㋐　ｍ進数を２進数に変換し、２進数をｎ進数に変換する。

㋑　ｍ進数を10進数に変換し、10進数をｎ進数に変換する。

㋒　ｍ＞ｎの場合、ｍをｎで割り、商と余りを求める計算を商がｎより小さくなるまで繰返し、商がｎより小さくなると、最後の商を先頭に求めた余りを最後から先頭の順に並べる。

④ ２進数、８進数、10進数、16進数の対比

２進数	８進数	10進数	16進数
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	10	A
1011	13	11	B
1100	14	12	C
1101	15	13	D
1110	16	14	E
1111	17	15	F

例題演習

１０進数の０．６８７５を２進数で表したものはどれか.

ア　０．１００１　　イ　０．１０１１
ウ　０．１１０１　　エ　０．１１１１

解答解説

１０進数の小数を２進数に変換する問題である。
０．６８７５に２を掛けると次のようになる。
０．６８７５×２＝１．３７５……１
０．３７５×２＝０．７５……０
０．７５×２＝１．５……１
０．５×２＝１．０……１
答えは０．１０１１となる。求める答えはイとなる。

例題演習

１６進数０．７５と等しいものはどれか。

ア　$2^{-2}+2^{-5}+2^{-7}+2^{-8}$　　イ　$2^{-2}+2^{-3}+2^{-4}+2^{-6}+2^{-8}$
ウ　$2^{-1}+2^{-2}$　　エ　$2^{-1}+2^{-2}+2^{-3}+2^{-4}+2^{-6}$

解答解説

１６進数の小数を１０進数に変換する問題である。
１６進数の０．７５を２進数に変換すると次のようになる。
$(0.75)_{16}=(0.01110101)_2$
小数１位から順次下位桁に2^{-1}、2^{-2}、2^{-3}、…、を乗じると、次のようになる。
$0.01110101=2^{-2}+2^{-3}+2^{-4}+2^{-6}+2^{-8}$
求める答えはイとなる。

例題演習

２進数の１０１．１１を１０進数で表したものはどれか。

ア　５．１１　　イ　５．３　　ウ　５．５５　　エ　５．７５

解答解説

２進数の小数を１０進数に変換する問題である。
整数部分を１０進数に変換すると、$(101)_2$＝４＋１＝５
小数部分を１０進数に変換すると、０．１１＝０．５＋０．２５＝０．７５
両者の和を求めると、５＋０．７５＝５．７５ となり、求める答えはエとなる。

例題演習

１０進数の分数１／３２を１６進数の小数で表したものはどれか。

ア　０．０１　　イ　０．０２　　ウ　０．０５　　エ　０．０８

解答解説

１０進数を１６進数に変換する基数変換の問題である。

１０進数の分数を２進数の小数に変換し、２進数の小数を１６進数に変換する。

$1／32＝1／2^{5}＝(0.00001000)_{2}＝(0.08)_{16}$

１６進数の値は０．０８となり、求める答えはエとなる。

例題演習

次の式は，何進法で成立するか。

１０１５÷５＝１３１（余り０）

ア　６　　イ　７　　ウ　８　　エ　９

解答解説

基数変換に関する問題である。

アの場合、１０１５を１０進数に変換すると、

$6^{3}＋6^{1}＋5＝216＋6＋5＝227$

１３１を１０進数に変換すると、

$6^{2}＋3×6＋1＝36＋18＋1＝55$

２２７÷５＝４５．４≠５５となる。

イの場合、１０１５を１０進数に変換すると、

$7^{3}＋7^{1}＋5＝343＋7＋5＝355$

１３１を１０進数に変換すると、

$7^{2}＋3×7＋1＝49＋21＋1＝71$

３５５÷５＝７１となり、右辺の値と一致する。求める答えはイとなる。

ウの場合、１０１５を１０進数に変換すると、

$8^{3}＋8^{1}＋5＝512＋8＋5＝525$

１３１を１０進数に変換すると、

$8^{2}＋3×8＋1＝64＋24＋1＝89$

５２５÷５＝１０５≠８９となる。

エの場合、１０１５を１０進数に変換すると、

$9^{3}＋9^{1}＋5＝729＋9＋5＝743$

１３１を１０進数に変換すると、

$9^{2}＋3×9＋1＝81＋27＋1＝109$

７４３÷５＝１４８．６≠１０９となる。

txt010112 補数と２進数の負数

①　補数の求め方

ⓐ　２の補数の求め方

㋐　絶対値のＬＳＢ(最も右のビット)から順に、２の補数の各ビットを㋑～㋓によって求める。

㋑　ＬＳＢから連続する０は、そのままにしておく。

㋒　最初に出現する１も、そのままにしておく。

㋓　その次のビットからは各ビットを反転させる。

ⓑ　小数を含む２進数の２の補数の求め方

㋐　小数点位置に関係なく(小数点位置はそのままで)、絶対値の最下位から順に、次の㋑～㋓によって２の補数の各桁を求める。

㋑　最下位からの０の連なり(連続する０)は、そのまま０にしておく。

㋒　最初の１の桁も、そのまま１にする。

㋓　その次のビットからは各ビットを反転させる。

ⓒ　ｒ進数の補数の求め方

㋐　小数点位置に無関係に(小数点位置はそのままで)、絶対値の最下位から順に、次の㋑～㋓によってｒの補数の各桁の値を求める。

㋑　最下位からの０の連なり(連続する０)は、そのまま０にしておく。

㋒　最初の０でない最下位桁は、ｒからその桁の値を減算し、その桁の補数を求める。

㋓　その次の上位桁からは、(ｒ－１)から各桁の値を減算し、各桁の補数を求める。

具体例

８ビットの整数および小数を含む２進数の２の補数を求める。

①　０１１１０１００の２の補数を求める。

０１１１０１００
　↓　　↓↓↓
１０００１１００

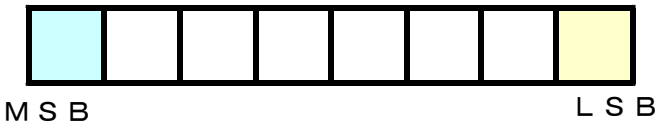
ＬＳＢから連続する０はそのまま、最初の１もそのままにし、その後の０と１を反転する

②　０１１０．１００１ の２の補数を求める。

０１１０．１００１
　　↓　　↓
１００１．０１１１

小数点の位置に関係なく、ＬＳＢから連続する０はそのまま、最初の１もそのままにし、その後の０と１を反転する

最も左のビットをＭＳＢ、最も右のビットをＬＳＢという。



ⓓ　ｒ進数のｒの補数、（ｒ－１）の補数

ある数ｎ桁のｒ進数のｒの補数は、ｎ＋１桁の1000…00からｒ進数の値を減じると求めることができる。ｒ進数のある数の（ｒ－１）の補数は、ｒの補数の最下位桁から１を減算すれば求まる。ｎ桁のｒ進数のある数の（ｒ－１）の補数の各桁は、（ｒ－１）の値から元のｒ進数の各桁の値を減算すれば求まる。

ｒ進数のある数Ｎの絶対値が同じで、符号が逆の（ｒ－１）の補数、ｒの補数は、各桁のｒ－１の補数、ｒの補数を求め、最上位の左の桁に（ｒ－１）の値を付加する（負数の符号）。

ⓔ　２進数の１の補数の求め方

２進数の１の補数は２進数の２の補数の最下位桁から１を減ずるか、または、２進数の１の補数は各桁の１から２進数の各桁の値を減ずると求めることができる。

具体例

２進数10110101の８桁の１の補数を求める。

２の補数は01001011であるから１の補数は最下位桁から１を減算すると、01001010となる。または、８桁の11111111から元の２進数の減算を行うと

11111111－10110101＝01001010

となる。

ⓕ　10進数の補数の求め方

㋐　小数点位置に無関係に(小数点位置はそのままで)、絶対値の最下位から順に、次の㋑～㋓によって10の補数の各桁を求める。

㋑　最下位からの０の連なり(連続する０)は、そのまま０にしておく。

㋒　最初の０でない最下位桁は、10からその桁の値を減算し、その桁の補数を求める。

㋓　その次の上位桁からは、９から各桁の値を減算し、各桁の補数を求める。

ⓖ　10進数の９の補数

10進数の９の補数は10の補数の最下位桁から１を減算すれば得られる。または、10進数の９の補数は、各桁の９の値から10進数の各桁の値を減算すれば得られる。

10進数Ｎの絶対値が同じで、符号が逆の９の補数、10の補数は、各桁の９の補数、10の補数を求め、最上位の左に９を付加する（負数の符号)。

具体例

10進数の642の９の補数は、999－642＝357となる。

10進数の642の10の補数は、1000－642＝358となる。

９の補数	999	10の補数	1000
	－ 642		－ 642
	357		358

10進数の64200の10の補数を上記の手順で求めると、次のようになる。

㋐　右からの２つの０はそのまま０となる。
㋑　最初の２は10－２＝８、次から９－４＝５、９－６＝３となる。
㋒　10の補数は35800となる。

10進数の64200の９の補数は35799となる。

10進数の－312.14の９の補数は9687.85であり、10の補数は0.01を加算して、9687.86となる。

②　２進数の負数

ⓐ　負数の表し方

２進数の負数は２の補数で表す。負数を２の補数で表現する２進数は、ＭＳＢは正または負の符号を表すものとする。ＭＳＢが０の場合を正または０、ＭＳＢが１の場合が負である。

ⓑ　基準値、真数、補数、負の値の関係

㋐　正の真数Ｐは、図のＰの長方形（赤斜線）、負数の真数－Ｐは基準値と反対側の－Ｐの長方形（緑縞模様）になる。

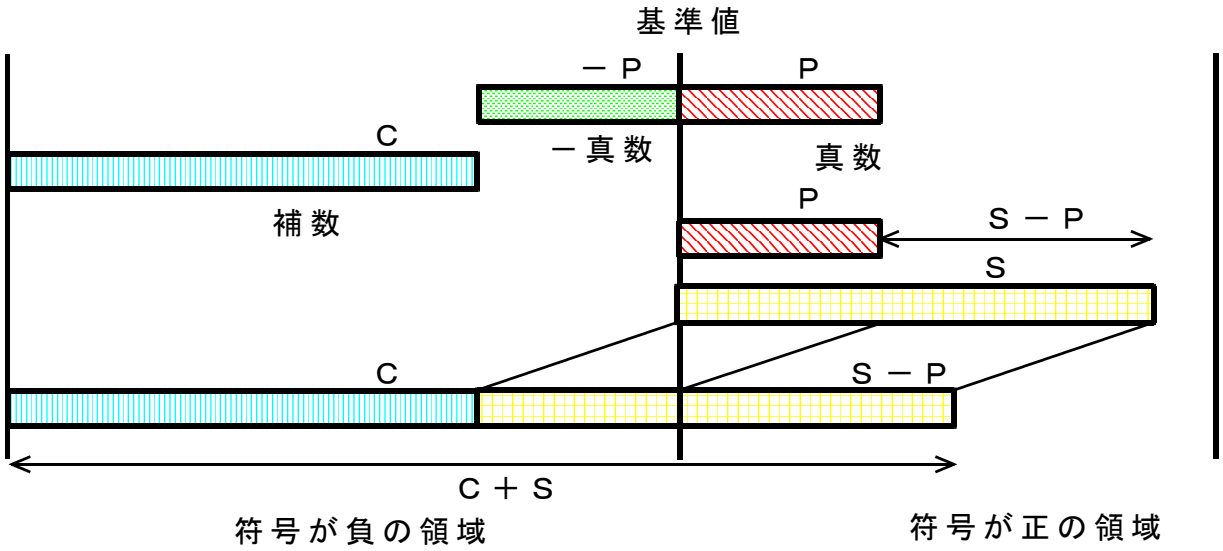
㋑　２進数では真数Ｐの補数は、基準値からＰの値を減じたＣの長方形（青縦線）になる。従って、負数を補数で表すと、Ｐの負数はＣの長方形（青縦線）で表すことになる。

㋒　補数Ｃに真数Ｐを加算すると、基準値になりこれが０となる。

㋓　値Ｓ（黄格子模様）から値Ｐを減じるＳ－Ｐの計算に、補数Ｃを用いると、次のように表すことができる。

Ｓ－Ｐ＝Ｓ＋Ｃ
＝Ｃ＋Ｐ＋（Ｓ－Ｐ）
＝Ｓ＋Ｃ－（Ｃ＋Ｐ）

Ｃ＋Ｐは基準値であるから、Ｓ－ＰはＳ＋Ｃの結果から基準値を減じた値となる。



具体例

10進数の正の整数321について考える。

整数321の補数は679である。

500－321＝179の計算を補数を使用して考える。

㋐　この計算を補数を使用して求めると、次のようになる。

500－321＝500＋679＝1179

㋑　先頭の１を除くと179となる。

㋒　先頭の１を除く操作は、基準値1000を減じたことになる。1000を減じるのは、1000からの増分を求めるためである。1179の演算結果を増分179で表現する。

具体例

４ビットの２進数1001から0110を減ずる演算について考える。

㋐　補数を使用して計算すると次のようになる。

1001－0110＝1001＋1010＝10011

㋑　先頭の１を除くと、0011となる。

㋒　先頭の１を除く処理は４ビットの２進数の基準値10000を減じることに等しい。10000を減じるのは、10000からの増分を求めるためである。10011の演算結果を増分0011で表現する。

㋓　1001は10進数で９、0110は10進数の６となる。従って、1001－0110は10進数の演算では、９－６＝３となる。

ⓒ　２進数の負数に１の補数を使用する場合

２進数の負数に１の補数を使用する場合がある。

この考え方の負数は、正数のビットパターンを反転することによって求めることができる。

具体例

㋐　負数を１の補数で表す４ビットの２進数を0101とすると、負数は各ビットを反転して次のようになる。

1010

㋑　0101は10進数の５であるから、1010は、10進数の－５である。

㋒　負数を２の補数で表す場合、0101の負数は1011となるから、同じ10進数に対して、１の補数は２の補数に比べ１つ小さい値になる。

ⓓ　１の補数と２の補数の値の並び

負数に１の補数を用いた場合と２の補数を用いた場合の３ビットの２進数の値の並びを表すと次のようになる。

１の補数	10進数の値	２の補数	10進数の値
011	＋３	011	＋３
010	＋２	010	＋２
001	＋１	001	＋１
000	＋０	000	＋０
111	－０	111	－１
110	－１	110	－２
101	－２	101	－３
100	－３	100	－４

ⓔ　負数に１の補数を使用する場合の問題点

２進数110に010を加算する場合を考える。

110＋010＝1000

負数に２の補数を用いる場合の処理を行うと、演算結果は000となる。

㋐　１の補数を用いる場合、110は10進数の－１であり、010は10進数の２であるから、演算結果は(－１)＋２＝＋１となり、矛盾する。これは10進数の０に相当する値が111と000の２つあることによって発生する矛盾である。

㋑　この矛盾を解消する手段として、加算結果に１のあふれが発生すると、あふれた１を最後の桁に加算する処理を実行する。

110＋010　→　1000　（１のあふれが発生）

→　000＋001

→　001

001は10進数の１であるから、正しい結果が求まる。

ⓕ　１の補数の使用例

１の補数の考え方は、ＩＰチェックサムや画像反転に利用される。

ＩＰチェックサムは以下の方法で計算する。

㋐　パケット全体を２オクテット(１オクテットは８ビット)ごとの16ビット列にする。

㋑　それぞれを１の補数として和を求める。

㋒　その１の補数をチェックサムとして使用する。

㊓　計算時にはチェックサムのエリアは０として計算する。

この数値表現体系は古いコンピュータでは一般的だった。ＰＤＰ−１とかＵＮＩＶＡＣ1100/2200seriesなど多くのシステムが１の補数を使っていた。

具体例

例えば、16ビットのビットパタンが次の場合

0101101101111001　……　ａ

１の補数を求めると、次のようになる。

1010010010000110　……　ｂ

㋐　ａ、ｂのビットパターンをチェック時の計算に使用して、両ビットを１の補数として加算すると、

1111111111111111＝FFFF

となり、結果は０(FFFFは１の補数では０)となる。

㋑　１の補数をチェックサムに利用すると検査回路が簡単になる。もし、どれかのビットにエラーが発生していると検査結果が０にならないので１ビットの誤りを検出することが可能になる。

具体例

16ビットの２進数0101101101111001を８ビット列単位にＩＰチェックサムの処理を実行すると、次のようになる。

㋐　８ビットの２進数01011011、01111001を１の補数として和を求める。

01011011＋01111001＝11010100

㋑　１の補数を求める。

00101011

㋒　㋑で求めた２進数をチェックサムとして用いると、元の16ビットの２進数は次の24ビットの２進数になる。

010110110111100100101011

ⓖ　10進数の負の整数を２進数に変換する手順

㋐　負の10進数の絶対値の２進数を求める。

㋑　㋐で求めた２進数の２の補数を求める。

具体例

㋐　負の10進数－92の絶対値92を２進数に変換する。

$(92)_{10}=(01011100)_{2}$

㋑　求めた２進数01011100の２の補数を求める。

01011100の２の補数は10100100となる。

㋒　10進数の－92を２進数で表すと10100100となる。

ⓗ　２進数の負の整数を10進数に変換する手順

㋐　負の２進数の補数を求める。

㋑　㋐で求めた補数の２進数を10進数に変換する。

㋒　この10進数に－の符号を付ける。

具体例

㋐　負の２進数10100100の補数を求める。

10100100の２の補数は01011100となる。

㋑　求めた補数を10進数に変換する。

01011100を10進数に変換すると、

64＋16＋８＋４＝92

㋒　この10進数の92に－の符号を付けると、－92となる。

③　２進数の表現範囲

ⓐ　８ビットの２進数の表現範囲

㋐　正の固定小数点数の整数部の２進数表現範囲

00000000～11111111

これを10進数に変換すると、０～２^{8}－１で、０～255となる。

㋑　11111111を10進数に変換する方法

11111111＋00000001＝100000000＝２^{8}

であるから、$(11111111)_{2}$＝２^{8}－１となる。

㋒　負数を２の補数で表した場合の２進数で表現できる範囲

10000000～01111111

これを10進数に変換すると、－2^7～＋2^7－１で、－128～＋127となる。

ⓑ　16ビットの２進数の表現範囲

㋐　正の固定小数点数の整数部の２進数表現範囲

0000000000000000～1111111111111111

これを10進数に変換すると、０～2^{16}－１で、０～65535となる。

㋑　負数を２の補数で表した場合の２進数で表現できる範囲

1000000000000000～0111111111111111

これを10進数に変換すると、－2^{15}～＋2^{15}－１で、－32768～＋32767となる。

ⓒ　ｎビットの２進数の表現範囲

㋐　正の固定小数点数の整数部の２進数で表現できる範囲

０００……００　～　１１１……１１
０がｎ個　　１がｎ個

これを10進数に変換すると、０～2^n－１となる。

㋑　負数を２の補数で表した場合、２進数で表現できる範囲

１００……００　～　０１１……１１
０がｎ－１個　　１がｎ－１個

これを10進数に変換すると、－2^{n-1}～＋2^{n-1}－１となる。

ⓓ　表現範囲のまとめ

	正数の場合	負数を含む場合
８ビット	０～255	－128～＋127
16ビット	０～65535	－32768～＋32767
ｎビット	０～2^n－１	－2^{n-1}～＋2^{n-1}－１

例題演習

２の補数で表された負数10101110の絶対値はどれか。

ア　01010000　　イ　01010001　　ウ　01010010　　エ　01010011

解答解説

２進数の絶対値に関する問題である。

２進数の絶対値は２進数の正数の値であるから、２進数の負数の絶対値は、与えられた２進数の補数を求めればよい。

10101110の２の補数は01010010となる。求める答えはウである。

例題演習

符号１ビット、整数部５ビット、小数部２ビットの数を、２の補数表示で表す。これで表現できる最大値はどれか。

ア　31.75　　イ　32　　ウ　63.5　　エ　127.25

解答解説

２進数の最大値を求める問題である。

２進数で表現できる最大の数は、符号は０、整数部は11111、小数部は11となるため、最大の８ビットの２進数は011111.11となる。

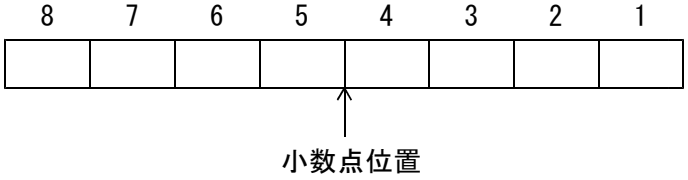
これを10進数に変換すると、

16＋８＋４＋２＋１＋0.5＋0.25＝31.75

求める答えはアである。

例題演習

10進数－5.625を、８ビット固定小数点形式による２進数で表したものはどれか。ここで、小数点位置は、４ビット目と５ビット目の間とし、負数は２の補数表現を用いる。



ア　01001100　　イ　10100101　　ウ　10100110　　エ　11010011

解答解説

10進数の負数を２進数に変換する問題である。

10進数の負数を２進数に変換する手順

①　負数の10進数の絶対値を２進数に変換する。

② 小数点の表示位置を考慮し、８ビットのビットパターンを作成する。

③ 求めた２進数の補数を求める。

10進数5.625を２進数に変換すると、0101.1010となる。

この２進数の補数を求めると、10100110となる。求める答えはウである。

例題演習

負数を２の補数で表す８ビットの数値がある。この値を10進数で表現すると－100である。この値を符号なしの数値として解釈すると，10進数で幾らか。

ア 28　　イ 100　　ウ 156　　エ 228

解答解説

補数に関する問題である。

10進数－100を２進数に変換すると、10011100

この２進数を符号なしの数値として求めると、128＋16＋８＋４＝156

求める答えはウとなる。

例題演習

８ビットのデータ10101100の解釈として、正しいものはどれか。ア～エの中から選べ。

ア ２の補数表示の符号付き小数として解釈すると、－0.3475である。

イ ２の補数表示の符号付き整数として解釈すると、－48である。

ウ 絶対値表示の符号付き整数として解釈すると、－44である。

エ 符号なし小数として解釈すると、0.875である。

解答解説

８ビットのビットパターンに対して次のような㋐～㋕６通りの解釈が成り立つ。

㋐ 負数を２の補数表示として解釈し、整数の場合

㋑ 負数を２の補数表示として解釈し、小数の場合

㋒ 符号＋絶対値の値で表し、整数の場合

㋓ 符号＋絶対値の値で表し、小数の場合

㋔ 符号なしの整数の場合

㋕ 符号なしの小数の場合

ア～エの場合について、10進数の値に変換する。

アは２の補数表示の小数の場合で、10101100の補数は01010100であるから10進数に変換すると、0.5＋0.125＋0.0312５＝0.65625となる。従って、－0.65625となり、誤りである。

イは２の補数表示の整数の場合で、補数は01010100であるから10進数に変換すると、４＋16＋64＝84、従って、－84となり、誤りである。

ウは0101100を10進数に変換すると、４＋８＋32＝44であるから、－44で正しい。求める答えはウとなる。

エは符号なし小数であるから、10進数に変換すると、0.5＋0.125＋0.03125＋0.015625＝0.671875で誤り。

例題演習

負の整数を表現する代表的な方法として，次の３種類がある。

ａ　１の補数による表現

ｂ　２の補数による表現

ｃ　絶対値に符号を付けた表現（左端ビットが０の場合は正，１の場合は負）

４ビットのパターン１１０１をａ～ｃの方法で表現したものと解釈したとき，値が小さい順になるように三つの方法を並べたものはどれか。ア～エの中から選べ。

ア　ａ，ｃ，ｂ　　イ　ｂ，ａ，ｃ

ウ　ｂ，ｃ，ａ　　エ　ｃ，ｂ，ａ

解答解説

１の補数の場合（ａ）

1101の補数は0010となり、10進数で２、従って元の２進数は－２となる。

２の補数の場合（ｂ）

1101の補数は0011となり、10進数で３、従って元の２進数は－３となる。

絶対値に符号をつけた場合（ｃ）

1101は－５となる。

従って大小関係は小さい順に並べると、ｃ、ｂ、ａとなり、求める答はエとなる。

例題演習

負数を２の補数で表現する固定小数点表示法において、ｎビットで表現できる整数の範囲はどれか。ここで、小数点の位置は最下位ビット（ＬＳＢ）の右とする。

ア　-2^{n}～2^{n-1}　　イ　-2^{n-1}～2^{n-1}

ウ　-2^{n-1}～$2^{n-1}-1$　　エ　$-2^{n-1}-1$～2^{n-1}

解答解説

２進数の範囲に関する問題である。

次の順序で考える。

①　負数を２の補数で表すｎビットの２進数で表現できる整数の範囲を求める。

②　２進数の範囲を１０進数の範囲に変換する。

一般に、ｎ桁ｍ進数の１０進数の表現範囲は０～$m^{n}-1$である。

２進数の場合、

固定小数点の整数部　　０～$2^{n}-1$

負数を２の補数表現　　-2^{n-1}～$2^{n-1}-1$

となる。２の補数表現では、負数の絶対値が正数の絶対値よりも１だけ多いことに注意する。

８ビット２進数の場合

固定小数点の整数部　　０～255

負数を２の補数表現　　－128～＋127

１６ビットの２進数の場合

固定小数点の整数部　　０～65535

負数を２の補数表現　　－32768～＋32767

負数を２の補数表現で表すため先頭ビットは符号になる。

ｎビットの表現範囲を２進数で表すと次のようになる。

1000……00～0111……11

ｎ－１ビット　　ｎ－１ビット

下線の部分のｎ－１ビットが絶対値になる。この２進数を10進数に変換すると

-2^{n-1}～$+2^{n-1}-1$

となり、求める答えはウとなる。

例題演習

ｎビットのすべてが１である２進数“1111…11”が表す数値又はその数式はどれか。ここで，負数は２の補数で表す。

ア　$-(2^{n-1}-1)$　　イ　－1

ウ　０　　エ　$2^{n}-1$

解答解説

２進数を10進数に変換する問題である。

負数を２の補数で表すｎビットの２進数111……11（１の数がｎ個ある）を10進数に変換する問題である。

２進数111……11（１の数がｎ個ある）の補数を求めると、000…001（０がｎ－１個連続する）となる。２進数111……11（１の数がｎ個ある）は10進数に変換すると－１となる。求める答えはイとなる。

txt010113 浮動小数点数

①　浮動小数点数

ⓐ　固定小数点数と浮動小数点数

コンピュータによる数値表現の方法に固定小数点方式と浮動小数点方式がある。

㋐　固定小数点方式

固定小数点方式は小数点が置かれる桁を固定して表された数のことで、ある桁数のうちのある場所に小数点が固定されているものとして扱う方式である。８ビットの２進数で考えて、小数点の位置を８ビットの２進数の右端に設定すると整数となり、８ビットの２進数の左端に設定すると小数となる。

固定小数点方式は表現できる値の範囲はビット数によって限定されるが、高速に演算できる利点がある。

㋑　浮動小数点方式

浮動小数点方式は、固定小数点方式で表現できない非常に大きな数や非常に小さい数字を表現するために、基数を使用して固定長の仮数部と指数部の３つの数値を利用し、少ない桁数で広い範囲の数値を表現する方式である。

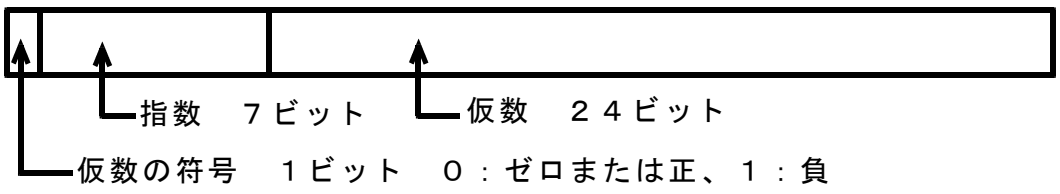
浮動小数点方式は指数表現によって可能な十分に広い絶対値の範囲内において、仮数部の桁数に依って常に一定の範囲内の相対誤差で任意の実数を近似できるという特徴がある。

ⓑ　任意の大きさの実数の表現

実数＝ｍ×Ｂ^{E}

但し、ｍ：仮数、Ｂ：底（基数、通常は16または２を使用する）、Ｅ：指数

ⓒ　１語を32ビットの浮動小数点データの表現



②　基数16の場合の浮動小数点数の求め方

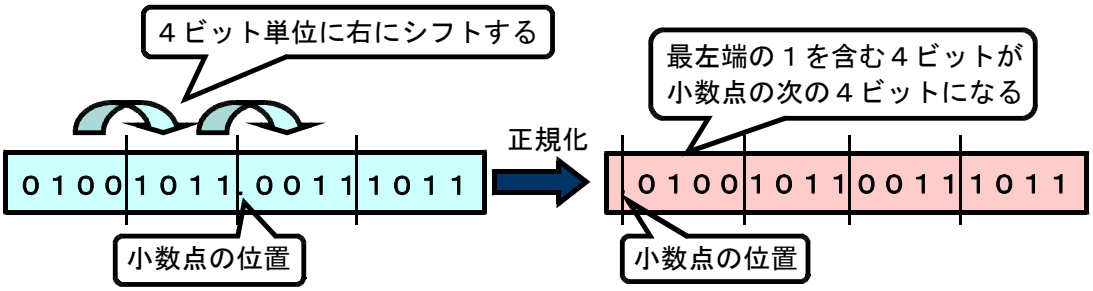
ⓐ　求め方の手順

㋐　10進数の絶対値を２進数に変換する。

10進数の絶対値の整数部分、小数部分をそれぞれ２進数に変換する。

整数部分、小数部分の２進数がそれぞれ４の倍数で、かつ合計が24ビットになるようにビット数を調整する。

㋑　正規化する。



小数点を起点として、整数部分、小数部分をそれぞれ４ビット単位にくくる。４ビット内に１を含む最上位の４ビットを、小数点の次の４ビットになるように４ビット単位に右または左シフトする。

仮数部のビット数が不足する場合は、最下位に０ビットを付加してビット数を調整する。

㋒　指数部を調整する。

正規化のシフト結果を利用して指数部の補正値を求める。

４ビット単位に右にＮ回シフトした場合、指数部にＮを加える。４ビット単位に左にＮ回シフトした場合、指数部からＮを減じる。

調整の方法にはバイアス方式と補数表現方式がある。

バイアス方式は、「1000000±指数部補正値」を使用して指数部の値を求める。

補数表現方式は、「0000000±指数部補正値」を使用して指数部の値を求める。

バイアス方式

１０００００００　±　指数部補正値

正規化で求めたＮの値を用いる

補数表現方式

０００００００ ± 指数部補正値

正規化で求めたＮの値を用いる

基数が16の場合の指数部の１の増加は、仮数部の値の16倍に相当し、指数部の１の減少は仮数部の値の16^{-1}倍に相当する。

基数が16の場合、仮数部を４ビット右にシフトすると、指数部を１だけ増加し、仮数部を４ビット左にシフトすると指数部を１だけ減ずることになる。

㊓　仮数の符号を決める。

10進数の値が０または正の場合は０、10進数の値が負の場合は１を設定する。

㊔　浮動小数点数を表示する。

符号部１ビットを２進数で表示する。指数部７ビットを２進数で表示する。

仮数部24ビットを２進数で表示する。仮数部が16ビットの場合は16ビットに調整する。

㊕　２進数を16進数に変換する。

符号部から４ビット単位に順次くくる。２進数４ビットを16進数に変換する。

具体例

10進数の　－125.5を、基数16、補数表現方式の浮動小数点数に変換し、結果を16進数で表せ。但し、符号部１ビット、指数部７ビット、仮数部16ビットする。

㋐　10進数の絶対値　125.5を２進数に変換すると01111101.1000となる。

㋑　正規化すると４ビット単位に２回右にシフトして、次のようになる。

01111101.1000　→　.0111110110000000

㋒　指数部は0000000＋0000010＝0000010となる。

㋓　符号部は10進数が負であるから１となる。

㋔　浮動小数点数の表示は10000010011111010110000000となる。

㋕　16進数に変換すると827D80となる。

③　基数２の場合の浮動小数点数の求め方

ⓐ　基数２の正規化

基数２の正規化は最左端の１のビットを小数１位になるように左または右にシフトする操作である。

ⓑ　指数部の値の増減

指数部の１の増加は、仮数部の値の２倍に相当し、指数部の１の減少は仮数部の値の2^{-1}倍に相当する。

基数が２の場合、仮数部を１ビット右にシフトすると、指数部を１だけ増加し、仮数部を１ビット左にシフトすると指数部を１だけ減ずることになる。

④　ＩＥＥＥの浮動小数点数

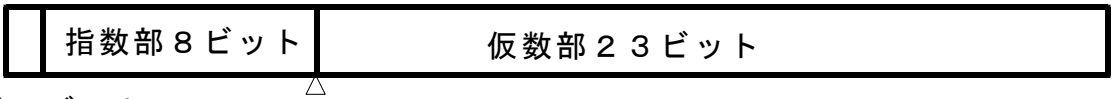
ⓐ　基数16の場合の浮動小数点数との相違

㋐　指数部が８ビットで、本来の指数部の値に＋127された値を表示する。

㋑　仮数部は23ビットで、仮数－１の２進数小数が記録される。

㋒　指数部の基数は２である。

ⓑ　ＩＥＥＥの浮動小数点数の表示形式



符号１ビット

小数点の位置

浮動小数点数で表現される値：$(-1)^{S} \times 2^{E-127} \times (1+F)$

仮数の符号：仮数部で表現されたデータが正あるいは負であることを表す。
０の場合が正、１の場合が負となる。

指数部　：２を基数として実際の指数に＋１２７した数値を表す。
Ｅ＝２５５、Ｆ≠０のとき非数、Ｅ＝２５５、Ｆ＝０のとき$(-1)^{S} \infty$
Ｅ＝０、Ｆ≠０のとき$(-1)^{S} 2^{-128} (0.F)$
Ｅ＝０、Ｆ＝０のとき$(-1)^{S}$０（ゼロ）

仮数部　：ここで表現できるのは１以下の２進数小数とする。

ⓒ　求め方の手順

㋐　基数変換する。

10進数を２進数に変換する。

㋑　正規化する。

２進数にシフト演算を行い、ＭＳＢのビットを１の位にシフトする。

シフト後の値と元の値の調整を行う。シフト演算のビットシフトが右にＮビットの場合、指数部の調整は＋Ｎ、シフト演算のビットシフトが左にＮビットの場合、指数部の調整は－Ｎとなる。

㋒　指数部の値を決める。

01111111±Ｎで求める。Ｎの値は正規化の段階で求めた値を使用する。

㋓　仮数部の値を決める。

正規化で求めた値から１を引いた小数の２進数を用いる。仮数部が23ビットになるように調整する。

㋔　符号部の値を決める。

元の10進数が正の場合は０、負の場合は１となる。

㋕　浮動小数点数を２進数で表示する。

符号部の１ビットは⑤で求めた値を使用する。指数部の８ビットは③で求めた値を使用する。仮数部の23ビットは④で求めた値を使用する。

具体例

10進数の45.625をＩＥＥＥ754の浮動小数点数の表示形式で表せ。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	…	0
0	1	0	0	0	0	1	0	0	0	1	1	0	1	1	0	1	0	…	0

IEEE 754 の形式による 45.625 の表現

㋐　10進数の45.625を２進数で表すと，$(101101.101)_2$となる。

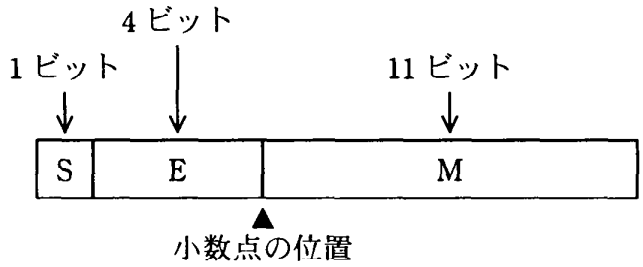
㋑　これを正規化した指数表現にすると，$(1.01101101)_2 \times 2^5$となる。

㋒　ＩＥＥＥ754の形式で表すと，図に示すとおりとなる。

㋓　符号部は$(0)_2$，指数部は指数の$(5)_{10}$に$(127)_{10}$を加えて$(10000100)_2$となり，仮数部は整数部分を省略するので，$(011011010\cdots0)_2$となる。

例題演習

数値を16ビットの浮動小数点数で図に示す形式で表す。10進数0.375を正規化した表現は図中のア～エのうちのどれか。ここでの正規化は，仮数部の最上位けたが０にならないように指数部と仮数部を調節する操作である。



S ：仮数部の符号（0：正，1：負）
E ：指数部（2 を基数とし，負数は 2 の補数で表現）
M ：仮数部（2 進数 絶対値表示）

ア

0	0001	11000000000

イ

0	1001	11000000000

ウ

0	1111	11000000000

エ

1	0001	11000000000

解答解説

基数が２の場合の正規化の考え方を使用し、符号１ビット、指数４ビット、仮数11ビットの計16ビットの２進数で表示する。指数部は２の補数表現を用いる。

㋐ 10進数の0.375を２進数に変換すると、0.011となる。
正規化のためには、左に１ビットシフトすると、次のようになる。

$(0.011)_2 = (0.11)_2 \times 2^{-1}$

㋑ 指数部は２を基数とし、負数は２の補数表現であるから、指数部の値は次のようになる。

0000－0001＝0000＋1111＝1111

㋒ 仮数部の符号は正であるから、16ビットの浮動小数点は次のようになる。

0111111000000000

㋓ 求める答えはウとなる。

例題演習

浮動小数点表示において、仮数部には正規化された表現を用いる。その理由として最も適切なものはどれか。

ア　扱う数値の範囲が拡大できるため　　イ　演算回路が簡単になるため
ウ　演算速度が速くなるため　　エ　精度を保つため

解答解説

浮動小数点表示の正規化に関する問題である。

正規化とは基数が16の場合で考えると、対象の仮数を小数点を基点に上下にそれぞれ４ビットずつのグループ(16進数の１桁に相当する)を形成し、最も上位の０でない４ビットを小数１位から小数４位の間にビットシフトすることである。

ビットシフトは基数が16であるから４ビット単位に行う。基数が２の場合には、仮数の対象のグループは１ビットとなり、最上位の１ビットを小数１位にビットシフトすることになる。

正規化することによって、有限桁数で表示される仮数の有効桁数は最も大きくなり、精度が向上する。求める答えはエとなる。

例題演習

10進数 0.046875を長さ24ビットの浮動小数点バイアス方式、16進数で表したものはどれか。但し、左端の１ビットは仮数の符号で、０は正、１は負を表し、指数部は７ビット、仮数部は16ビットで表すものとする。

ア　3FC000　　イ　7FC000　　ウ　01C000　　エ　BFC000

解答解説

10進数を16進数の浮動小数点表示に変換する問題である。

次の手順で求める。

① 10進数0.046875 を２進数に変換すると、$(0.000011)_2$となる。

② 正規化すると、0.1100×16^{-1}となる。

③ 指数部を求めると、バイアス方式であるから
1000000－0000001＝1000000＋1111111＝0111111

④ 符号を求めると、元の10進数が正であるから、０となる。

⑤ これを24ビットで表現すると、
001111111100000000000000

⑥ これを16進数に変換すると、3FC000となり、求める答えはアとなる。

txt01012 算術演算と精度

txt010121 ２進数の四則演算

①　２進数の加減算

ⓐ　２進数の加算の規則

0	0	1	1
＋0	＋1	＋0	＋1
0	1	1	10

１＋１＝10となり桁上げで表す。桁上げされた１は上位の桁に加算される。

具体例

㋐

```
  １０１１
＋    １０
  １１０１
```

㋑

```
  １１０１
＋   １１０
  １００１１
```

桁上げされた１は上位の桁に加算

ⓑ　加算結果の“あふれ”

負数を２の補数表現で表す場合の異符号の加算は、和の絶対値はどちらの演算数の絶対値よりも大にはならないため「あふれ」は生じない。

負数を２の補数表現で表す場合の同符号の加算は、演算結果の符号が被加数、加数の符号と異なると「あふれ」が生じて正しい答えを求めることができない。加算結果の値がそのビット数で表現できる範囲の値を超えるため「あふれ」が発生する。

具体例

00100111は10進数の39、10111010は10進数の－70、その和は11100001となり、10進数の－31となる。

２進数		10進数
00100111	……	39
＋　10111010	……	－70
11100001	……	－31

具体例

２進数0101と0010の和は0111となる。２進数0101は10進数の５であり、２進数0010は10進数の２であるから５＋２＝７となり、２進数で表すと0111となる。

２進数0101と0100の和は1001となる。２進数0101は10進数の５であり、２進数0100は10進数の４であるから５＋４＝９となり、２進数で表すと1001となる。

しかし、負数を２の補数で表す２進数の加算の場合、1001は10進数の－７となり、10進数の加算の結果と一致しなくなる。これは、同符号の加算で、加算結果の符号が被加数、加数の符号と異なる結果となっており、あふれが発生したためである。

```
  ０１０１  ……   ５          ０１０１  ……   ５
＋００１０  ……  ＋２        ＋０１００  ……  ＋４
  ０１１１  ……   ７          １００１  ……   ９
  ↑                            ↑
  └── 符号が変わらない          └── 符号が変わる
```

ⓒ　２進数の減算の規則

```
  0      0      1      1
－0    －1    －0    －1
  0     －1      1      0
```

０－１＝－１となり、上位の桁に１があれば、桁借りが生じる。10－１＝１となる。

具体例

```
  10011
－ 1001
  1010
  ↑
  └── ０－１は上位の桁から１を借りて、２－１＝１となる。
```

ⓓ　負数を２の補数表現で表す２進数の減算

２の補数を用いて加算で実現できる。0010－1111＝0010＋0001＝0011

具体例

0010－1111＝0010＋0001＝0011

0010は10進数の２、1111の補数は0001で、10進数の－１となるため10進数の演算で表すと、２－（－１）＝２＋１＝３となる。

ⓔ　２進数の異符号の減算結果の「あふれ」

負数を２の補数表現で表す２進数の異符号の減算を、２の補数を用いて加算で行った結果、

被加数・加数の符号と加算結果の符号が異なると「あふれ」が生じて正しい答えを求めることができない。

具体例

0111－1010＝0111＋0110＝1101

0111は10進数の７、1010の補数は0110で、10進数の－６となるため10進数の演算で表すと、７－(－６)＝７＋６＝13となり、４ビットの２進数の整数の範囲の－８～＋７を越えるためあふれが生じる。

②　シフト演算

ⓐ　論理シフト

ビットシフトは、ビット列を右または左に「ずらす」ことである。右に１ビットずらすことを“右に１ビットシフトする”といい、左に１ビットずらすことを　“左に１ビットシフトする”という。符号を考えない場合を論理シフトという。右にシフトして空欄になった左の桁や左にシフトして空欄になった右の桁を０で補う。

具体例

「右へ３ビット論理シフトする」場合

01101010　→　00001101

０で補う

となり、010の３ビットが右にシフトアウトする。空欄になった左側の３桁を０で補う。

「左へ３ビット論理シフトする」場合

01101010　→　01010000

０で補う

となり、011の３ビットが左にシフトアウトする。空欄になった右側の３桁を０で補う。

ⓑ　算術シフト

算術シフトは最上位のビット(ＭＳＢ)を符号と考える場合で、符号ビットはシフトの対象外とする。右に算術シフトして、符号の後の空になった所には符号と同じビットが入る。従って、正の場合は０、負の場合は１が入る。左に算術シフトする場合も符号はシフトの対象にならない。下位の空いた桁には０が入る。

具体例

８ビットの２進数を「３ビット右へ算術シフトする」場合

01101010　→　00001101

符号はシフトの対象外　　０で補う

10101100　→　11110101

符号はシフトの対象外　　１で補う

具体例

８ビットの２進数を「３ビット左へ算術シフトする」場合

01101010　→　01010000

符号はシフトの対象外　　０で補う

11111100　→　11100000

符号はシフトの対象外　　０で補う

③　２進数の乗除算

ⓐ　２進数の乗算

被乗数を$(10)_2=(2)_{10}$倍、$(100)_2=(2^2)_{10}$倍、…、する場合、被乗数を左へ１ビット、２ビット、…、シフトすることによって求めることができる。被乗数を2^n倍する場合は、元の被乗数を左へｎビットシフトすればよい。下位の桁の空いたところには０を詰めていく。

具体例

２進数1101を$(10)_2$倍すると、１ビット左にシフトして11010となり、２進数1101を$(100)_2$倍すると、２ビット左にシフトして110100となる。

２進数1101を110倍数する場合、1101×100と1101×10の結果の和として求める。

1101×110＝1101×100＋1101×10

＝110100＋11010＝1001110

ⓑ　被乗数を10進数のｍ倍する乗算

10進数ｍを２進数に変換し、左へのビットシフトと加算によって求める。

２進数Ｙをｍ＝17倍する乗算は次のようになる。

㋐　10進数17を２進数に変換する。$(17)_{10}=(10001)_2$

㋑　左へ４ビットシフトした結果を求める。Ｙ×10000

㋒　㋑で求めた結果と元の２進数の和を求める。

Ｙ×(10000＋１)＝Ｙ×2^4＋Ｙ

ⓒ　乗算結果の“あふれ”

正数の場合、左に算術シフトして、１のオーバフローが発生すると、答えは正しくない。負数の場合、左に算術シフトして、０のオーバフローが発生すると、答えが正しくない。

具体例

４ビットの２進数0110を１ビット左に算術シフトすると、0100となる。

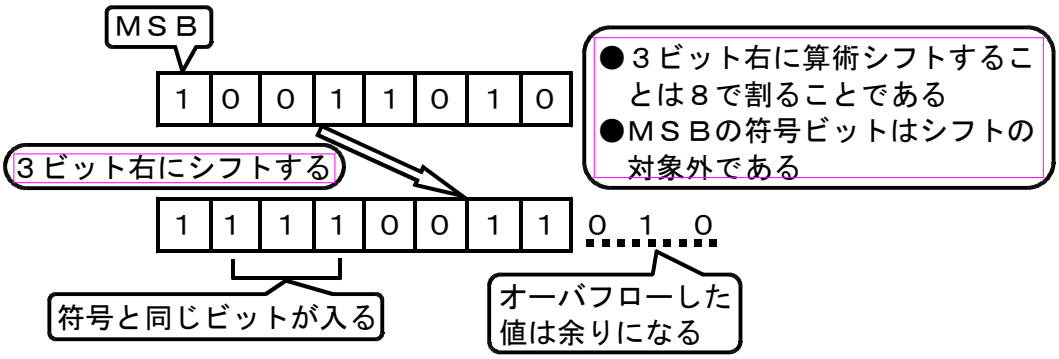
２進数の0110は、10進数の６であるから、２倍すると結果は12となる。しかし、算術シフトした0100は４となり、12とはならない。

具体例

４ビットの２進数1001を１ビット左に算術シフトすると、1010となる。

２進数の1001は、10進数の－７であるから、２倍すると結果は－14となる。しかし、算術シフトした1010は－６となり、－14とはならない。

ⓓ　２進数の除算



被除数を$(10)_2=(2)_{10}$、$(100)_2=(2^2)_{10}$、…、で割る場合、被除数を右へ１ビット、２ビット、…、シフトすることによって求めることができる。

ⓔ　被除数を２ⁿで割る場合の計算要領

㋐　正の２進数の除算は、被除数を２ⁿで割るとき、もとの被除数を右へｎ桁シフトし、空いたところには符号と同じ０を入れる。

㋑　負の２進数の除算は、被除数を２ⁿで割るとき、もとの被除数を右へｎ桁シフトし、空いたところには符号と同じ１を入れる。

㋒　被除数を10進数のｍで割る場合は、10進数のｍを２進数に変換し、除数のビットシフトと減算によって求める。

具体例

２進数10101を３で割る場合の計算要領は次のようになる。

㋐　10進数の３を２進数に変換する　　111

（３）$_{10}$＝(11)$_{2}$　　11）10101

㋑　10101から1100を引く　　1100

10101－1100＝1001　　1001

㋒　㋑の結果の1001から110を引く　　110

1001－110＝11　　11

㋓　㋒の結果から11を引く　　11

11－11＝０　　０

㋔　計算結果が０になるか、11より小さい値になると減算を終了し、㋑、㋒、㋓で用いた除数×倍数の倍数のみの和を求める。

100＋10＋１＝111

ⓕ　除算の計算要領のまとめ

㋐～㋔の計算要領をまとめると、次のようになる。

㋐　被除数をＡ、除数をＢとして、左にビットシフトしてＡより小さくて最も大きい倍数Ｃを求める

㋑　Ａ－Ｃの減算結果Ｄを求める。

㋒　このステップでの商はＢからＣへのビットシフトの量である

㋓　㋑の減算結果Ｄが０になるか、またはＢより小さくなると除算完了である

㋔　㋓でないならばＣを１ビット右にシフトして新しい倍数Ｃを求める

㋕　㋑で求めたＤが新しい倍数Ｃより大きい場合、ＤをＡに代入して㋑に戻る。

㋖　㋑で求めたＤが新しい倍数Ｃより小さい場合、更に、Ｃを１ビット右にシフトして新しい倍数Ｃを求め㋕に戻る。

㋗　求める商は㋒で求めた各ステップのビットシフト量の和になる。

ⓖ　余りの発生

最下位の桁から１のアンダーフローがあると余りが発生する。符号が正の場合、シフトアウトしたビットが余りとなる。符号が負の場合、シフトアウトしたｎビットから、2^{n}を引いたものが余りとなる。11の２ビットがシフトアウトすると、余りは３－2^{2}＝－１となる。符号が負

の場合の商は、２進数から求めた10進数に１を加算した値となる。

２進数の割算のあまりの求め方

- **正数の場合** ― シフトアウトしたビットが余りとなる
- **負数の場合** ― シフトアウトしたｎビットから、2^nを引いたものが余りとなる

具体例

８ビットの２進数の10101010は10進数の－86で、２ビット右に算術シフトすると、２進数は11101010で10進数の－22となる。

従って、商は－21、余りは－２となる。

④　浮動小数点数の演算

ⓐ　整数と浮動小数点数との演算

整数と整数、浮動小数点数と浮動小数点数だけではなくに整数と浮動小数点数に対して演算を行うことも可能である。ただし、演算の結果得られる数値が整数になるか浮動小数点数になるかを注意する必要がある。

表に示すように、演算子の左辺又は右辺のどちらかが浮動小数点数の場合は演算結果は浮動小数点数になる。

演算子が除算の場合、次の注意が必要となる。

㋐　整数を整数で除算した場合、結果が整数となるため割り切れない場合は小数点以下が切り捨てられた整数となる。

㋑　除算の結果が浮動小数点数であった場合、左辺又は右辺のどちらかを浮動小数点数にする必要がある。

左辺	右辺	演算結果
整数	整数	整数
整数	浮動小数点数	浮動小数点数
浮動小数点数	整数	浮動小数点数
浮動小数点数	浮動小数点数	浮動小数点数

ⓑ　浮動小数点数の加減算

仮数部を加減算する時点で桁合わせを行う必要がある。

２数のどちらか一方の指数部を他の指数部と同じとなるように、仮数部をシフトアップ／ダウンする。通常は、絶対値の大きい方を基準とし、絶対値の小さい方の指数部を大きい方と同じにして仮数部をダウンシフトすることで桁合わせを行う。ダウンシフト量は２数の指数部の差になる。

ⓒ　浮動小数点数の加減算手順

㋐　２つの浮動小数点数Ａ、Ｂの指数部の値が同じ場合、Ａ、Ｂの仮数部の値を使用して演算する。

㋑　２つの浮動小数点数Ａ、Ｂの指数部の値がＡ≧Ｂならば、Ｂの指数部の値がＡの指数部の値と同じになるようにＢの仮数部の値を右にシフトし、シフト後のＡ、Ｂの仮数部の値を使用して演算する。右にシフトする量はＡの指数部の値とＢの指数部の値の差となる。

㋒　２つの浮動小数点数Ａ、Ｂの指数部の値がＡ＜Ｂならば、Ａの指数部の値がＢの指数部の値と同じになるようにＡの仮数部の値を右にシフトし、シフト後のＡ、Ｂの仮数部の値を使用して演算する。右にシフトする量はＢの指数部の値とＡの指数部の値の差となる。

具体例

次の二つのＩＥＥＥ754の表示形式で表した浮動小数点数を加算せよ。

㋐　指数部が10000000と10000010であるから、加算前に指数部の調整が必要である。

㋑　両者の指数を10000010に調整すると、前者の仮数は1.01000から調整して0.0101000となり、後者の仮数は1.0100000となる。

㋒　両者を加算すると1.1001000となる。

㋓　演算結果の浮動小数点数の表示は01000001010010000…000となる。

31	30	29	28	27	26	25	24	23	22	21	20	19	18	…	0
0	1	0	0	0	0	0	0	0	0	1	0	0	0	…	0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	…	0
0	1	0	0	0	0	0	1	0	0	1	0	0	0	…	0

例題演習

	演算	オペランドｘ	オペランドｙ
ａ	ｘ＋ｙ	正	正
ｂ	ｘ＋ｙ	正	負
ｃ	ｘ＋ｙ	負	正
ｄ	ｘ＋ｙ	負	負
ｅ	ｘ－ｙ	正	正
ｆ	ｘ－ｙ	正	負
ｇ	ｘ－ｙ	負	正
ｈ	ｘ－ｙ	負	負

コンピュータを使用して整数の加減算を行う場合、あふれ(オーバフロー)に注意する必要がある。次の表のうち、あふれ(オーバフロー)の可能性がある組合せはどれか。ア～エの中から選べ。

ア　ａ．ｄ．ｆ．ｇ　　イ　ｂ．ｃ．ｅ．ｈ　　ウ　ｂ．ｅ　　エ　ｃ．ｅ．ｈ

解答解説

整数の演算結果があふれを発生させるのは、次の場合である。

①　同符号の加算である。従って、ａまたはｄの場合である。

②　異符号の減算である。従って、ｆまたはｇの場合である。

③　加算結果の符号が変化する。

この表では加算結果の符号の変化については評価できない。従って、可能性のある①、②の場合が求める答えとなり、ａ、ｄ、ｆ、ｇの場合で、求める答えはアとなる。

例題演習

次の８ビットで表現された二つの２進数の加算結果を、10進数で表現したものはどれか。なお、最上位ビットは符号で、負数は２の補数で表示されている。

00001110＋10000100

ア　146　　イ　10　　ウ　－18　　エ　－110

解答解説

四則演算に関する問題である。

```
  00001110
＋10000100
----------
  10010010
```

２進数10010010を10進数に変換する。

補数を求めると、01101110となり、10進数に変換すると

64＋32＋８＋４＋２＝110

答えは－110となり、求める答えはエとなる。

例題演習

８ビットのレジスタに、ある負数が２の補数表示で入っている。これを４ビット右へ算術シフトをした結果として、あり得るビット列はどれか。

ア　00000111　　イ　00001111　　ウ　10000110　　エ　11111111

解答解説

算出シフトに関する問題である。

レジスタの数値は負であるから先頭ビットは１であり、次の範囲の２進数である。

10000000～11111111

算術シフトの結果は、先頭ビットは１であり、右に算術シフトした空きの桁には符号ビットと同じビットが入るため、先頭の５ビットは11111……となる。先頭の５ビットが１のものはエである。求める答えはエとなる。

ア、イは先頭ビットが０であり、正しくない。

ウは４ビット右へ算術シフト結果の表現として正しくない。

例題演習

１６進小数0.FEDCを４倍した値はどれか。

ア　1.FDB8　　イ　2.FB78　　ウ　3.FB70　　エ　F.EDC0

解答解説

２進数の四則演算に関する問題である。

次の手順で求める。

①　16進数の小数を２進数に変換する。

②　2^n倍の掛け算はビットを左にｎビットシフトする、2^n倍のわり算はビットを右にｎビットシフトする。

③　４倍であるから、2^2で、左に２ビットシフトすればよいことになる。

16進小数を２進数に変換する。

$(0.FEDC)_{16}$＝0.1111111011011100となる。

４倍することは、左に２ビットシフトすることであるから、次のようになる。

0011.1111101101110000＝3.FB70

となり、求める答えはウとなる。

例題演習

負数を２の補数で表現する２進整数値の最下位２ビットが“11”であった。この２進数を10進数の４で割ったとき、その余りはどれか。ただし、除算の商は、その絶対値の端数が切り捨てられるものとする。

ア　その２進数が正であれば３
イ　その２進数が負であれば－３
ウ　その２進数の正負にかかわらず０
エ　その２進数の正負にかかわらず**３**

解答解説

右に算術シフトした場合の余りに関する問題である。

負数を２の補数で表す場合の２進数を10進数の４で割る場合であるから、右に２ビット算術シフトすることになる。

① 符号が正の場合、余りはオーバフローしたビットの値になる。オーバフローするのが11の場合であるから、符号が正の場合は余りは３になる。

② 符号が負の場合、ｎビットオーバフローすると、ｎビットの値から2^nを引いた結果が余りとなる。オーバフローするのが11の場合であるから、２進数の11は10進数の３であり、その値から2^2を引いた値が余りになる。符号が負の場合の余りは$3-2^2=-1$となる。

アは２進数の符号が正の場合で余りは３となり、正しい。求める答えはアとなる。

イは負の場合で余りは３－４＝－１となるため正しくない。

ウ、エは正の場合は３、負の場合は－１であるから正しくない。

例題演習

数値を２進数で表すレジスタがある。このレジスタに格納されている正の整数ｘを10倍にする操作はどれか。ここで，シフトによるけたあふれは，起こらないものとする。

ア　ｘを２ビット左にシフトした値にｘを加算し，更に１ビット左にシフトする。
イ　ｘを２ビット左にシフトした値にｘを加算し，更に２ビット左にシフトする。
ウ　ｘを３ビット左にシフトした値と，ｘを２ビット左にシフトした値を加算する。
エ　ｘを３ピット左にシフトした値にｘを加算し，更に１ビット左にシフトする。

解答解説

シフト演算に関する問題である。

10進数の10を２進数に変換すると、$10=(1010)_2$となる。従って、正の整数ｘを左に１ビットシフトした結果と３ビットシフトした結果を求め、その和を求めればよいことになる。

従って、まず、ｘを２ビット左にシフトした結果にｘを加算し、その結果を左に１ビットシフトすればよい。求める答えはアとなる。

アの場合、100＋１＝101、1010→10倍

イの場合、100＋１＝101、10100→20倍

ウの場合、1000＋100＝1100→12倍

エの場合、1000＋１＝1001、10010→18倍

例題演習

浮動小数点表示方式に関する記述のうち、正しいものはア～エのうちのどれか。浮動小数点表示は、数値を次のように表現するものである。ここで、Ｓは符号であり、Ｎが非負数のとき０、負数のとき１とする。

$N = (-1)^{S} \times a \times r^{e}$

ア　ａは仮数と呼ばれ、固定小数点形式で表現し、１＜ａ≦ｒとする。これを正規化という。

イ　ｅは指数で、負の場合は２の補数で表現する。

ウ　ｒは指数の基準と呼ばれ、多くの場合は10が用いられる。

エ　浮動小数点表示にすると、同じビット数の固定小数点表示に比べて、数値の範囲は広くなるが、有効桁数は少なくなる。

解答解説

浮動小数点表示方式に関する問題である。

アの仮数ａの範囲は、０≦ａ＜１となる。

イの指数部の表示は、バイアス方式と補数表現方式があり、バイアス方式では負の場合でも、正の整数となる。

ウの基数は、16または２を用いる。

エの同じビット数の場合、浮動小数点表示は、固定小数点表示に比べて、指数表示を使用するため数値の範囲は広くなるが、仮数の桁数は少なくなるため有効桁数は少なくなる。求める答えはエとなる。

txt010122 計算機の誤差

① 正確度、単精度、倍精度

ⓐ 正確度と精度

正確度は正確性の尺度であり、精度は再現性の尺度である。正確度は、その値が真値に近い値であることを示す尺度であり、系統誤差が小さいことである。精度は、複数回の測定または計算の結果、その複数回の値の間での互のばらつきの小ささを表す尺度である。偶然誤差が小さいことである。

ⓑ 単精度と倍精度

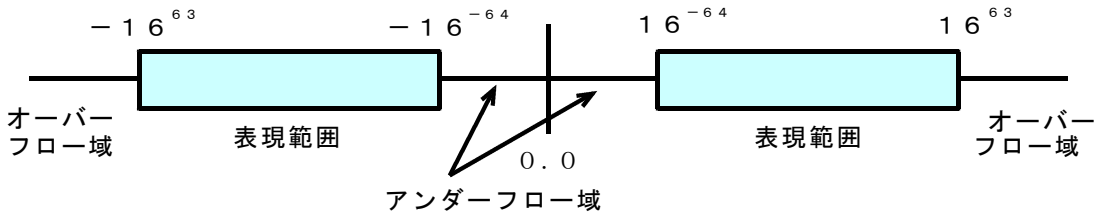
実数表現において４バイト／32ビットで実数を表現する方法を単精度といい、単精度の２倍のビット数64ビットを用いて実数を表現したものを倍精度という。

浮動小数点数の場合、単精度では、符号部が１ビット/仮数部が24ビット/指数部が７ビットで表現され、仮数部の精度は10進数で６桁となる。一方、倍精度では仮数部52ビット/指数部が11ビットとなり、精度は15桁に上がる。

有限桁を使用して演算するコンピュータの処理では精度が限られているため、浮動小数点の計算には誤差が付きものである。

② 浮動小数点数の表示範囲

ⓐ 基数16の表示範囲



浮動小数点数は、-16^{-64}～０の範囲、０～16^{-64}の範囲のような０の近傍の値、-16^{63}より小さい値、16^{63}より大きい値は表現不能となる。

ⓑ 誤差の発生

浮動小数点数の表示範囲が限られているため、指数調整や演算結果が表現不能範囲になると、誤差が発生する。表現が有限桁であるため、桁外れが発生したり、有効桁数が減少し、誤差を発生させる。また、計算機の演算回数を有限回に制約することによって誤差が発生したり、演算方法、演算順序によって誤差の大きさが異なったりする。

③　浮動小数点演算と誤差

ⓐ　情報落ち

浮動小数点演算において、絶対値の大きな値に絶対値の小さな値を加算する場合、指数部を揃えて計算するため、絶対値の小さな値が無視され誤差が発生する。

加算の場合、両者の指数部を合わせるとき、小さい方の仮数部が右に大きくシフトし、有効ビット範囲から下位の桁が欠落する。この現象を情報落ちという。

２進数Ａ：01000101000100000001100000000000

Ｂ：00111111101000000000000000000000

の場合、Ｂの２進数をＡの２進数の指数に合わせると

Ｂ：01000101000000000000000000000000

となり、Ｂの仮数部の1010は24桁の有効範囲から外れて、情報落ちとなる。

情報落ちの誤差は計算を実行する前に発生する誤差である。

ⓑ　桁落ち（アンダーフロー）

桁落ちは、浮動小数点演算において、非常に小さな指数表記の浮動小数点数同士の乗算を行うと、指数が指数部で表現できる範囲を下回ってしまうことであり、ほぼ等しい値の浮動小数点同士を減算すると、結果が非常に小さい値になるため表現できる範囲からもれ、有効桁数が大幅に減ってしまうことである。

浮動小数点表示では、０近傍の値を表現することが不可能なためにこの現象が発生する。演算結果、発生する誤差である。

ⓒ　桁あふれ（オーバフロー）

桁あふれは、非常に大きい値同士を掛けると、指数部で表現できる最大の範囲を超えてしまう現象である。２進数のビット数を増加させることで、誤差の発生を少なくすることができる。

ⓓ　丸め誤差

丸め誤差は、入力データとして有限桁の数値を使用する場合、または数値計算の結果、四捨五入、切り上げ、切り捨てによって意味ある有効桁に丸めるときに発生する誤差である。

10進数の値を２進数に変換する場合、２進数を限られたビット数で表示するときに誤差が発生する。10進数の小数、0.1、0.2、0.3、0.4、0.6、0.7、0.8、0.9を２進数の小数に変換すると循環小数となり、有限けた数で表現すると、丸め誤差が発生する。

④　演算方法と誤差

ⓐ　誤差を少なくする演算法

全てのデータを絶対値の昇順に並べ替え、先頭から順に加える。正数は加算し、負数は減算

する計算法で誤差の発生が最も小さくなる。

ⓑ　情報落ちによる誤差を少なくする方法

絶対値の差が小さい値同士の演算を行えば指数調整による誤差を少なくすることができる。

ⓒ　複数の演算を続けて実行する場合の対策

加算結果の値と次に加算する値の差を小さくすると、誤差を小さくすることができる。

⑤　数値計算と誤差

ⓐ　誤差が生じる原因

誤差が生じる原因で分類すると、過失誤差、系統誤差、偶然誤差がある。過失誤差は数値の読み違いや測定器の誤差、操作ミスなど数学的に意味のない誤差である。

系統誤差はある一定の規則や原因によって生じる誤差である。偶然誤差は統計的に全く偶然に起こる誤差であり、ガウスの誤差分布法則に従っていて原則的には除去できない誤差である。

ⓑ　解析誤差

解析誤差は現象を数式化するときの手法や数値計算の方法(アルゴリズム)、近似公式の使い方等の差異で生じる誤差である。適切なモデルや計算公式を工夫すればある程度回避できる。

公式誤差は近似式や関数近似の使い方の違いによる誤差、算法誤差はアルゴリズムやモデル選定が不適当なために生じる誤差である。

ⓒ　演算誤差

打ち切り誤差は数値計算を途中で打ち切るために生じる誤差である。

数学的に正確な関数値をｆ(ｘ)、有限回の演算操作で近似した関数値をｆａ(ｘ)とすると、ｆ(ｘ)－ｆａ(ｘ)を打ち切り誤差という。無限級数の部分和で展開されている近似公式を使うとき、定積分の値を数値積分の台形公式やシンプソンの公式で近似するとき、微分方程式を差分方程式に置き換えて解くときなどに問題になる。

丸めの誤差は入力データとして有限桁の数値を使うとき、または数値計算の結果を意味のある有効桁に丸めるときに生じる誤差である。丸め誤差は各段階で小さくても、計算を反復して行くにつれて次第に累積し、大きな誤差になることがある。

変換誤差は10進数を２進数に変換するときに生じる誤差である。

内在誤差(データ誤差)はデータ本来が既にもっている誤差である。

⑥　誤差と近似値

ⓐ　相対誤差

観察や実験によって得られる測定値は、ある限られた桁までしか意味を持たない近似値である。

ある測定値の真値Mの近似値をmとするとき

$\varepsilon = m - M$　あるいは　$M = m - \varepsilon$

ε を近似値mの誤差という。

この場合 $|\varepsilon|$ を絶対誤差、近似値の誤差と真値との比 $\varepsilon_r = \varepsilon / M$ を相対誤差という。一般には真値Mは知ることはできないから近似値mをMの代わりに使用して $\varepsilon_r = \varepsilon / m$ を相対誤差という。

ⓑ　誤差の限界

ある小さな任意の正数を α とするとき、$|\varepsilon| \leqq \alpha$ となるような α の値を誤差限界という。誤差のとりうる範囲をあらかじめ指定しておきたいときに使う。

誤差限界 α と近似値mとの比 α / m を相対誤差の限界という。

真値Mは未知の量であって正確に知ることは原理的にできない。誤差の限界 α がわかっていれば、この α を十分小さくとることによって、$m - \alpha \leqq M \leqq m + \alpha$ から真値Mに十分近い値として知ることができる。

ⓒ　有効数字

量を表す数値はすべて誤差を含んでいるからふつう数値は小数部分をもった実数の形で表される。このとき下位の桁ほど誤差のために不確かになるから、その部分を適当な方法で処理する。そのとき残った意味のある数値を有効数字といい、その意味のある数値の桁数を有効桁数という。数値計算では有効数字の桁数の多い数値ほど精密な近似値である。

真値Mの近似値mの精度は真値に対する誤差 $|\varepsilon|$ の割合で表される。相対誤差 $|\varepsilon|$ の逆数pを精度と定義する。

コンピュータでは数値の精度を有効数字の桁数で表すことが多い。実際の数値計算では数値の有効桁数が多いほどその近似値は真値に近く正確である。桁数が多いと計算が面倒になり、計算時間も長くなるから下位の桁を適当に省略する。

例題演習

浮動小数点演算において，絶対値の大きな数と絶対値の小さな数の加減算を行ったとき，絶対値の小さな数の有効けたの一部又は全部が結果に反映されないことを何というか。次のア～エの中から選べ。

ア　打切り誤差　　イ　けた落ち　　ウ　情報落ち　　エ　絶対誤差

解答解説

情報落ちに関する問題である。

アの打ち切り誤差は、無限回計算すれば正しい値が得られる計算を途中で打ち切ったために発生する誤差である。

イの桁落ちは、絶対値のほぼ等しい２つの数の絶対値の差を求めたときに、有効桁数が減じてアンダーフローの現象を起こすことである。

ウの情報落ちは、絶対値の大きな値と小さな値を加算するとき、指数部を揃えるときに小さな値が右に大きくシフトし、表現可能な範囲をはずれて無視されることにより、発生する誤差である。求める答えはウとなる。

エの絶対誤差は、真の値と実測値の差である。

例題演習

次の10進小数のうち、２進数で表現すると無限小数になるものはどれか。ア～エの中から選べ。

ア　0.25　　イ　0.45　　ウ　0.5　　エ　0.75

解答解説

無限小数に関する問題である。

ア～エの10進数小数を２進数に変換すると

アの0.25は0.01となる。

イの0.45は0.0111001100……となり、循環小数となる。求める答えはイとなる。

ウの0.5は0.1となる。

エの0.75は0.11となる。

例題演習

10進小数0.1 を次のような２進10ビット固定小数点方式の小数.0001100110に変換したときに生じる相対丸め誤差はおよそ幾らか。ア～エの中から選べ。

ア　0.004%　　イ　0.04%　　ウ　0.4%　　エ　４%

解答解説

丸め誤差に関する問題である。

２進数の.0001100110を10進数に変換すると、

$2^{-4}+2^{-5}+2^{-8}+2^{-9}=0.0625+0.03125+0.0039062+0.0019531=0.0996093$

従って、相対誤差は

$(0.1-0.0996093)/0.1=0.003907=0.004$

0.4(%)となる。求める答えはウとなる。

例題演習

有効けた数４けたで次の10進浮動小数点演算を行うとき、発生する誤差に関する記述のうち、正しいものはどれか。ア～エの中から選べ。ここで、計算は加算、減算の順に行うものとする。

計算式　1234＋1.987－1233

ア　加算、減算ともにけた落ちが生じる。
イ　加算、減算ともに情報落ちが生じる。
ウ　加算でけた落ちが、減算で情報落ちが生じる。
エ　加算で情報落ちが、減算でけた落ちが生じる。

解答解説

発生する誤差の種類に関する問題である。

浮動小数点演算における誤差の発生は、情報落ち、桁あふれ、桁落ちがある。ここで問題になっているのは情報落ちと桁落ちである。

情報落ち：絶対値の大きな値と小さな値を加算すると、指数部を揃えるときに小さな値が無視される。
桁落ち　：ほぼ等しい値を減算すると結果が小さい値になり表現不能になる。
桁あふれ：絶対値の大きい範囲の加算では、オーバフローが発生しやすく、絶対値の小さい範囲の加算ではアンダーフローが発生しやすい。

加算による誤差の発生は情報落ちであり、減算による誤差の発生は桁落ちである。計算の順序は加算、減算であるから、情報落ち、桁落ちになる。求める答えはエとなる。

例題演習

1,000個の実数値のデータをコンピュータを使用して浮動小数点演算で加算するとき、計算誤差を最も小さくするものはどれか。

ア　すべてのデータを降順に並べ替え、先頭から順に加える。
イ　すべてのデータを昇順に並べ替え、先頭から順に加える。
ウ　すべてのデータを絶対値の降順に並べ替え、先頭から順に加える。
エ　すべてのデータを絶対値の昇順に並べ替え、先頭から順に加える。

解答解説

計算機の誤差に関する問題である。

浮動小数点演算は指数部の値を合わせるとき、仮数部をビットシフトする。この場合、加算する２つの数値の指数部の値の差が大きいと、小さい方の数値が情報落ちになり誤差となる。絶対値の大きい数値から計算を進めると、小さい数値を加算する段階で情報落ちが生じる。従って、絶対値の小さいものから加算すると誤差が小さくなる。絶対値の大きい範囲の加算では、オーバフローが発生しやすく、絶対値の小さい範囲の加算ではアンダーフローが発生しやすい。アンダフローよりオーバフローの方が誤差が大きい。従って、絶対値の小さい方から計算する。絶対値の小さいものから大きいものへ計算し、加算、減算を組み合わせ計算する方式が最も誤

差が小さくなる。従って、絶対値の昇順に並べ替えて計算する。

アは、正数の絶対値の大きいものから計算をはじめ、負数の絶対値の大きいものへと計算をしていく手順であり、正数の領域では情報落ちが発生し、更にアンダーフローやオーバフローが発生しやすく誤差が生じやすい。

イは、負数の絶対値の大きいものから計算を初めて、正数の絶対値の大きいものへと計算をしていく手順であり、負数の領域で情報落ちが発生し、更にアンダーフローやオーバフローが発生しやすく誤差が生じやすい。

ウは、正数負数に関係なく、絶対値の大きいものから順に正数は加算、負数は減算する方法であり、情報落ちやオーバフローの誤差が生じやすい。

エは、正数負数の関係なく、絶対値の小さいものから順次大きいものへ、正数は加算し、負数は減算する計算法で誤差の発生が最も小さい。求める答えはエとなる。

例題演習

数多くの数値の加算を行う場合，絶対値の小さなものから順番に計算するとよい。これは，どの誤差を抑制する方法を述べたものか。ア～エの中から選べ。

ア　アンダフロー　　イ　打切り誤差

ウ　けた落ち　　エ　情報落ち

解答解説

情報落ちを防ぐ処置に関する問題である。

情報落ちを防ぐためには次の処置が必要である。

①　加算を行う場合に差の少ない数値の順に加算する。

②　絶対値の小さい値から加算する。

絶対値の小さい順に計算するため情報落ちの防止である。求める答えはエとなる。

txt01013 集合と命題

txt010131 集合と論理

① 集合

ⓐ 集合とは

集合は明確な基準が与えられている事実またはものの集まりのことである。例えば、１桁の正の奇数の集まり１、３、５、７、９は１つの集合である。１桁の正の奇数という範囲のはっきりした数値の集まりで一つの集合が形成されている。

ⓑ 集合の要素

集合を構成している１つ１つのものを、その集合の要素または元という。１桁の正の奇数の集合の要素は、１、３、５、７、９の各数値である。

ⓒ 集合の表現

集合Ａを構成する要素がａの場合、ａ∈Ａで表し、ａは集合Ａに属することを意味する。

集合を表す方法には、次の２つの方法がある。

㋐ 要素を１つ１つ書き並べる。

１桁の正の奇数の集合Ａは、Ａ＝{１、３、５、７、９}

㋑ 要素の満たす条件を示す。

１桁の正の奇数の集合Ａは、Ａ＝{ｘ｜１≦ｘ≦９，ｘは奇数}

または　　Ａ＝{２ｎ－１｜１≦ｎ≦５、ｎは整数}

集合の要素の個数が多い場合や無限の要素がある場合は、省略記号…を用いて、次のように表す。

100以下の正の奇数の集合Ａは、Ａ＝{１、３、５、…97、99}

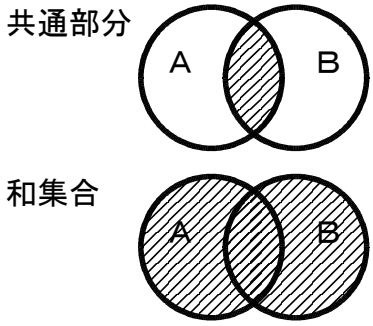
正の偶数全体の集合Ａは、Ａ＝{２、４、６、…}

ⓓ 部分集合、共通部分、和集合

集合Ａが集合Ｂに含まれる(または集合Ｂが集合Ａを含む)場合、集合Ａを集合Ｂの部分集合といい、Ａ⊂Ｂで表す。集合Ａと集合Ｂの関係を包含関係という。

２つの集合Ａ、Ｂについて、集合Ａ、Ｂの両方に属する要素の集合を、ＡとＢの共通部分といい、記号Ａ∩Ｂで表す。また、集合Ａ、Ｂの少なくとも一方に属する要素の集合を、ＡとＢの和集合といい、記号Ａ∪Ｂで表す。

㋐　２つの集合

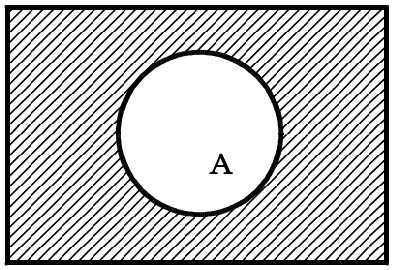


全体集合Ｕのうち、集合Ａに属さない要素の集合を、Ａの補集合といい、記号$\overline{A}$または¬Ａで表す。

要素を１つも持たない集合を空集合という。空集合は「∅」で表す。空集合は全ての集合の部分集合である。

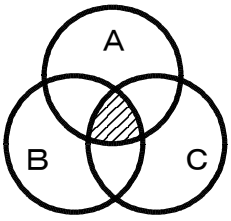
要素の個数が有限である集合を有限集合、無限の要素からなる集合を無限集合という。

㋑　補集合

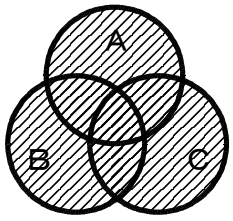


３つの集合Ａ、Ｂ、Ｃについて、Ａ、Ｂ、Ｃの全てに属する要素の集合をＡ∩Ｂ∩Ｃで表し、Ａ、Ｂ、Ｃの少なくとも１つに属する要素の集合をＡ∪Ｂ∪Ｃで表す。

㋒　３つの集合



共通部分



少なくとも１つの集合
に属する要素の集合

３つの集合について、Ａ∩Ｂ∪Ｃのような表現は使用しない。（Ａ∩Ｂ）∪Ｃ、Ａ∩（Ｂ∪Ｃ）のいずれの意味なのかが不明であり、両者のベン図は異なる。

集合に関するド・モルガンの表現は次のようになる。

$\overline{A \cap B} = \overline{A} \cup \overline{B}$　または　$\overline{A \cup B} = \overline{A} \cap \overline{B}$

ⓔ　集合の演算

㋐　Ａ∪Ｂ　集合Ａと集合Ｂの和集合
㋑　Ａ∩Ｂ　集合Ａと集合Ｂの積集合
㋒　Ａ－Ｂ　集合Ａの要素であるが集合Ｂの要素でない集合、つまりＡからＢの要素を取り去った残りの集合、差集合である。
㋓　$\overline{A}$　ある全体集合Ｕの部分集合Ａがあるとき、Ｕに属しているがＡには属さない要素の集合、集合Ａの補集合

ⓕ　集合の諸法則

㋐　べきの法則　Ａ∪Ａ＝Ａ　　Ａ∩Ａ＝Ａ
㋑　交換の法則　Ａ∪Ｂ＝Ｂ∪Ａ　　Ａ∩Ｂ＝Ｂ∩Ａ
㋒　結合の法則　(Ａ∪Ｂ)∪Ｃ＝Ａ∪(Ｂ∪Ｃ)
　　(Ａ∩Ｂ)∩Ｃ＝Ａ∩(Ｂ∩Ｃ)
㋓　分配の法則　Ａ∩(Ｂ∪Ｃ)＝(Ａ∩Ｂ)∪(Ａ∩Ｃ)
　　Ａ∪(Ｂ∩Ｃ)＝(Ａ∪Ｂ)∩(Ａ∪Ｃ)

ⓖ　集合の要素の個数

有限集合Ａの要素の個数をｎ(Ａ)で表すと、Ａ、Ｂが有限集合の時、次の関係が成り立つ。

㋐　和集合の要素の個数

ｎ(Ａ∪Ｂ)＝ｎ(Ａ)＋ｎ(Ｂ)－ｎ(Ａ∩Ｂ)

Ａ∩Ｂが空集合の時

ｎ(Ａ∪Ｂ)＝ｎ(Ａ)＋ｎ(Ｂ)

㋑　補集合の要素の個数

$n(\overline{A})=n(U)-n(A)$

Ｕは全体集合、$\overline{A}$はＡの補集合

②　集合と論理

ⓐ　命題と条件、命題の真偽

式や文章で表された事柄で、正しいか正しくないかが明確に決まるものを命題という。命題が正しいとき、その命題は真であるといい、正しくないとき、その命題は偽であるという。命題は真偽を判定する基準が明示されていなければならない。

２つの条件ｐ、ｑについて、命題「ｐならばｑ」をｐ⇒ｑで表し、ｐをこの命題の仮定、ｑをこの命題の結論という。

「ｐならばｑ」かつ「ｑならばｐ」をｐ⇔ｑで表す。

ⓑ　条件と集合

２つの条件ｐ、ｑを満たす全体の集合を、それぞれＰ、Ｑとすると、次の集合の関係が成り立つ。

「命題ｐ⇒ｑが真である」とき、条件ｐを満たすものは必ずｑを満たすからＰ⊂Ｑが成り立ち、逆に、Ｐ⊂Ｑならば、ｐ⇒ｑが真であることがいえる。

「命題ｑ⇒ｐが真である」とき、条件ｑを満たすものは必ずｐを満たすからＱ⊂Ｐが成り立ち、逆に、Ｑ⊂Ｐならば、ｑ⇒ｐが真であることがいえる。

「命題ｐ⇔ｑが真である」とき、Ｐ＝Ｑが成り立つ。

「命題ｐ⇒ｑが偽である」とき、Ｐの中にｑを満たさない要素が少なくとも１つある。この満たさない要素を反例という。

ⓒ　条件の合成と否定

「でない」、「かつ」、「または」を用いて作られる条件と集合は、全体集合をＵとし、ｐ、ｑを満たす全体の集合をＰ、Ｑとすると、次のようになる。

㋐　Ｐ∩Ｑ

条件ｐ、ｑが共に成り立つ、即ち、ｐが真かつｑが真の場合、Ｐ∩Ｑで表す。

㋑　Ｐ∪Ｑ

条件ｐ、ｑの少なくとも一方が成り立つ、即ち、ｐが真またはｑが真の場合、Ｐ∪Ｑで表す。

㋒　$\overline{P \cap Q} = \overline{P} \cup \overline{Q}$

ド・モルガンの法則から、共通部分Ｐ∩Ｑの否定は、集合ＰまたはＱのどちらか一方が否定された場合である。従って、ｐかつｑの否定は、ｐの否定またはｑの否定となる。

㋓　$\overline{P \cup Q} = \overline{P} \cap \overline{Q}$

ド・モルガンの法則から、和集合Ｐ∪Ｑの否定は、集合Ｐの否定および集合Ｑの否定が同時に成り立つ場合である。従って、ｐとｑの和集合の否定は、ｐの否定かつｑの否定となる。

ⓓ　必要条件、十分条件

２つの条件ｐ、ｑにおいて、ｐ⇒ｑが真であるとき、ｑはｐであるための必要条件、ｐはｑであるための十分条件である。

ｐ⇒ｑとｑ⇒ｐが共に真であるとき、即ち、ｐ⇔ｑが成り立つとき、ｑはｐであるための、または、ｐはｑであるための必要十分条件であるという。このとき、ｐとｑは互いに同値であ

るという。

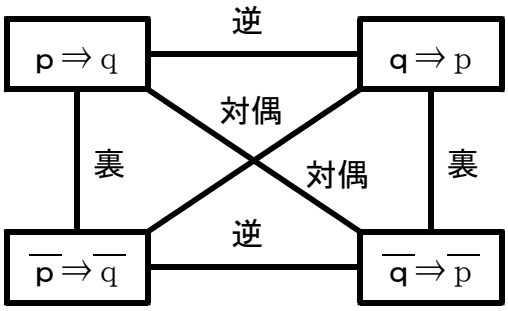
ⓔ　「すべて」と「ある」とその否定

全体集合をＵ、条件ｐを満たすｘ全体の集合をＰとする。

㋐　Ｐ＝Ｕの時、命題「すべてのｘについてｐである」は真である。
㋑　Ｐ≠∅ の時、命題「あるｘについてｐである」は真である。
㋒　命題「すべてのｘについてｐである」の否定は「あるｘについて$\overline{\text{ｐ}}$である」
㋓　命題「あるｘについてｐである」の否定は「すべてのｘについて$\overline{\text{ｐ}}$である」

ⓕ　逆、裏、対偶

命題ｐ⇒ｑに対して、逆、裏、対偶は次の図のようになる。



命題の真偽とその対偶の真偽は一致する。ｐ⇒ｑの真偽を検討する場合、$\overline{\text{ｑ}}$⇒$\overline{\text{ｐ}}$の真偽を検討し解を求めることができる。

命題の真偽とその逆、裏の真偽は必ずしも一致しない。

ⓖ　背理法

背理法は、ある命題Ｘに対して、Ｘが成り立たないと仮定して、矛盾を導くことによって、Ｘが成り立つことを示す証明法である。「○○である」ことを証明するのに「○○でない」ことを仮定して矛盾を導く。

$\sqrt{6}$が無理数であることを証明する場合、$\sqrt{6}$が有理数であると仮定して、証明する。

$\sqrt{6}$＝ｐ／ｑ（ｐ、ｑは互いに素である自然数）とし、両辺を２乗すると

６＝ｐ^2／ｑ^2　　６ｑ^2＝ｐ^2　　左辺は偶数であるから、右辺も偶数になる。

ｐ＝２ｒとすると、６ｑ^2＝４ｒ^2　　３ｑ^2＝２ｒ^2　となり、左辺は偶数となり、ｑは偶数となる。

ｐ、ｑ共に偶数となり、ｐ、ｑ互いに素と矛盾する。従って、$\sqrt{6}$は無理数となる。

例題演習

集合ＡとＢについて，常に成立する関係はどれか。ア～エの中から選べ。ここで，∩は積集合，∪は和集合，$\overline{A}$はＡの補集合，Ａ⊆Ｂは“ＡはＢの部分集合である”ことを表す。

ア　$A \subseteq (A \cap \overline{B})$　　イ　$(A \cup B) \subseteq (\overline{A} \cup \overline{B})$

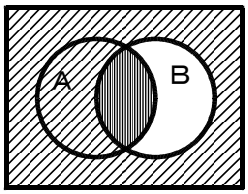
ウ　$(A \cap B) \subseteq (A \cup \overline{B})$　　エ　$(A \cap B) \subseteq (\overline{A} \cap \overline{B})$

解答解説

アの場合、ＡＢの共通部分が$A \cap \overline{B}$の外部となり、Ａは$A \cap \overline{B}$の部分集合とならない。

イの場合、$A \cup B$は$\overline{A} \cup \overline{B}$の外部となり、$A \cup B$は$\overline{A} \cup \overline{B}$の部分集合とならない。

ウの場合、$A \cap B$は$A \cup \overline{B}$の部分集合となる。次のベン図で$A \cap B$は縦線の部分、$A \cup \overline{B}$は斜線と縦線の部分を表す。求める答はウとなる。



エの場合、$\overline{A} \cap \overline{B}$には$A \cap B$の共通部分が含まれないため部分集合にならない。

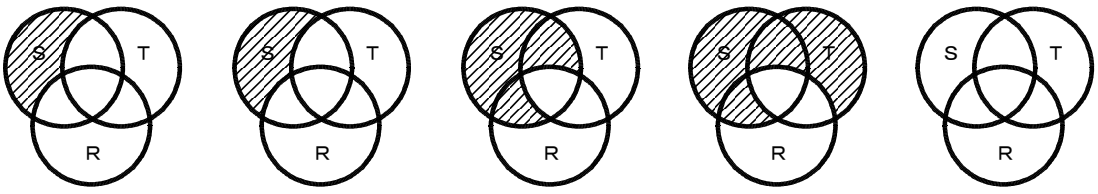
例題演習

集合Ｓ－(Ｔ∪Ｒ)に等しいものはどれか。ア～エの中から選べ。ここで，∩は積集合，∪は和集合，－は差集合の各演算を表す。

ア　(Ｓ－Ｔ)－Ｒ　　イ　(Ｓ－Ｔ)∪(Ｓ－Ｒ)

ウ　(Ｓ－Ｔ)∪(Ｔ－Ｒ)　　エ　(Ｓ－Ｔ)∩(Ｔ－Ｒ)

解答解説



Ｓ－(Ｔ∪Ｒ)　ア　イ　ウ　エ

次に与えられた集合と４つの解答のベン図を求めると図のようになる。

エの場合は満足する領域が存在しない。Ｓ－(Ｔ∪Ｒ)と一致するのはアのベン図である。求める答えはアとなる。

例題演習

差集合、Ｓ－Ｔに等しいものはどれか。ここで，∪は和集合，∩は積集合，$\overline{X}$はＸの補集合の各演算を表す。

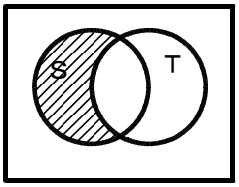
ア　Ｓ∪(Ｓ∩Ｔ)　　イ　Ｓ∪$\overline{T}$

ウ　Ｓ∩(Ｓ∪Ｔ)　　エ　Ｓ∩$\overline{T}$

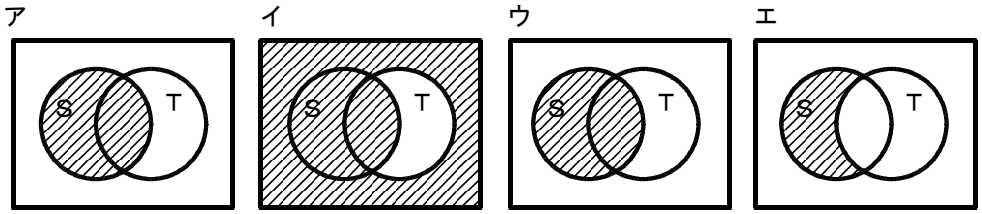
解答解説

集合に関する問題である。

Ｓ－Ｔのベン図を描くと、右図のようになる。



ア～エのベン図を描くと、図のようになる。



アはＳ、Ｔの共通部分とＳの和であるから、Ｓとなる。

イはＳとＴの否定の和であるから、イの図になる。

ウはＳとＴの和とＳの共通部分であるからウの図となり、Ｓ－Ｔの図と一致する。

エはＳとＴの否定の共通部分で、Ｓ－Ｔの図と一致する。求める答えはエとなる。

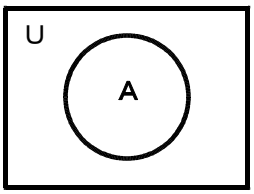
txt010132 ベン図とその利用

① ベン図

ⓐ ベン図とは

ベン図は長方形とその中に描いた円形を利用して、論理変数の真と偽を図解する方法である。長方形は真偽全体の領域を表し、円形の内部は論理変数の真の領域、円の外部は論理変数の偽の領域を表す。

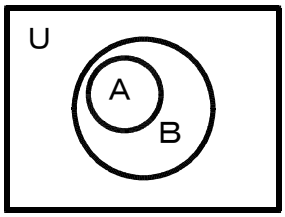
ⓑ 論理変数Ａのベン図



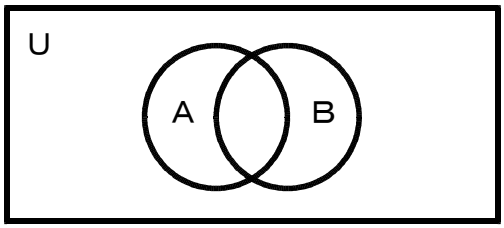
Ｕ：論理変数Ａの真と偽の領域
Ａ：円の内部は論理変数Ａの真の領域
　　円の外部は論理変数Ａの偽の領域

ⓒ 二つの論理変数Ａ、Ｂのベン図

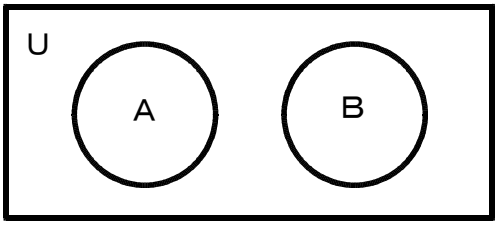
㋐ ＡがＢに属する場合



㋑ ＡとＢが交わる場合



㋒ ＡとＢが交わらない場合



②　ベン図の利用

ⓐ　ベン図を論理式の証明に利用

同じベン図を利用して、図の見方を変えることによって論理式の証明ができる。

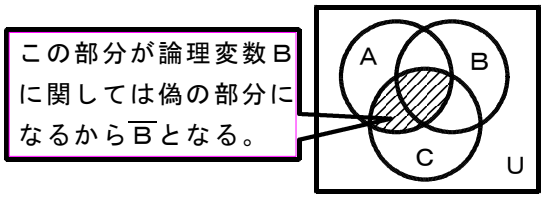
以下に、具体例を使用して、３つの論理変数Ａ、Ｂ、Ｃからなる２または３の異なる論理式が同じベン図で表すことができることを証明する。

ⓑ　論理式の証明の具体例

論理式の（１）は、斜線の部分を２分割して、それぞれの部分を３つの論理変数Ａ、Ｂ、Ｃの論理積として表し、その論理積の論理和として表した論理式である。

論理式の（２）は、論理変数Ｂに関しては斜線の部分に真と偽の領域が存在するため論理変数Ｂは無視できて、論理変数Ａと論理変数Ｃの論理積で表すことができる。

論理式の（１）、（２）は、同じ斜線の部分を表しているため２つの論理式は等しいことになる。



（１）　Ａ・$\overline{B}$・Ｃ＋Ａ・Ｂ・Ｃ

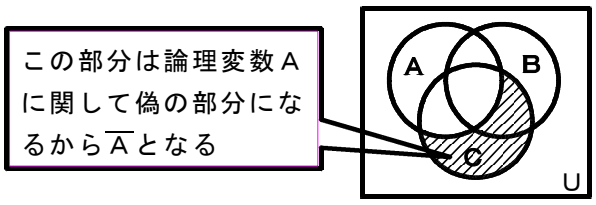
（２）　Ａ・Ｃ

ⓒ　論理式の証明の具体例

（１）の論理式は、論理変数Ｂに関しては斜線の部分に真と偽の領域があるため論理変数Ｂは無視できて、論理変数Ａと論理変数Ｃの論理積で表わすことができる。

（２）の論理式は、斜線の部分を２分割してそれぞれの部分を論理変数Ａ、Ｂ、Ｃの論理積として表し、２つの部分の論理和として表した論理式である。

（１）、（２）の論理式は同じ斜線の部分を表しているため等しくなる。



（１）　$\overline{A}$・Ｃ

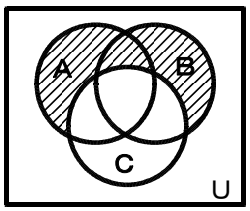
（２）　$\overline{A}$・Ｂ・Ｃ＋$\overline{A}$・$\overline{B}$・Ｃ

ⓓ　論理式の証明の具体例

（１）の論理式は、一方は論理変数Ｂに関しては斜線の部分に真と偽の領域があるため論理変数Ｂは無視できて、論理変数Ａと論理変数Ｃの偽の論理積で表わすことができ、もう一方は論理変数Ａに関しては斜線の部分に真と偽の領域があるため論理変数Ａは無視できて、論理変数Ｂと論理変数Ｃの偽の論理積で表わすことができ、両者の論理和で表せる。

（２）の論理式は、斜線の部分を３つの部分に分割して、それぞれの部分を論理変数Ａ、Ｂ、Ｃの論理積を求め、その３つの部分の論理和を求めた論理式である。

（1）、（2）の論理式は同じ斜線の部分を表しているため等しくなる。



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

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

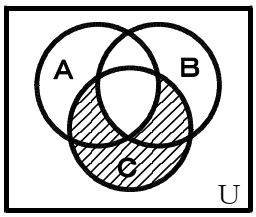
ⓔ　論理式の証明の具体例

（1）の論理式は、論理変数Ａの偽とＢの偽の論理和を求め、その論理和の部分と論理変数Ｃの論理積を表したものである。

（2）の論理式は、斜線の部分を３つの部分に分割して、それぞれの部分を論理変数Ａ、Ｂ、Ｃの論理積として表し、３つの論理積の部分の論理和として表したものである。

（3）の論理式は、論理変数Ａの偽とＣの真の論理積と論理変数Ｂの偽とＣの真の論理積との論理和を表したものである。

（1）、（2）、（3）の論理式は同じ斜線の部分を表しているため等しくなる。



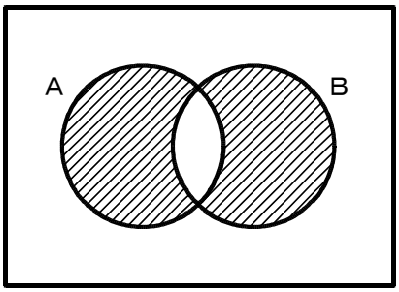
（1）　$(\overline{A} + \overline{B}) \cdot C$

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

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

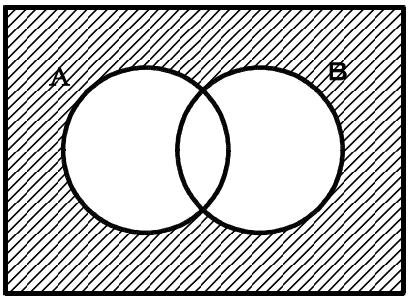
ⓕ　ドモルガンの法則とベン図

㋐　$\overline{A \cap B} = \overline{A} \cup \overline{B}$



Ａ∩Ｂの集合は、ベン図の斜線がない部分である。その否定が$\overline{A \cap B}$であるから、ベン図の斜線部分は$\overline{A \cap B}$を表す。一方、$\overline{A}$は集合Ａの円の外側であり、$\overline{B}$は集合Ｂの円の外側であるから、$\overline{A}$と$\overline{B}$の和集合はベン図の斜線部分になる。

㋑ $\overline{A \cup B} = \overline{A} \cap \overline{B}$

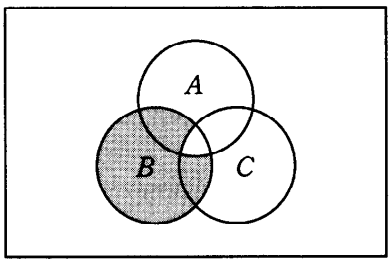


$A \cup B$の集合は、ベン図の斜線がない部分である。その否定が$\overline{A \cup B}$であるから、ベン図の斜線部分は$\overline{A \cup B}$を表す。一方、$\overline{A}$は集合Ａの円の外側であり、$\overline{B}$は集合Ｂの円の外側であるから、$\overline{A}$と$\overline{B}$の共通部分はベン図の斜線部分になる。

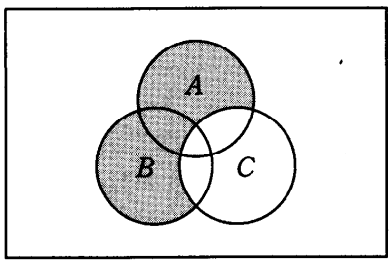
例題演習

集合$(\overline{A} \cap B \cap C) \cup (A \cap B \cap \overline{C})$を網掛け部分（　　）で表しているベン図どれか。ア～エの中から選べ。ここで，∩は積集合，∪は和集合，$\overline{X}$はＸの補集合を表す。

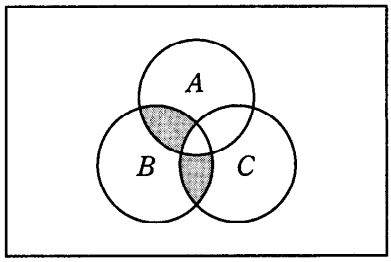
ア



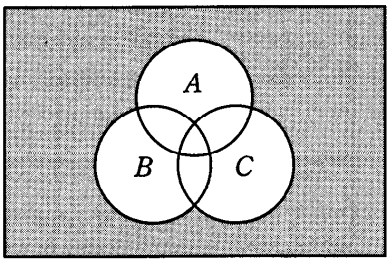
イ



ウ



エ



解答解説

図に示すように、Ａ、Ｂの共通の斜線の部分は、円Ａ、Ｂの中にあり、円Ｃの外にある。従って

$A \cap B \cap \overline{C}$

となり、Ｂ、Ｃの共通の斜線の部分は

$\overline{A} \cap B \cap C$

となる。従って、斜線の部分は

$(\overline{A} \cap B \cap C) \cup (A \cap B \cap \overline{C})$

となり、求める答えはウとなる。

ベン図から論理式を求める方法は次の通りになる。

① ３論理変数Ａ、Ｂ、Ｃの場合、３変数の論理積を作成する。$A \cap B \cap C$

② 各論理変数について、斜線部分が円の内部にあれば肯定で何もしない、外部にあれば否定に変更すると、２カ所の斜線部分について考えると、$\overline{A} \cap B \cap C$、$A \cap B \cap \overline{C}$となる。

③ 斜線部分を論理和で結びつける。$(\overline{A} \cap B \cap C) \cup (A \cap B \cap \overline{C})$

解答例の論理式を作成すると次のようになる。

アは斜線部分は３カ所で、$(\overline{A} \cap B \cap C) \cup (A \cap B \cap \overline{C}) \cup (\overline{A} \cap B \cap \overline{C})$となる。

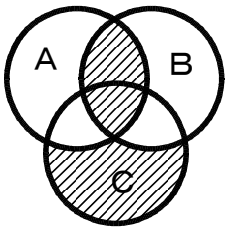
イは斜線部分は３カ所で、$(A \cap \overline{B} \cap \overline{C}) \cup (A \cap B \cap \overline{C}) \cup (\overline{A} \cap B \cap \overline{C})$となる。

ウは斜線部分は２カ所で、$(\overline{A} \cap B \cap C) \cup (A \cap B \cap \overline{C})$となる。

エは斜線部分は１カ所で、$\overline{A} \cap \overline{B} \cap \overline{C}$となる。

例題演習

ベン図の網掛け部分で表現される集合はどれか。ここで、$X \cup Y$はＸとＹの和集合、$X \cap Y$はＸとＹの積集合、¬Ｘまたは$\overline{X}$はＸの補集合を表す。



ア $(A \cup B) \cap C$

イ $(A \cap B) \cup (C \cap \overline{A \cup B})$

ウ $\overline{(\neg A \cap \neg B) \cap C}$

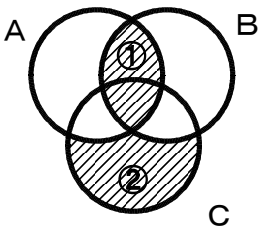
エ $\overline{C} \cap (A \cup B)$

解答解説

ベン図の①の部分はＡとＢの論理積になる。即ち、$A \cap B$となる。

ベン図の②の部分は$\overline{A \cup B}$とＣの論理積になる。従って、ベン図の斜線の部分は次の論理式になる。

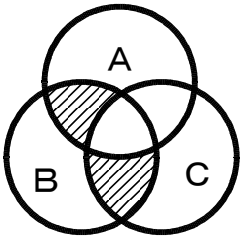
$(A \cap B) \cup (\overline{A \cup B} \cap C)$



求める答えはイとなる。

例題演習

次のベン図の網掛け部分（▨）の集合を表す式はどれか。ここで，ＸＵＹはＸとＹの和集合，Ｘ∩ＹはＸとＹの積集合，$\overline{X}$はＸの補集合を表す。



ア　$\overline{(A \cap B \cap C)} \cap B$

イ　$(A \cup B) \cap \overline{C}$

ウ　$(\overline{A} \cap B \cap C) \cup (A \cap B \cap \overline{C})$

エ　$\overline{A \cup B \cup C}$

解答解説

ベン図に関する問題である。

図に示すように、Ａ、Ｂの共通の斜線の部分は、円Ａ、Ｂの中にあり、円Ｃの外にある。従って

$A \cap B \cap \overline{C}$

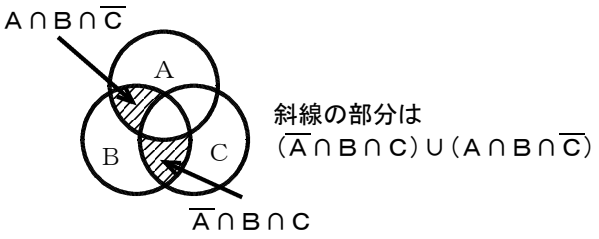
となり、Ｂ、Ｃの共通の斜線の部分は

$\overline{A} \cap B \cap C$

となる。従って、斜線の部分は

$(\overline{A} \cap B \cap C) \cup (A \cap B \cap \overline{C})$

となり、求める答えはウとなる。



例題演習

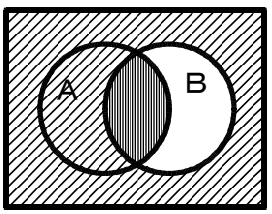
集合ＡとＢについて，常に成立する関係はどれか。ここで，∩は積集合，∪は和集合，$\overline{A}$はＡの補集合，Ａ⊆Ｂは“ＡはＢの部分集合である”ことを表す。

ア　$A \subseteq (A \cap \overline{B})$　　イ　$(A \cup B) \subseteq (\overline{A} \cup \overline{B})$

ウ　$(A \cap B) \subseteq (A \cup \overline{B})$　　エ　$(A \cap B) \subseteq (\overline{A} \cap \overline{B})$

解答解説

集合に関する問題である。



アの場合、ＡＢの共通部分が$A \cap \overline{B}$の外部となり、Ａは部分集合とならない。

イの場合、Ａ∪Ｂは$\overline{A} \cup \overline{B}$の外部となり、部分集合とならない。

ウの場合、Ａ∩Ｂは$A \cup \overline{B}$の部分集合となる。右のベン図でＡ∩Ｂは縦線の部分、$A \cup \overline{B}$は斜線と縦線の部分を表す。求める答はウとなる。

エの場合、$\overline{A} \cap \overline{B}$にはＡ∩Ｂの共通部分が含まれないため部分集合にならない。

txt01014 論理演算

txt010141 論理演算の基礎

① 論理演算と真理値表

ⓐ 論理演算とは

真または偽のいずれかの値をとる変数を論理変数という。２つ以上の論理変数の間で行われる論理和や論理積、排他的論理和などの演算を論理演算という。論理定数は真または偽のことである。

ⓑ 論理展開の手法

㋐ 論理公式
㋑ 真理値表
㋒ ベン図
㋓ ベッチ・カルノー図

ⓒ 真理値表

真理値表は、論理変数の取り得るすべての条件とコンピュータの基本演算である否定、論理和、論理積、排他的論理和などの論理演算の結果を表形式で表したものである。２つの論理変数Ａ、Ｂと論理演算の結果の間には次の関係が成立する。表中１は真、０は偽を表す。

A	B	$\overline{A}$	A∨B	A∧B	A∀B
0	0	1	0	0	0
0	1	1	1	0	1
1	0	0	1	0	1
1	1	0	1	1	0

ⓓ 論理演算を表す論理演算子

論理演算	論理演算子
論理和	∨または＋
論理積	∧または・
排他的論理和	∀または⊕
論理否定	¬または￣

ⓔ　演算子の優先順位

括弧、算術演算子（べき乗、積・商、和・差）、論理演算子（論理否定、論理積、論理和、排他的論理和）の順に高い。同じ優先順位のものは、左側から先に実行する。

②　論理公式

ⓐ　論理公式

論理公式は集合と集合の演算あるいは集合に関する法則を示したものである。

ⓑ　主要な論理公式

㋐　べき等則

$A \vee A = A$　　$A \wedge A = A$　　$A \forall A = 0$

㋑　交換則

$A \vee B = B \vee A$　　$A \wedge B = B \wedge A$

㋒　分配則

$A \wedge (B \vee C) = (A \wedge B) \vee (A \wedge C)$

$A \vee (B \wedge C) = (A \vee B) \wedge (A \vee C)$

㋓　吸収則

$A \vee (A \wedge B) = A$　　$A \wedge (A \vee B) = A$

$A \vee (\overline{A} \wedge B) = A \vee B$　　$A \wedge (\overline{A} \vee B) = A \wedge B$

㋔　復元則

$\neg\neg A = A$

㋕　ド・モルガンの法則

$\overline{A \vee B} = \overline{A} \wedge \overline{B}$　　$\overline{A \wedge B} = \overline{A} \vee \overline{B}$

㋖　重要な論理公式

$A \vee 1 = 1$　　$A \vee 0 = A$　　$A \vee \overline{A} = 1$

$A \wedge 1 = A$　　$A \wedge 0 = 0$　　$A \wedge \overline{A} = 0$

ⓒ　ド・モルガンの法則の証明

真理値表を使用してド・モルガンの法則の$\overline{A \vee B} = \overline{A} \wedge \overline{B}$を証明すると、表のようになる。

A	B	A∨B	$\overline{A \vee B}$	$\overline{A}$	$\overline{B}$	$\overline{A} \wedge \overline{B}$
0	0	0	1	1	1	1
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	0

$\overline{A \vee B}$のビットパターン1000、$\overline{A} \wedge \overline{B}$のビットパターン1000が一致する。２つの論理変数Ａ、Ｂの論理和の否定は、論理変数Ａ、Ｂのそれぞれの否定の論理積に等しい。

$\overline{A \wedge B} = \overline{A} \vee \overline{B}$についても同様にして証明できる。

ⓓ　双対の原理

二つの論理式で、一方が他方の全ての論理積を論理和で、論理和を論理積で、または０を１で、１を０で置換した表現であるとき、両者の論理式は互いに双対であるという。互いに双対な論理式は、一方が成立すると他方も成立する。これを双対の原理という。

具体例

Ａ∨０＝ＡとＡ∧１＝Ａ

Ａ∨１＝１とＡ∧０＝０

Ａ∨Ａ＝ＡとＡ∧Ａ＝Ａ

Ａ∨Ｂ＝Ｂ∨ＡとＡ∧Ｂ＝Ｂ∧Ａ

Ａ∨Ａ∧Ｂ＝ＡとＡ∧（Ａ∨Ｂ）＝Ａ

ⓔ　論理公式を利用した論理式の証明方法

左辺の論理式または右辺の論理式を論理公式を用いて変形し、右辺の論理式または左辺の論理式を導く。左辺、右辺の論理式を論理公式を用いて共に変形し、左辺＝右辺を求める。

具体例

$A \cdot B + A \cdot C + A \cdot \overline{B} \cdot \overline{C} = A \cdot (B + C + \overline{B} \cdot \overline{C})$　［$C + \overline{C} \cdot \overline{B} = C + \overline{B}$］

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

$= A \cdot (B + C + \overline{B})$　［$B + \overline{B} = 1$］

$= A \cdot (1 + C)$　［$1 + C = 1$］

$= A$　［$A \cdot 1 = A$］

次のようにして求める。

㋐　３つの論理式を論理変数Ａで括る。

㋑　$C + \overline{C} \cdot \overline{B} = C + \overline{B}$ を利用して変形する。

㋒　$B + \overline{B} = 1$を利用して変形する。

㋓　１＋Ｃ＝１を利用して変形してＡを得る。

具体例

（Ａ＋Ｂ）・（Ｂ＋Ｃ）・（Ｃ＋$\overline{A}$）＝（Ａ＋Ｂ）・（Ｃ＋$\overline{A}$）

左辺＝（Ａ＋Ｂ）・（Ｂ＋Ｃ）・（Ｃ＋$\overline{A}$）
＝（Ａ・Ｂ＋Ａ・Ｃ＋Ｂ＋Ｂ・Ｃ）・（Ｃ＋$\overline{A}$）
＝Ａ・Ｂ・Ｃ＋Ａ・Ｃ＋Ｂ・Ｃ＋$\overline{A}$・Ｂ
＋$\overline{A}$・Ｂ・Ｃ　［ＢＣ（Ａ＋$\overline{A}$）＝ＢＣ］
＝Ｂ・Ｃ＋Ａ・Ｃ＋$\overline{A}$・Ｂ　［ＢＣ＋ＢＣ＝ＢＣ］

右辺＝Ａ・Ｃ＋Ａ・$\overline{A}$＋Ｂ・Ｃ＋Ｂ・$\overline{A}$
＝Ａ・Ｃ＋Ｂ・Ｃ＋Ｂ・$\overline{A}$

左辺＝右辺

次のようにして求める。
㋐　左辺の式の括弧を外すため、各項の論理積を求める。
Ａ・Ｃ・Ｃ＝Ａ・Ｃ、Ｂ・Ｃ＋Ｂ・Ｃ
＝Ｂ・Ｃ、Ａ・Ｂ・$\overline{A}$
＝０、Ａ・Ｃ・$\overline{A}$＝０
㋑　ＢＣ（Ａ＋$\overline{A}$）＝ＢＣ、ＢＣ＋ＢＣ＝ＢＣを利用して変形する。
㋒　右辺の式の括弧を外すため、各項の論理積を求める。Ａ・$\overline{A}$＝０
㋓　左辺＝右辺が得られる。

③　任意のビットパターンの桁別演算

ⓐ　８ビットの任意のビットパターンＸとＦＦの桁別論理演算

論理和はＸのすべてのビットが１となる。論理積はＸのビットは変化しない。排他的論理和はＸのすべてのビットは反転する。

具体例

	10110110		10110110		10110110
ＯＲ	11111111	ＡＮＤ	11111111	ＥＯＲ	11111111
	11111111		10110110		01001001

ⓑ　８ビットの任意のビットパターンＸと００の桁別論理演算

論理和はＸのビットは変化しない。論理積はＸのすべてのビットが０になる。排他的論理和はＸのビットは変化しない。

具体例

10110110	10110110	10110110
ＯＲ　00000000	ＡＮＤ　00000000	ＥＯＲ　00000000
10110110	00000000	10110110

ⓒ　８ビットの任意のビットパターンＸ同士の桁別論理演算

論理和はＸのビットは変化しない。論理積はＸのビットは変化しない。排他的論理和はＸのすべてのビットが０になる。

具体例

10110110	10110110	10110110
ＯＲ　10110110	ＡＮＤ　10110110	ＥＯＲ　10110110
10110110	10110110	00000000

ⓓ　８ビットの任意のビットパターンＸと80の桁別論理演算

論理和は、ＭＳＢ、Ｘのビットは変化しない。論理積は、ＭＳＢのビットは変化せず、他のビットは０になる。排他的論理和は、ＭＳＢのビットは反転し、他のビットは変化しない。

具体例

10110110	10110110	10110110
ＯＲ　10000000	ＡＮＤ　10000000	ＥＯＲ　10000000
10110110	10000000	00110110

④　マスキングとその利用

ⓐ　マスキングとは

マスキングはマスクを使ってビットパターンの一部を抽出する方法で、対象となるビットパターンとマスクとの間で桁別論理演算を行う。８ビットの任意のビットパターンと$(03)_{16}$の論理積を求めると、任意のビットパターンの下位２ビットのビットパターンを抽出できる。

具体例

ビットパターン10110011の最下位の２ビットを抽出する。

10110011　　上位６ビットはすべて０、下位２ビットはそのまま

ＡＮＤ　00000011

00000011

ⓑ　ビットパターンの抽出

８ビットのビットパターン10110100を下位のビットから順次２ビットずつ抽出して８ビットのビットパターンを形成する場合の手続きは次の手順で行う。

㋐　10110100と00000011の論理積を求める。

10110100∧00000011＝00000000

㋑　10110100を右に２ビットシフトして00101101を求める。

㋒　00101101と00000011の論理積を求める。

00101101∧00000011＝00000001

㋓　00101101を右に２ビットシフトして00001011を求める。

㋔　00001011と00000011の論理積を求める。

00001011∧00000011＝00000011

㋕　00001011を右に２ビットシフトして00000010を求める。

㋖　00000010と00000011の論理積を求める。

00000010∧00000011＝00000010

㋗　00000000、00000001、00000011、00000010の４個のビットパターンを求めることができる。

この４個のビットパターンから、ビットシフトと論理和を利用して、10110100の８ビットのビットパターンを合成することができる。

例題演習

真理値表の演算結果を表す論理式はどれか。ア～エの中から選べ。ここで，＋は論理和，・は論理積を表す。

x	y	z	演算結果
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

ア　（ｘ・ｙ）＋ｚ
イ　（ｘ＋ｙ）・ｚ
ウ　ｘ・（ｙ＋ｚ）
エ　ｘ＋（ｙ・ｚ）

解答解説

真理値と論理式に関する問題である。

解答群の論理式に対する真理値表を作成すると次のようになる。

x	y	z	演算結果	ア	イ	ウ	エ
0	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0
0	1	0	0	0	0	0	0
0	1	1	0	1	1	0	0
1	0	0	0	0	0	0	1
1	0	1	1	1	1	1	1
1	1	0	1	1	0	1	1
1	1	1	1	1	1	1	1

ビットパターンが一致するのはウの場合である。求める答えはウとなる。

例題演習

論理式 $\overline{(\overline{A}+B)\cdot(A+\overline{C})}$ と等しいものはどれか。ア～エの中から選べ。ここで，・は論理積，＋は論理和，$\overline{X}$はＸの否定を表す。

ア　$A\cdot\overline{B}+\overline{A}\cdot C$
イ　$\overline{A}\cdot B+A\cdot\overline{C}$
ウ　$(A+\overline{B})\cdot(\overline{A}+C)$
エ　$(\overline{A}+B)\cdot(A+\overline{C})$

解答解説

ド・モルガンの法則の応用に関する問題である。

ド・モルガンの法則を利用して、論理式を変形する。

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

となり、求める答えはアとなる。

例題演習

８ビットのビット列の下位４ビットが変化しない操作はどれか。

ア　16進表記０Ｆのビット列との論理積をとる。
イ　16進表記０Ｆのビット列との論理和をとる。
ウ　16進表記０Ｆのビット列との排他的論理和をとる。
エ　16進表記０Ｆのピット列との否定論理積をとる。

解答解説

桁別論理演算に関する問題である。

８ビットの２進数を10111101として、ア～エの演算を行うと次のようになる。

アの場合、10111101と00001111の論理積は、00001101となり、下位４ビットは変化しない。求める答えはアとなる。

イの場合、10111101と00001111の論理和は、10111101となる。

ウの場合、10111101と00001111の排他的論理和は、10110000となる。

エの場合、10111101と00001111の否定論理積は、11110010となる。

例題演習

８ビットで表される符号なし２進数ｘが16の倍数であるかどうかを調べる方法として，適切なものはどれか。

ア　ｘと２進数00001111のビットごとの論理積をとった結果が０である。
イ　ｘと２進数00001111のビットごとの論理和をとった結果が０である。
ウ　ｘと２進数11110000のビットごとの論理積をとった結果が０である。
エ　ｘと２進数11110000のビットごとの論理和をとった結果が０である。

解答解説

マスキングに関する問題である。

８ビットの２進数であるから、16の倍数は00010000～11110000の範囲の数値になる。00010000の場合で考えると

アの場合　00010000 AND 00001111＝00000000
イの場合　00010000 OR 00001111＝00011111
ウの場合　00010000 AND 11110000＝00010000
エの場合　00010000 OR 11110000＝11110000

従って、結果が０になるのは、ｘと２進数00001111のビットごとの論理積をとった場合である。求める答えはアとなる。

txt010142 論理回路

① 論理回路とＭＩＬ記号

ⓐ 論理回路とは

論理回路はＯＮまたはＯＦＦの入力信号を受けて、それに論理演算を施して出力する回路である。入力は１つであることもあるし、複数であることもある。論理回路の基本は、論理和演算、論理積演算、否定、排他的論理和演算である。

ⓑ ＭＩＬ記号

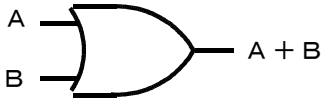
ＭＩＬ記号はアメリカの国防総省の軍機構で使用する規格である。論理回路の表現にＭＩＬ規格の論理記号が使用される。

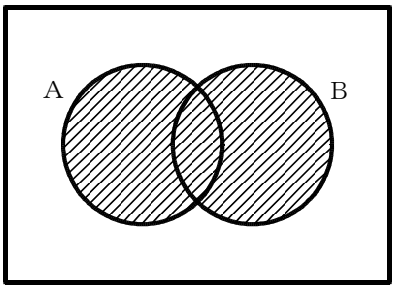
② 論理回路の種類

ⓐ 論理和

論理和は２つの論理変数の少なくとも一つが真の時、結果が真となる論理演算である。この論理演算を回路にしたものを論理和回路（ＯＲ回路）という。入力の一方または両方が１の時、出力が１になる構成である。「Ａ ＯＲ Ｂ」または、「＋」や「∪」という演算子を用いて「Ａ＋Ｂ」、「Ａ∪Ｂ」のように表す。真理値表、ＭＩＬ記号、ベン図を示すと、図のようになる。

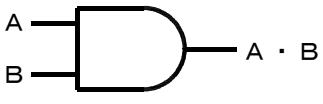
A	B	A＋B
0	0	0
0	1	1
1	0	1
1	1	1

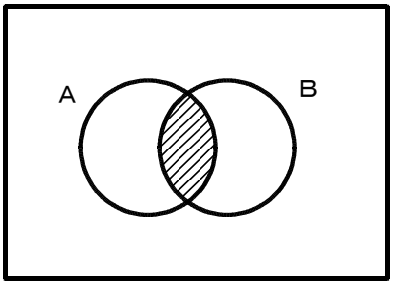




ⓑ 論理積

A	B	A・B
0	0	0
0	1	0
1	0	0
1	1	1



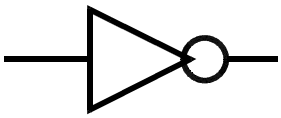


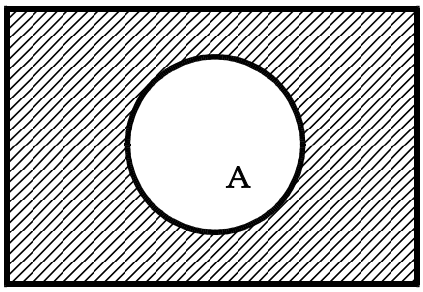
論理積は論理変数の両方が真の時、結果が真となる論理演算である。この論理演算を回路にしたものを論理積回路（ＡＮＤ回路）という。両方の入力が１の時だけ、出力が１となる構成である。「Ａ　ＡＮＤ　Ｂ」または、「・」や「∩」という演算子を用いて「Ａ・Ｂ」、「Ａ∩Ｂ」ように表す。真理値表、ＭＩＬ記号、ベン図を示すと、図のようになる。

ⓒ　否定

否定は論理変数が真の時結果は偽となり、偽の時結果は真となる論理演算である。この論理演算を回路にしたものを否定回路（ＮＯＴ回路）という。入力が１の時、出力は０、入力が０の時、出力は１となる。「ＮＯＴ　Ａ」または、「$\overline{A}$」、「¬Ａ」で表す。真理値表、ＭＩＬ記号、ベン図を示すと、図のようになる。

Ａ	$\overline{A}$
0	1
1	0

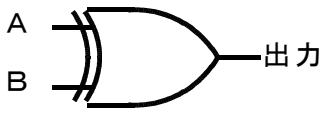


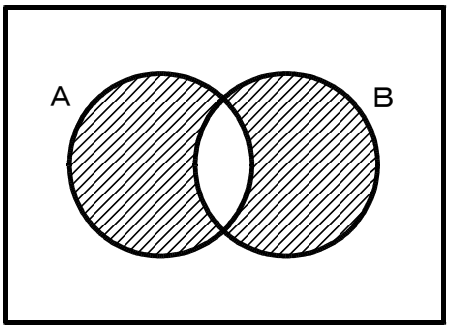


ⓓ　排他的論理和

排他的論理和は、論理変数のどちらか一方が真の時だけ、結果が真となる論理演算である。この論理演算を回路にしたものを排他的論理和回路という。ＡまたはＢのどちらか一方が１のとき、出力が１となる構成である。「Ａ　ＥＯＲ　Ｂ」または、「Ａ　∀　Ｂ」、「Ａ　・　Ｂ」で表す。真理値表、ＭＩＬ記号、ベン図を示すと、図のようになる。

Ａ	Ｂ	Ａ・$\overline{B}$＋$\overline{A}$・Ｂ
0	0	0
0	1	1
1	0	1
1	1	0

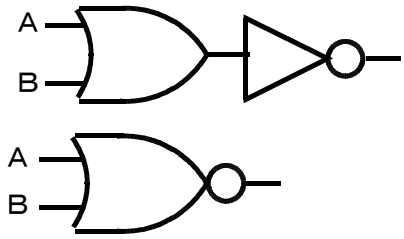


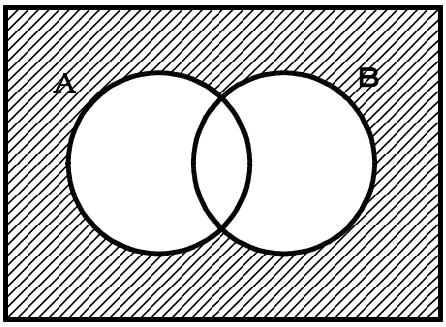


ⓔ　否定論理和

否定論理和は、２つの論理値を入力して論理和を否定した論理値を出力する。この論理演算を回路にしたものを否定論理和回路という。ＯＲ回路にＮＯＴ回路を接続したものである。ＯＲ回路の出力を反対にしたものが出力になる構成である。真理値表、ＭＩＬ記号、ベン図を示すと、図のようになる。

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

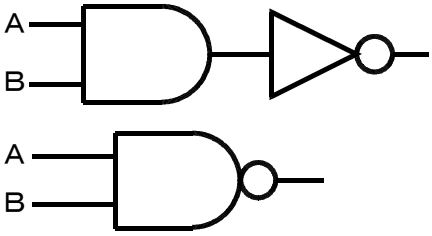


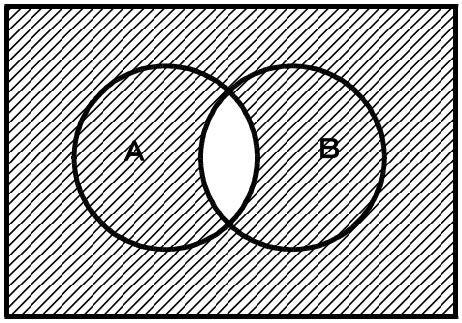


ⓕ　否定論理積

否定論理積は、２つの論理値を入力して論理積を否定した論理値を出力する。この論理演算を回路にしたものを否定論理積回路という。ＡＮＤ回路にＮＯＴ回路を接続したものである。真理値表、ＭＩＬ記号、ベン図を示すと、図のようになる。

A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0





③　正論理と負論理

ⓐ　正論理・負論理とは

㋐　正論理

２進法では真と偽を表す時、真を１とし、偽を０として表すが、この１と０は２つの状態を区別する記号である。自然数の１は０より１つ多いものであり、１を真に、０を偽に割り当てる論理演算では、多い方を真、少ない方を偽と考えるので、この考え方が正論理である。論理回路では電圧や電流が高いか多い場合を１とし、相対的に低い、または少ない場合を０とする。これが正論理である。

㋑　負論理

負論理は、物理的なものの値が相対的に低いか少ないものを真とし、偽は高いか多いもので表したり処理したりする手法である。

ⓑ　ＡＮＤ、ＯＲの論理回路

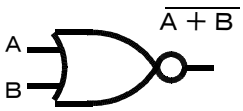
㋐　ＡＮＤ回路

ＡＮＤはすべての入力端子に１が入力されたときのみ１を出力し、それ以外はすべて０となる。逆に、少なくとも１つの入力端子に０が入力されると、出力は０となる。ＡＮＤ回路の入力を反転し、演算回路にＮＯＲ回路を使用すると、同じビットパターンを得る。

A	B	A・B
0	0	0
0	1	0
1	0	0
1	1	1

入力反転

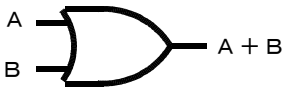
A	B	$\overline{A+B}$
1	1	0
1	0	0
0	1	0
0	0	1



㋑　ＯＲ回路

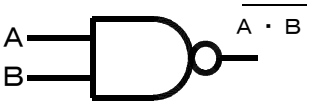
ＯＲ回路は少なくとも１つの入力端子に１が入力されたときに１を出力し、それ以外は０となる。逆に、すべての入力端子に０が入力されたときのみ０を出力する。ＯＲ回路の入力を反転し、演算回路にＮＡＮＤ回路を使用すると、同じビットパターンを得る。

A	B	A＋B
0	0	0
0	1	1
1	0	1
1	1	1



入力反転

A	B	$\overline{A \cdot B}$
1	1	0
1	0	1
0	1	1
0	0	1



ⓒ　ＮＡＮＤ、ＮＯＲの論理回路

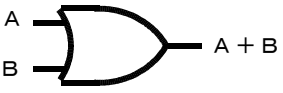
㋐　ＮＡＮＤ回路

ＮＡＮＤはすべての入力端子に１が入力されたときのみ０を出力し、それ以外はすべて１となる。逆に、少なくとも１つの入力端子に０が入力されたときに１を出力する。ＮＡＮＤ回路の入力を反転し、演算回路にＯＲ回路を使用すると、同じビットパターンを得る。

A	B	$\overline{A \cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0

入力反転

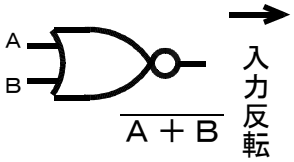
A	B	A＋B
1	1	1
1	0	1
0	1	1
0	0	0



㋑　ＮＯＲ回路

ＮＯＲ回路は少なくとも１つの入力端子に１が入力されたときに０を出し、それ以外は１となる。逆に、すべての 入力端子に０が入力されたときのみ１を出力する。ＮＯＲ回路の入力を反転し、演算回路にＡＮＤ回路を使用すると、同じビットパターンを得る。

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



A	B	A・B
1	1	1
1	0	0
0	1	0
0	0	0

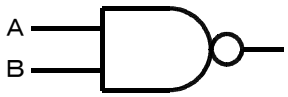
④　論理回路の応用

ⓐ　２値入力ＮＡＮＤ回路

２値入力ＮＡＮＤ回路は、２入力Ａ、Ｂの論理変数のうち、少なくとも０が一つあると、演算の結果は１となり、２変数ともに１のときに０となる。２つの論理変数が１の場合は０となり、そうでないときには１となる論理回路である。

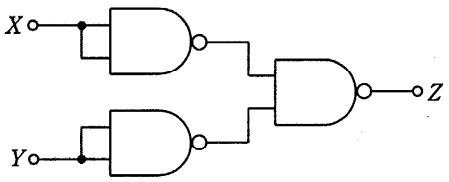
論理回路と真理値表を次に示す。

A	B	$\overline{A\cdot B}$
0	0	1
0	1	1
1	0	1
1	1	0



ⓑ　２値入力で３個のＮＡＮＤ回路を使用した回路

３個のＮＡＮＤ回路を使用して、図に示す論理回路を考えるとき、Ｘ、Ｙに特定のビットを与えたときの演算結果Ｚのビットパターンを求めると、次のようになる。

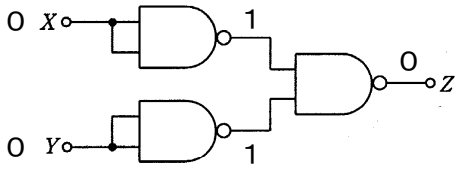


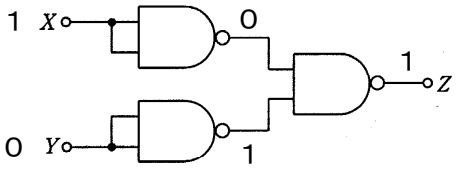
Ｘ＝０、Ｙ＝１とＸ＝１、Ｙ＝０の演算結果は同じになるから、演算結果は次に示す３パタ

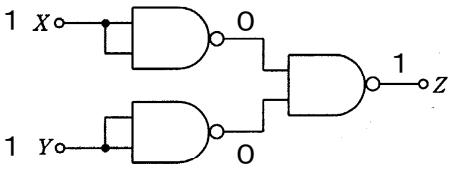
ーンとなる。

Ｘ、Ｙの入力がともに０の場合、結果は０となり、そうでない場合は１となる。

（Ｘ，Ｙ）＝（０，０）＝０、（Ｘ，Ｙ）＝（０，１）＝（１，０）＝（１，１）＝１となる。







ⓒ　４値入力で３個のＮＡＮＤ回路を使用した回路

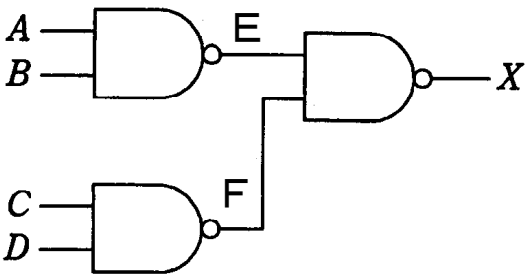
４値入力ＮＡＮＤ回路の論理回路と真理値表を図に示す。

真理値表から言えることは、入力端Ａ、ＢまたはＣ、Ｄのいずれかが共に１となるとき、出力Ｘは１となり、そうでなければ出力Ｘは０となる。

４値入力真理値表

Ａ	Ｂ	Ｃ	Ｄ	Ｅ	Ｆ	Ｘ
０	０	０	０	１	１	０
０	０	０	１	１	１	０
０	０	１	０	１	１	０
０	０	１	１	１	０	１
０	１	０	０	１	１	０
０	１	０	１	１	１	０
０	１	１	０	１	１	０
０	１	１	１	１	０	１
１	０	０	０	１	１	０
１	０	０	１	１	１	０
１	０	１	０	１	１	０
１	０	１	１	１	０	１
１	１	０	０	０	１	１
１	１	０	１	０	１	１
１	１	１	０	０	１	１
１	１	１	１	０	０	１

４値入力ＮＡＮＤ回路



Ｅは入力Ａ、Ｂの否定論理積の出力、Ｆは入力Ｃ、Ｄの否定論理積の出力である。

４値入力ＮＡＮＤの入力の値と演算結果の値の関係は次のようになる。

㋐　演算結果が１になる場合

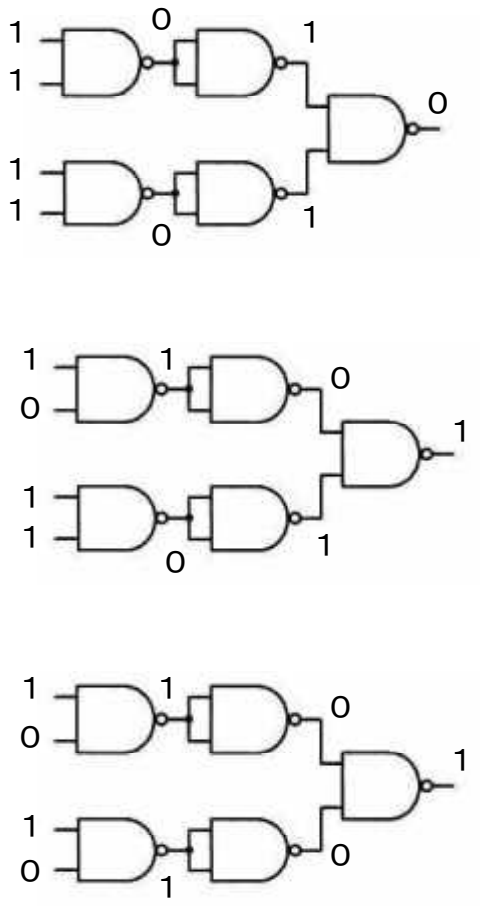
（Ａ、Ｂ、Ｃ、Ｄ）＝（１、１、１、１）＝（１、１、１、０）＝（１、１、０、１）
＝（１、１、０、０）＝（１、０、１、１）＝（０、１、１、１）＝（０、０、１、１）＝１

㋑　演算結果が０になる場合

（Ａ、Ｂ、Ｃ、Ｄ）＝（０，０，０，０）＝（０，０，０，１）＝（０，０，１，０）
＝（０，１，０，０）＝（０，１，０，１）＝（０，１，１，０）＝（１，０，０，０）
＝（１，０，０，１）＝（１，０，１，０）＝０

ⓓ　４値入力のＮＡＮＤ回路

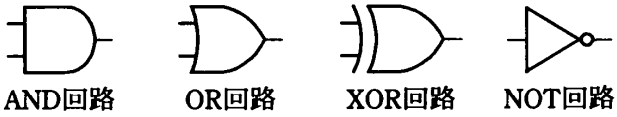
４値入力ＮＡＮＤ回路は、４つの入力のいずれかが０であるときに１を出力し、全ての入力が１のときに０を出力する論理回路である。



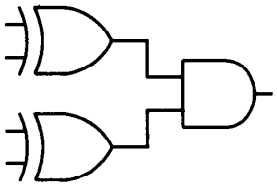
Ａ	Ｂ	Ｃ	Ｄ	Ｘ
０	０	０	０	１
０	０	０	１	１
０	０	１	０	１
０	０	１	１	１
０	１	０	０	１
０	１	０	１	１
０	１	１	０	１
０	１	１	１	１
１	０	０	０	１
１	０	０	１	１
１	０	１	０	１
１	０	１	１	１
１	１	０	０	１
１	１	０	１	１
１	１	１	０	１
１	１	１	１	０

例題演習

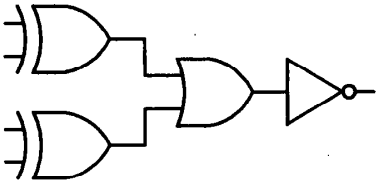
４ビットのデータを入力し，“１の入力数が０個又は偶数個のとき出力が１，奇数個のとき出力が ０”になる回路はどれか。ここで，各回路の図記号は次のとおりとする。



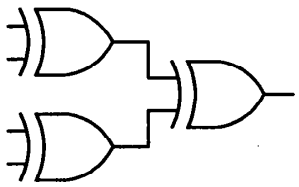
ア



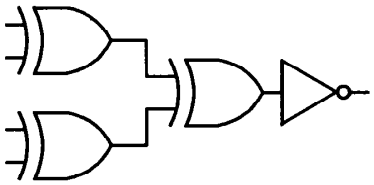
イ



ウ



エ



解答解説

ＭＩＬ記号に関する問題である。

１の数が偶数個、奇数個によって出力が１または０になる問題であるから、１の数が２つの偶数個について検討し、出力が１になる回路について１の個数が奇数の場合について出力が０になることを検討すればよい。

入力のビットパターンとしては、(１１００)、(１０１０)の２通りと、(１０００)について考える。

アの場合、(１１００)→(００)→０となり、１にはならない。

イの場合、(１１００)→(００)→０→１となり、１となる。

(１０１０)→(１１)→１→０となり、１にはならない。

ウの場合、(１１００)→(１０)→１となり、１となる。

(１０１０)→(１１)→０となり、１とならない。

エの場合、(１１００)→(００)→０→１となり、１となる。

(１０１０)→(１１)→０→１となり、１となる。

(００００)→(００)→０→１となり、１となる。

(１１１１)→(００)→０→１となり、１となる。

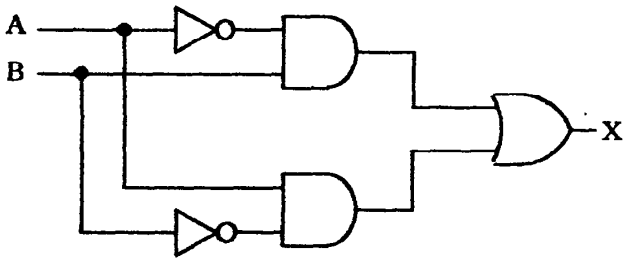
(１０００)→(１０)→１→０となり、０となる。

(１１１０)→(０１)→１→０となり、０となる。

求める答えはエとなる。

例題演習

回路構成を表す論理式として、正しいものはどれか。ア～エの中から選べ。ここで、“・”は論理積（ＡＮＤ）、“＋”は論理和（ＯＲ）、$\overline{A}$はＡの否定（ＮＯＴ）を表す。



ア　$X=\overline{A\cdot B}+\overline{A\cdot B}$　　イ　$X=(\overline{A}\cdot B)+(A\cdot\overline{B})$

ウ　$X=(\overline{A}+B)\cdot(A+\overline{B})$　　エ　$X=\overline{(A+B)}\cdot\overline{(A+B)}$

解答解説

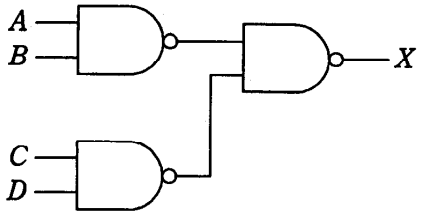
ＭＩＬ記号から論理式を導くことが可能である。

論理積の入力は一方が否定されているため、入力Ａ、Ｂが異なるビットの場合に論理積のどちらかの出力が１となり、二つの論理積の出力の論理和も１になる。入力Ａ、Ｂが同じビットの場合、論理積の出力は共に０となり、論理和の出力は０になる。与えられた論理回路は排他的論理和を示している。

論理回路から、Ａの否定とＢの論理積であるから$\overline{A}\cdot B$、もう一つの論理積は、$A\cdot\overline{B}$となり、この二つの論理積の論理和が求める答えになる。従って、$\overline{A}\cdot B+A\cdot\overline{B}$となる。この論理式は排他的論理和を表している。求める答えはイとなる。

例題演習

図のＮＡＮＤゲートの組合せ回路で，入力Ａ，Ｂ，Ｃ，Ｄに対する出力Ｘの論理式はどれか。ここで，論理式中の“・”は論理積，“＋”は論理和を表す。



ア　（Ａ＋Ｂ）・（Ｃ＋Ｄ）　　イ　Ａ＋Ｂ＋Ｃ＋Ｄ

ウ　Ａ・Ｂ＋Ｃ・Ｄ　　エ　Ａ・Ｂ・Ｃ・Ｄ

解答解説

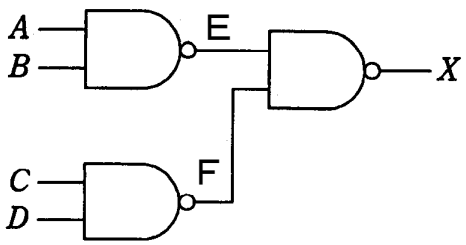
論理回路に関する問題である。

論理回路の入力Ａ、Ｂ、Ｃ、Ｄに０，１のビットを与え、論理回路での演算結果Ｘのビットパターンと解答のア、イ、ウ、エの論理式から求めたビットパターンを比較して一致する論理式が求める答えになる。　真理値表、論理回路を次に示す。

ＥはＡＢのNAND、ＦはＣＤのNANDの結果を示す。Ｘのビットパターンと解答ウのビットパターンが一致する。求める答えはウとなる。

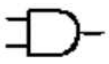
参考：真理値の中の（Ａ、Ｂ）＝（０、１）の４ケースと（Ａ、Ｂ）＝（１、０）の４ケースは論理回路が対象であり、演算結果は一致するため、どちらか一方を検討すればよい。

Ａ	Ｂ	Ｃ	Ｄ	Ｅ	Ｆ	Ｘ	ア	イ	ウ	エ
０	０	０	０	１	１	０	０	０	０	０
０	０	０	１	１	１	０	０	１	０	０
０	０	１	０	１	１	０	０	１	０	０
０	０	１	１	１	０	１	０	１	１	０
０	１	０	０	１	１	０	０	１	０	０
０	１	０	１	１	１	０	０	１	１	０
０	１	１	０	１	１	０	０	１	１	０
０	１	１	１	１	０	１	０	１	１	０
１	０	０	０	１	１	０	０	１	０	０
１	０	０	１	１	１	０	０	１	１	０
１	０	１	０	１	１	０	０	１	１	０
１	０	１	１	１	０	１	０	１	１	０
１	１	０	０	０	１	１	０	１	１	０
１	１	０	１	０	１	１	０	１	１	０
１	１	１	０	０	１	１	０	１	１	０
１	１	１	１	０	０	１	１	０	０	１



例題演習

右の論理回路において，Ｓ＝１，Ｒ＝１，Ｘ＝０，Ｙ＝１のとき，Ｓをいったん０にした後再び１に戻した。この操作を行った後のＸ，Ｙの値はどれか。

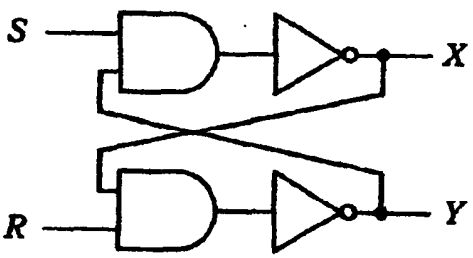
ここで、  論理積  否定 を表す。

ア　Ｘ＝０，Ｙ＝０

イ　Ｘ＝０，Ｙ＝１

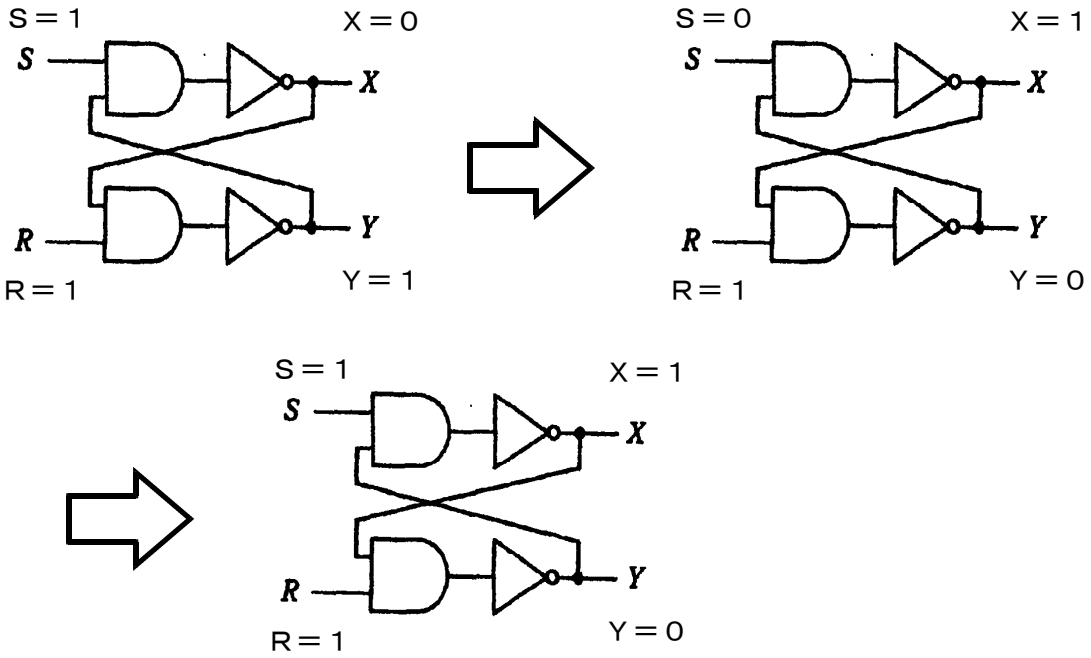
ウ　Ｘ＝１，Ｙ＝０

エ　Ｘ＝１，Ｙ＝１



解答解説

フリップフロップ回路の問題である。

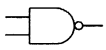


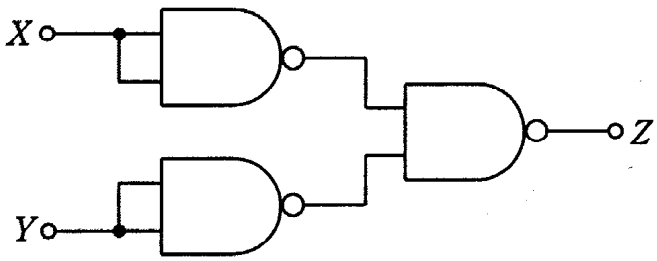
S	R	X	Y
1	1	0	1
0	1	1	0
1	1	1	0

S、Rの値を変化させてシミュレーションを実行すれば答えを得ることができる。

S、R、X、Yは表のように変化する。最後の状態はX＝1、Y＝0となる。求める答えはウとなる。

例題演習

ＮＡＮＤ回路による次の組合せ回路の出力Ｚを表す式はどれか。ア～エの中から選べ、ここで，  はＮＡＮＤ回路，・は論理積、＋は論理和，$\overline{X}$はＸの否定を表す。



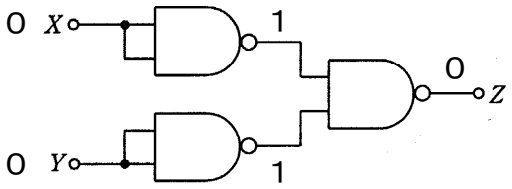
ア　X・Y　　イ　X＋Y　　ウ　$\overline{X+Y}$　　エ　$\overline{X \cdot Y}$

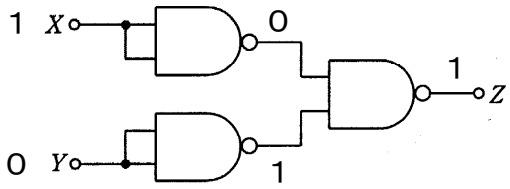
解答解説

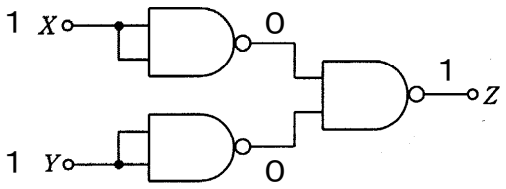
ＮＡＮＤ回路に関する問題である。

与えられた論理回路の演算内容、解答群の真理値表を作成すると図及び表のようになる。

論理回路の入力（Ｘ、Ｙ）＝（１、０）＝（０、１）は演算結果が同一となるため、入力（１、０）についてのみ考える。







X	Y	Z	X・Y	X＋Y	$\overline{X+Y}$	$\overline{X \cdot Y}$
0	0	0	0	0	1	1
1	0	1	0	1	0	1
0	1	1	0	1	0	1
1	1	1	1	1	0	0

Ｚのビットパターンと一致するのは、Ｘ＋Ｙであり、求める答えはイとなる。

txt010143 論理演算応用

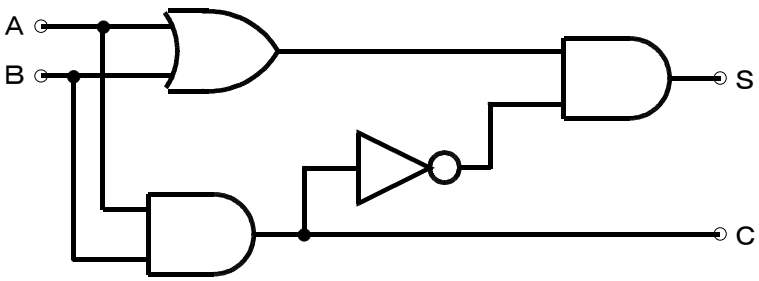
① 半加算器・全加算器

ⓐ 半加算器

半加算器は、２個の２進数の和を求めるが、桁上がりを考慮に入れない加算器で、１ビットの２進数の加算しかできない。排他的論理和と論理積の組み合わせで実現している。

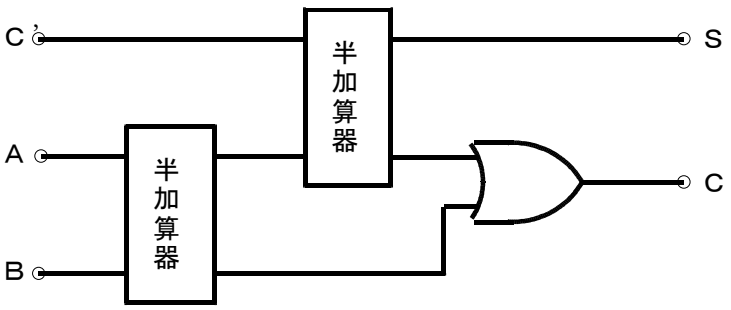
論理回路と真理値表を示す。Ｓは１桁の加算で、排他的論理和の結果であり、Ｃは桁上がりで、論理積の結果である。

入力		出力	
A	B	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1



ⓑ 全加算器

入力			出力	
A	B	C	S	C
0	0	0	0	0
0	1	0	1	0
1	0	0	1	0
1	1	0	0	1
0	0	1	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	1	1



全加算器は、２桁以上の２進数の加算を行う場合において、最下位の位以外の上位の各桁に、下位からの桁上げによる入力を考慮した加算回路である。下位からの桁上がりとその桁の２つの入力の３つの入力で、桁内の加算と桁上がりの２つの出力を得る回路となる。半加算器とＯＲ回路の組合せからできている。論理回路と真理値表を示す。

入力値の組合せ８通りに対して、４通りの出力値を得る。出力Ｓの値は３入力のうち１のビ

ットが奇数個の場合は１、偶数個の場合は０となり、出力Ｃの値は３入力のうち１のビットが２個以上あれば１、２個未満では０となる。

②　オートマトンと状態遷移図

ⓐ　オートマトン

オートマトンは状態遷移を持つ順序理論で構成された仮想的な機械で、入出力装置と状態制御装置で構成されている。初期状態、受理状態、入力、状態の変化などの複数の状態で構成されている。データを得る・計算する・結果を出力するなど、コンピュータを使用して問題解決する場合の処理手順をモデル化したものである。形式言語理論、文字列検索、コンパイラの構文解析などの分野に応用される。

ⓑ　状態遷移図

状態遷移図はシステムの取り得る状態の種別とその状態が遷移するための要因との関係をわかりやすく表現する記述形式である。基本的には円と矢印で表記する。

ⓒ　状態遷移図の構成要素

㋐　状態

ある時点のプロセスの状態を円で表す。

㋑　状態遷移

ある状態から他の状態への遷移を矢印で表す。

㋒　遷移条件と処理

ある状態から他の状態へ遷移する条件と、そのときに発生する処理を矢印の上に明記する。

ⓓ　タスク管理の状態遷移図の例

㋐　実行可能状態のタスクのうち最も優先順位の高いタスクを選択する。

選択されたタスクにＣＰＵの制御権が与えられ、実行状態に移される。ディスパッチャは、ＴＣＢの待ち行列を調べ、優先順位の高いタスクを実行可能状態タスクの中から選択するプログラムであり、この処理機能を実行することをディスパッチングという。

㋑　実行状態のタスクで入出力オペレーションが必要になると、そのタスクは要求した入出力動作が終了するまで待ち状態となる。

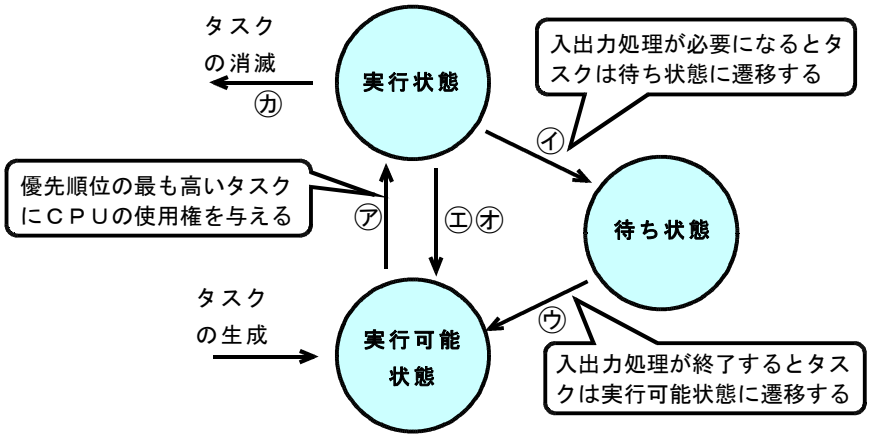
㋒　入出力オペレーションが終了すると、タスクは実行可能状態となる。

入出力オペレーションが終了すると、終了を知らせる信号が入出力装置からＣＰＵに送られる。この信号を受け取ってＣＰＵは実行中のタスクを中断し、入出力待ちのタスクを実行可能状態にする。この機能は入出力割込によって行われる。

㋓　実行中のタスクよりも優先順位の高いタスクが実行可能状態になった場合、実行状態のタスクは実行を中断し、実行可能状態に移り、その後、優先順位の高いタスクにＣＰＵの制御権を渡す。

㋔　タイムシェアリングシステムでは、接続されている端末に数十ミリ秒単位毎にＣＰＵ時間を配分し、この時間を経過する毎に実行中のタスクを実行可能状態に戻す。このＴＳＳの機能はタイマ割込機能と呼ばれる。

㋕　タスクの実行が完了すると、ＴＣＢはＴＣＢ待ち行列から削除される。この時、該当タスクが使用していたシステム資源は全て解放される。



ⓔ　自動販売機の状態遷移図の例

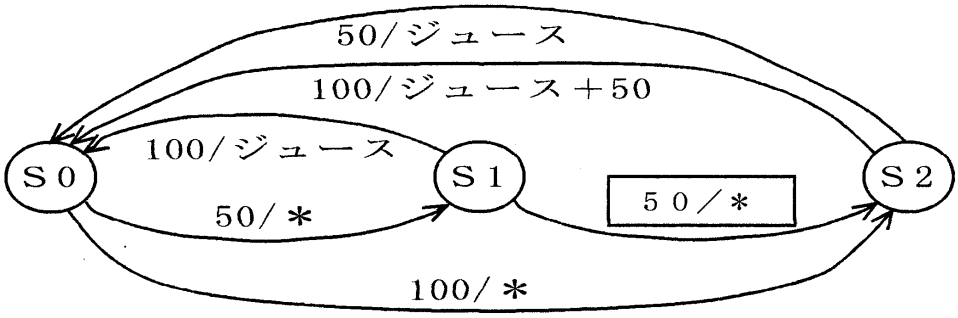
図は初期状態をＳ０とする自動販売機の状態遷移を示したものである。

㋐　初期状態Ｓ０で100円を自動販売機に投入すると、タスクはＳ０からＳ２に遷移し待機する。更に50円を投入するとジュースを提供してタスクはＳ０に戻る。

㋑　初期状態Ｓ０で50円を自動販売機に投入すると、タスクはＳ０からＳ1に遷移し待機する。更に100円を投入するとジュースを提供してタスクはＳ０に戻る。

㋒　初期状態Ｓ０で50円を２回自動販売機に投入すると、タスクはＳ０からＳ１、更にＳ２に遷移し待機する。更に50円を投入するとジュースを提供してタスクはＳ０に戻る。

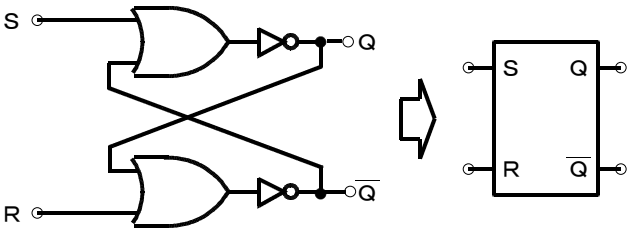
㊓　初期状態Ｓ０で100円を自動販売機に投入すると、タスクはＳ０からＳ２に遷移し待機する。更に100円を投入すると釣銭50円とジュースを提供してタスクはＳ０に戻る。



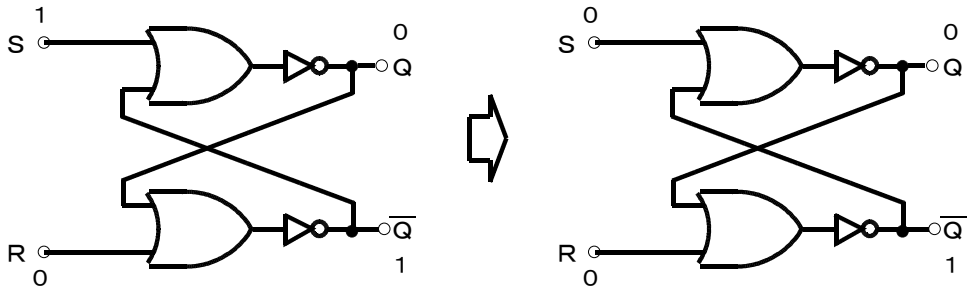
③　その他の応用回路

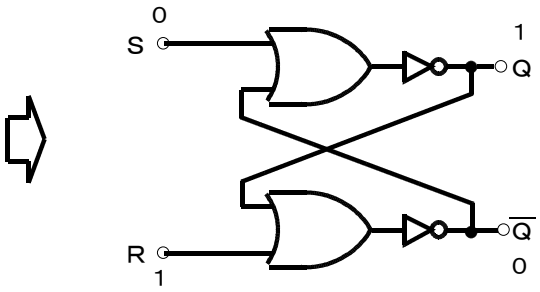
ⓐ　フリップフロップ回路

入力		出力	
S	R	Q	$\overline{Q}$
1	0	0	1
0	0	0	1
0	1	1	0



Ｓ＝１、Ｒ＝０からＳ＝０、Ｒ＝０となっても変化しない。更に、Ｓ＝０、Ｒ＝１になると出力は反転する



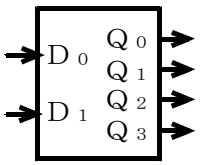


フリップフロップ回路は、０と１の二つの安定状態をもつ回路で、順序回路の基本構成要素である。２つの状態をとり、どちらの状態にも安定する回路であり、ある入力信号によって一方の状態に安定すると、次に他方の状態に変える入力信号がくるまで、その状態を保つ。１ビットのデータを記憶できる。ＳＲＡＭの記憶セルに使用する。

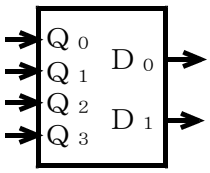
ⓑ　デコーダとエンコーダ

左の図はデコーダで、符号化された信号を解読する組み合わせ回路である。ＣＰＵ命令の解読、周辺回路の選択信号の作成などに使用する。デコーダの出力は相互に排他的であり、入力パターンに対応する出力だけ１とし、残りはすべて０となる。

右の図はエンコーダで、デコーダの反対の機能をもち、符号化機能を行う。複数の入力のうち１つだけが１となり、１になった入力に対応する出力パターンを生成する。



D_0	D_1	Q_0	Q_1	Q_2	Q_3
0	0	1	0	0	0
1	0	0	1	0	0
0	1	0	0	1	0
1	1	0	0	0	1



Q_0	Q_1	Q_2	Q_3	D_0	D_1
1	0	0	0	0	0
0	1	0	0	1	0
0	0	1	0	0	1
0	0	0	1	1	1

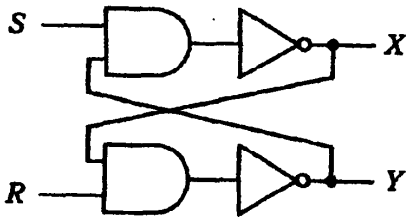
例題演習

論理回路において，Ｓ＝１，Ｒ＝１，Ｘ＝１，Ｙ＝０のとき，Ｒをいったん０にした後再び１に戻した。この操作を行った後のＸ，Ｙの値はどれか。ア～エの中から選べ。　ここで、を表す。


論理積


否定

ア　Ｘ＝０，Ｙ＝０
イ　Ｘ＝０，Ｙ＝１
ウ　Ｘ＝１，Ｙ＝０
エ　Ｘ＝１，Ｙ＝１



解答解説

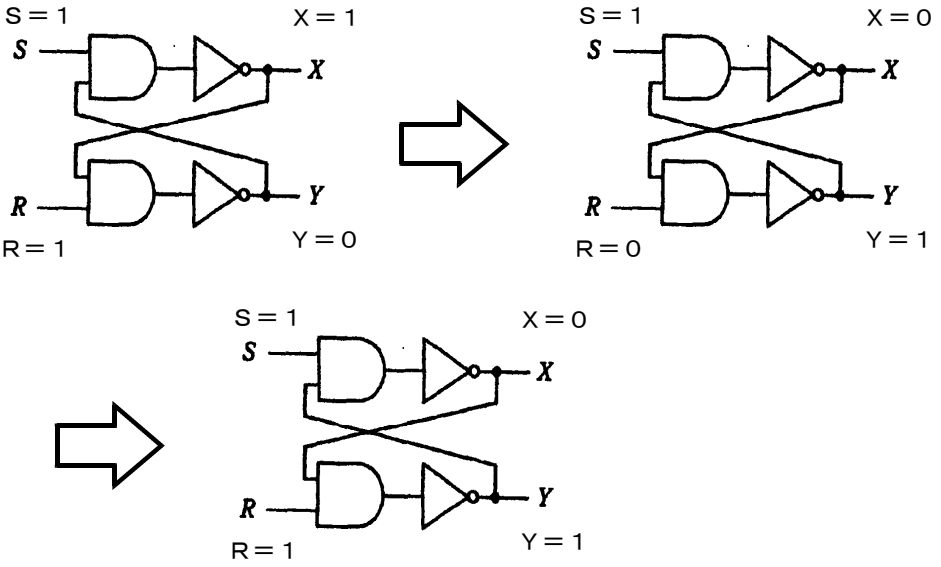
フリップフロップ回路の問題である。

S	R	X	Y
1	1	1	0
1	0	0	1
1	1	0	1

Ｓ、Ｒの値を変化させてシミュレーションを実行すれば答えを得ることができる。

Ｓ、Ｒ、Ｘ、Ｙは表のように変化する。最後の状態はＸ＝０、Ｙ＝１となる。求める答えは

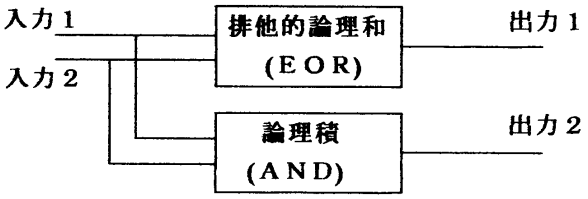
イとなる。



例題演習

図に示す構造の論理回路は、どれか。

ア　減算
イ　乗算
ウ　全加算
エ　半加算



解答解説

半加算器に関する問題である。

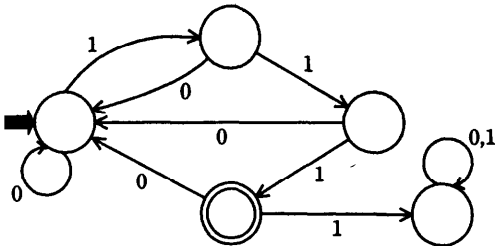
真理値表のビットパターンを検討し、論理回路との特徴を明確にする。

桁内の演算結果の数が排他的論理和で求まり、桁上げ数が論理積の演算で求まるから、半加算器である。求める答えはエとなる。

例題演習

図で表される有限オートマトンで受理される文字列はどれか。ア～エの中から選べ。ここで、→○は初期状態を，◎は受理状態を表す。

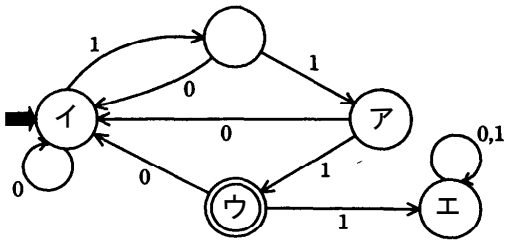
ア　01011
イ　01111
ウ　10111
エ　11110



解答解説

状態遷移図に関する問題である。

ア～エの解答群に従ってシミュレーションを実行すると、それぞれの終了状態は図に示すようになる。受理状態で終了するのはウの場合である。求める答えはウとなる。



例題演習

状態遷移表をもつシステムの状態がｓ１であるとき、入力信号（ｔ１、ｔ２、ｔ３、ｔ４、ｔ１、ｔ２、ｔ３、ｔ４）を順次入力したとき、最後の状態はどれか。ここで、空欄は状態が変化しないことを表す。

信号＼状態	ｓ１	ｓ２	ｓ３	ｓ４
ｔ１		ｓ３		
ｔ２	ｓ３		ｓ２	
ｔ３			ｓ４	ｓ１
ｔ４		ｓ１		ｓ２

ア　ｓ１　　イ　ｓ２　　ウ　ｓ３　　エ　ｓ４

解答解説

状態遷移表の応用に関する問題である。

入力信号ｔ１，ｔ２，ｔ３，ｔ４，ｔ１，ｔ２，ｔ３，ｔ４に対応する状態の遷移を求めると次のようになる。

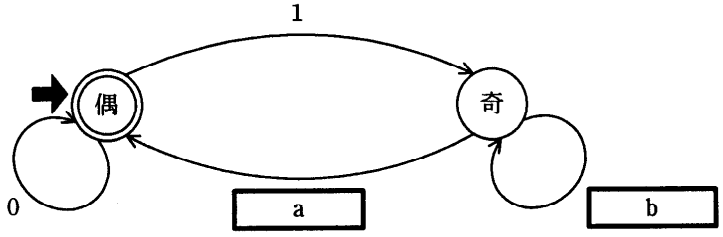
ｓ１→ｓ１→ｓ３→ｓ４→ｓ２→ｓ３→ｓ２→ｓ２→ｓ１

最後の状態はｓ１となる。求める答えはアとなる。

例題演習

図は１の数が偶数個のビット列を受理するオートマトンの状態遷移図であり，“偶”と書かれた二重丸が受理状態を表す。ａ，ｂの正しい組合せはどれか。

	a	b
ア	0	0
イ	0	1
ウ	1	0
エ	1	1



解答解説

オートマトンに関する問題である。

アの場合、偶→奇となり、受理しない

イの場合、偶→奇→奇となり、受理しない

ウの場合、偶→奇→偶となり、受理する。求める答えはウとなる。

エの場合、偶→奇→奇、偶→奇→偶で不安定となり、必ずしも受理しない。

例題演習

Ｎ個の観測値の和Ｓ（ただし，Ｓ＞０）を求め平均値を算出する。平均値は，小数部を四捨五入して整数値で求めるとしたとき，正しい式はどれか。ア～エの中から選べ。ここで，／は除算，［Ｘ］はＸ以下で最大の整数とする。

ア　［（Ｓ＋０．５）／Ｎ］　　イ　［（Ｓ－１）／Ｎ］＋１

ウ　［Ｓ／Ｎ＋０．５］　　エ　［Ｓ／Ｎ］＋１

解答解説

平均値の四捨五入に関する問題である。

小数１桁の四捨五入は、小数１桁の値が0.5～0.9の場合に１が加算され、０～0.4の場合は切り捨てられる。従って、小数１桁に0.5加算して、小数以下を切り捨てればよい。

平均値の四捨五入の問題であるから、平均値の計算結果について実行すればよい。

アの場合、観測値に対して＋0.5しているため、平均値が0.5大きくなってしまう。

イの場合、観測値から１減じて平均値を求め、１だけ小さい平均値を求めて小数以下を切り捨て、その結果に１を加算しても四捨五入にはならない。例えば、平均値が15.5の場合、平均値14.5を求め、切り捨てで14となり、１加算して15となる。正しい平均値は16である。

ウの場合、観測値から平均値を求め、その結果に0.5を加算して小数以下を切り捨てている。四捨五入が可能である。求める答えはウとなる。

エの場合、平均値を求め、小数以下を切り捨て、それに１を加算している。平均値が15.4の場合、切り捨てて15となり、これに１加算すると16となる。正しい平均値は15である。

例題演習

整数Ａを整数Ｂで割って余りを得るための関数mod（Ａ、Ｂ）が次のように定義されているとき、関数呼出によって得られる値として正しいものはどれか。

［定義］

ｍｏｄ（Ａ、Ｂ）は、除数Ｂと同じ符号をとり、その絶対値はＢの絶対値より小さい、適切な整数Ｎを選ぶことによって、Ａ＝Ｂ×Ｎ＋ｍｏｄ（Ａ、Ｂ）を満足する。

ア　mod（11，５）＝２　　イ　mod（11，－５）＝－１

ウ　mod（12，－５）＝－３　　エ　mod（－12，５）＝２

解答解説

余りを求める関数mod(A, B)に関する問題である。

次の点に注意して処理を検討する必要がある。

① 余りは除数Bと同じ符号である。

② 余りの絶対値は除数Bの絶対値より小さい。

③ A、B、余りの間には次の関係が成立する。

A＝B×N＋mod(A, B) 但し、Nは整数である

この条件を利用して、解答群のア～エについて検討する。

アは11÷５の余りを求める問題で、mod(11, ５)＝１であるから正しくない。

イは11÷(－５)の余りを求める問題で、①、②の条件から商は３、余りは－４となる。正しくない。

ウは12÷(－５)の余りで、商は３、余りは－３となる。正しい。求める答えはウとなる。

エは(－12)÷５の余りを求める問題である。①、②の条件から商は３で、余りは３となる。A＝５×(－３)＋３＝－12。正しくない。

例題演習

業務の改善提案に対する賞金が，次の決定表で決められる。改善提案１と改善提案２に対する賞金の総額は何円か。ア～エの中から選べ。

改善額10万円未満	Y	Y	N	N
期間短縮１週間未満	Y	N	Y	N
賞金： 500円	X	－	－	－
賞金：1,000円	－	X	X	－
賞金：3,000円	－	－	－	X

〔改善提案〕

改善提案１：改善額20万円、期間短縮３日

改善提案２：改善額５万円、期間短縮２週間

ア 1,000　　イ 1,500　　ウ 2,000　　エ 3,500

解答解説

改善提案１は、改善額20万円、期間短縮３日であるから、決定表から、改善額10万円未満がＮ、期間短縮１週間未満Ｙの場合で賞金1000円、改善提案２は、改善額５万円、期間短縮２週間であるから、決定表から、改善額10万円未満がＹ、期間短縮１週間未満Ｎの場合で賞金1000円となり、両者合わせて2000円となる。求める答えはウとなる。