10.6 Description of the EInt formats

A parameter with Eint format can be represented in two different formats (F). Either as a 15 bit unsigned integer format (F= 0) or a Emotron floating point format (F=1). The most significant bit (B15) indicates the format used. See detailed description below.
All parameters written to a register may be rounded to the number of significant digits used in the internal system.

The matrix below describes the contents of the 16-bit word for the two different EInt formats:

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
F=1	e3	e2	e1	e0	m10	m9	m8	m7	m6	m5	m4	m3	m2	m1	m0
F=0	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0

If the format bit (B15) is 0, then all bits may be treated as a standard unsigned integer (UInt)

If the format bit is 1, then is the number interpreted as this:

Value = M * 10^E, where M=m10..m0 represents a two- complement signed mantissa and E= e3..e0 represents a two- complement signed exponent.

NOTE: Parameters with EInt format may return values both as 15 bit unsigned int (F=0) or in Emotron floating point (F=1).

Example, resolution

If you write the value 1004 to a register and this register has 3 significant digits, it will be stored as 1000.

In the Emotron floating point format (F=1), one 16-bit word is used to represent large (or very small numbers) with 3 significant digits.

If data is read or written as a fixed point (i.e. no decimals) number between 0-32767, the 15 bit Unsigned integer format (F=0) may be used.

Detailed description of Emotron floating point format

```
e3-e0 4-bit signed exponent. Gives a value
range:
-8..+7 (binary 1000 .. 0111)
m10-m0 11-bit signed mantissa.Gives a value
range:
-1024..+1023 (binary
10000000000..01111111111)
```

A signed number should be represented as a two complement binary number, like below:

Value Binary

```
-8 1000
-7 1001
..
-2 1110
-1 1111
0 0000
1 0001
2 0010
..
6 0110
7 0111
```

The value represented by the Emotron floating point format is m·10e.

To convert a value from the Emotron floating point format to a floating point value, use the formula above.

To convert a floating point value to the Emotron floating point format, see the C-code example below.

Example, floating point format

The number 1.23 would be represented by this in Emotron floating point format,

```
F EEEE MMMMMMMMMMM
1 1110 00001111011
F=1 -> floating point format used
E=-2
M=123
```

The value is then $123x10^{-2} = 1.23$

Example 15bit unsigned int format

The value 72.0 can be represented as the fixed point number 72. It is within the range 0-32767, which means that the 15-bit fixed point format may be used.

The value will then be represented as:

B15	B14	B13	B12	B11	B10	B9	B8	B7	B6	B5	B4	B3	B2	B1	B0
0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0

Where bit 15 indicates that we are using the fixed point format (F=0).

Programming example:

```
typedef struct
{
  int m:11; // mantissa, -1024..1023
  int e: 4; // exponent -8..7
  unsigned int f: 1; // format, 1->special emoint format
}    eint16;
//---------------------------------------------------------------------------
unsigned short int float_to_eint16(float value)
{
  eint16 etmp;
  int dec=0;

  while (floor(value) != value && dec<16)
  {
     dec++; value*=10;
  }
  if (value>=0 && value<=32767 && dec==0)
     *(short int *)&etmp=(short int)value;
  else if (value>=-1000 && value<0 && dec==0)
  {
     etmp.e=0;
     etmp.f=1;
     etmp.m=(short int)value;
  }
  else
  {
     etmp.m=0;
     etmp.f=1;
     etmp.e=-dec;
     if (value>=0)
        etmp.m=1; // Set sign
     else
        etmp.m=-1; // Set sign
     value=fabs(value);
     while (value>1000)
     {
        etmp.e++; // increase exponent
        value=value/10;
     }
     value+=0.5; // round
     etmp.m=etmp.m*value; // make signed
  }
 return (*(unsigned short int *)&etmp);
}
//---------------------------------------------------------------------------
float eint16_to_float(unsigned short int value)
{
  float f;
  eint16 evalue;

  evalue=*(eint16 *)&value;
  if (evalue.f)
  {
     if (evalue.e>=0)
        f=(int)evalue.m*pow10(evalue.e);
     else
        f=(int)evalue.m/pow10(abs(evalue.e));
  }
  else
     f=value;

  return f;
}
//---------------------------------------------------------------------------
```