ディジタル量の扱い方

電気電子計測で扱う量の多くはアナログ量であるにもかかわらず、ディジタル技術を導入した計測器が多く使われるようになってきた。信号の劣化がない、データの保存が容易である、コンピュータに容易に取り込んでデータ処理やグラフ化が簡単にできる等がその理由であり、計測器を使いこなすためにはディジタル技術の基礎知識は必須である。そこで、本章では計測に必要なディジタル量の扱い方について学ぶことにする。

4.1 ２進法と１０進法

(a) バイトとビット

10 進法では数値を以下のように表わす。

$$D_n D_{n-1} D_{n-2} \cdots D_0 . D_{-1} D_{-2} \qquad (4.1)$$

D_n～D_{-2} は 0-9 までの整数を表わす。これは 10 のべき数を使って表示すると

$$D_n x10^n + D_{n-1} x10^{n-1} + D_{n-2} x10^{n-2} + \cdots D_0 x10^0 + D_{-1} x10^{-1} + D_{-2} x10^{-2} \qquad (4.2)$$

のようになる。10 進法の場合、10 を基数としている。それに対して２進法で

$$B_n B_{n-1} B_{n-2} \cdots B_0 . B_{-1} B_{-2} \qquad (4.3)$$

は、２のべき数を使って表わすと

$$B_n x2^n + B_{n-1} x2^{n-1} + B_{n-2} x2^{n-2} \cdots B_0 x2^0 + B_{-1} x2^{-1} + B_{-2} x2^{-2} \qquad (4.4)$$

のように表わされる。B_n~B_{-2} は 0 もしくは 1 のいずれかである。ディジタル信号では B_n~B_{-2} の一つを**ビット**(bit)と呼ぶ。また、８ビットを一つにまとめたものが**バイト**(byte)である。

例

２進数の 1111 は

$1x2^3+1x2^2+1x2^1+1x2^0$

となり、１０進数では１５になる。

２進法でデータを表現すると長くなるので、４ビットをまとめて一つの桁としたのが**１６進法**である。１０～１５の数値は A~F で表現する。

$$H_n H_{n-1} H_{n-2} \cdots H_0 . H_{-1} H_{-2} \qquad (4.5)$$

は 16 のべき数で表わすと

$$H_n x16^n + H_{n-1} x16^{n-1} + H_{n-2} x16^{n-2} \cdots H_0 x16^0 + H_{-1} x16^{-1} + H_{-2} x16^{-2} \qquad (4.6)$$

となる。この 16 進数もディジタル信号の表現方法としてよく用いられる。

例

16 進で F1 は１０進数で表わすと

$15x16^1+1x16^0=241$

のように計算される。

例

$(100)_{10\text{進}}$ を 16 進数及び 2 進数で表す。

$100=6x16^1+4x16^0=(64)_{16進}=(0110\ 0100)_{2進}$

（b）負数の表現方法

次に負のデータの表現は２進数でどうなるのだろうか？今mビットのデータを考える。そのとき、最上位の桁（MSB, Most Significant Bit）が０の時正、１の時は負であると考え、MSB の桁を符号ビットと呼ぶ。X（X＞０）に対してーX は

$$-X=[(2^m-1)-X]+1 \qquad (4.7)$$

のように表現する。ここで、2^m-1 はすべてのビットが１であり、それから X をひくということは、すべてのビットの 0 と１を反転させることを意味する。この[(2^m-1)-X]を X の**補数**という。―X は X の補数に１を足したものである。

このように負の数を２進数で表わすと、mビットで表わせるデータの範囲は

$$-(2^{m-1}-1)\le X\le 2^{m-1}-1$$

のようになる。

例

－５、－１００を８ビットの１６進数及び２進数で表すことを考える。

－５の場合：$[2^8$-1-5]+1=（251）$_{10進}=15x16^1+11x16^0=(FB)_{16進}$

$=(1111\ 1011)_{2進}$

－１００の場合：$[2^8$-1-100]+1=$(156)_{10進}=9x16^1+12x16^0=(9C)_{16進}$

$=(1001\ 1100)_{2進}$

例

８ビットデータ$(F0)_{16進}$は１０進数のいくらに対応するかを考える。

F0=$(11110000)_{2進}$をビット反転させると、$(00001111)_{2進}$

この値に１を加えると

$(00001111)_{2進}+(00000001)_{2進}=(00010000)_{2進}=$（16）$_{10進}$

すなわち, F0 は-16 に対応する。

通常は、このように符号を含めて考えるがデータ処理の過程で符号を考えないほうが都合がよい場合もある。その場合は**符号なし整数**として取り扱うようにする必要がある。たとえば、８ビットデータにおいて FF は-1 であるが、符号なしの整数とすれば、１０進数で 255 である。

例 8 ビットデータ$(80)_{16進}$を符号なし１０進数及び符号付き１０進数で表すことを考える。

符号無し１０進数の場合：$8x16^1+0x16^0=(128)_{10進}$

符号有り１０進数の場合：$(80)_{16進}$を反転させて１を足すと

$(1000\ 0000)_{2進}$の反転→$(0111\ 1111)_{2進}$

$(0111\ 1111)_{2進}+1=(1000\ 0000)_{2進}=１２８_{10進}$

つまり、－１２８となる。