

---

Subject: Re: converting ieee-float-format to pw-wave-float-format

Posted by [caron](#) on Thu, 25 Aug 1994 15:52:02 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

you need to examine the binary representation of floating points on the two machines you are using. If you say what the cpus are, perhaps someone here could tell you the representations.

I remember converting VAX 11/70 to Intel x86; only difference was byte order and the exponent bias. if exponent bias differed by 2, that would give you a factor of 4.

The nice thing about ieee is that its the same across all machines (is that true for its byte order also?).

---

---

Subject: Re: converting ieee-float-format to pw-wave-float-format

Posted by [grunes](#) on Thu, 25 Aug 1994 19:39:16 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

In article <33ien2\$17v@ncar.ucar.edu> caron@acd.ucar.edu (John Caron) writes:

> The nice thing about ieee is that its the same across all machines (is that  
> true for its byte order also?).

No. IEEE does not define byte order--at least it is in oposite order on PCs from the most common useage on most Unix Workstations (Suns, SGIs,... -- but watch out for Dec Alpha, which someone told me use backwards (least significant byte first, like the PC) byte order.

The easiest translation method is to read the data on someone's Sun, print it out formatted, and send it as text!

One thing puzzles me: Most floating point numbers translate between Suns and PCs simply throught BYTEORDER. But NAN values seem to behave differently. Does anyone know why?

Mitchell R Grunes (grunes@imsy1.nrl.navy.mil)

Allied-Signal Technical Services

c/o Code 7230 Naval Research Lab

---

---

Subject: converting ieee-float-format to pw-wave-float-format

Posted by [SLAMECZKA](#) on Fri, 26 Aug 1994 08:07:33 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

In <33ien2\$17v@ncar.ucar.edu> caron@acd.ucar.edu writes:

> you need to examine the binary representation of floating points on the

> two machines you are using. If you say what the cpus are, perhaps someone  
> here could tell you the representations.  
>  
> [snip]  
>  
> The nice thing about ieee is that its the same across all machines (is that  
> true for its byte order also?).

The manual of the program (EDAS) I am using on my PC 386 says, that the  
datas will be written binary to the harddrive. I copy this datas to my network  
harddrive, start pw\_wave 4.01 (Dec VAX), read the datas and all the values are  
correct, exept the floating points.

The manual only says; that the float is a 4 Byte floating point (float in  
"c", IEEE Format). Thats everything.

Now what is this IEEE format?

Which bits are the exponent, which are the mantisse and how to translate it  
to pw\_wave??

I hope for help  
thanks  
Michael

---

Subject: Re: converting ieee-float-format to pw-wave-float-format  
Posted by [grunes](#) on Fri, 26 Aug 1994 12:47:11 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

In article <33k7s5\$972@elna.ethz.ch> SLAMECZKA@EZINFO.VMSMAIL.ETHZ.CH  
(SLAMECZKA,MICHAEL) writes:

> Now what is this IEEE format?  
> Which bits are the exponent, which are the mantisse and how to translate it  
> to pw\_wave??

Be clear on one thing: there is no IDL or PV-Wave floating point format,  
when used with readu and writeu.

They use what ever is normal for their machines. i.e.--if you read it  
back on the same computer that you wrote it on, everything would  
be fine.

I don't recall the exact format, but on a MSB first machine I  
believe the order is:

- Sign Bit (NOT twos complement format)
- Base 2 exponent, excess notation
- Mantissa, with assumed (not present) leading 1 (except for true zero,  
which is represented by all zero bits)

On your PC (a LSB first machine) the bytes (NOT the fields) are in reversed order. Also: IEEE includes a variety of NaNs to represent overflows, underflows and undefined values.

If you experiment, you should be able determine how many bits of exponent and mantissa work. I seem to remember about 11 exponent bits, which would give about 32-1-11 mantissa bits (+ the assumed leading 1 bit).

Mitchell R Grunes (grunes@imsy1.nrl.navy.mil)

Allied-Signal Technical Services

c/o Code 7230 Naval Research Lab

---

Subject: Re: converting ieee-float-format to pw-wave-float-format  
Posted by [grunes](#) on Fri, 26 Aug 1994 13:51:38 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

I was wrong: 8 bit exponent.

If you understand Fortran, these routines, which haven't been completely tested, might help:

```
C-----
      function FromVaxR4(x)
c Function to convert Vax real*4 number to local floating point.
c Cannot handle NaNs or numbers which are too small or too large.
c By mitchell r grunes.
      integer*4  x,y,i           ! Really Vax real*4--but
                                ! must be kept in integers
                                ! so won't be "normalized".

      character*1 a(4)
      equivalence (y,a)
      parameter (Mask23=2**23-1)
      parameter (ioffset=128+24)

      y=x
      i=      iand(ichar(a(2)),255)
      i=ior(ishft(i,8),iand(ichar(a(1)),255))
      i=ior(ishft(i,8),iand(ichar(a(4)),255))
      i=ior(ishft(i,8),iand(ichar(a(3)),255))

      iexponent=iand(ishft(i,-23),255)-ioffset
      mantissa=iand(i,Mask23)

      if(i.eq.0)then
        FromVaxR4=0
      else
```

```

mantissa=ior(ishft(1,23),mantissa)
if(i.gt.0)then
  FromVaxR4= mantissa*2.**iexponent
else
  FromVaxR4=-mantissa*2.**iexponent
endif
endif
end

```

C-----

```

function FromIEEEER4(x)

```

c Function to convert IEEE real\*4 number to local floating point.  
c Assumes number written on a "most significant byte first" machine like  
c a Sun or SGI workstation.  
c Cannot handle NaNs or numbers which are too small or too large.  
c By mitchell r grunes.

```

integer*4 x,y,i      ! Really IEEE real*4--but
                    ! must be kept in integers
                    ! so won't be "normalized".

```

```

character*1 a(4)
equivalence (y,a)
parameter (Mask23=2**23-1)
parameter (ioffset=128+22)

```

```

y=x
i=      iand(ichar(a(1)),255)
i=ior(ishft(i,8),iand(ichar(a(2)),255))
i=ior(ishft(i,8),iand(ichar(a(3)),255))
i=ior(ishft(i,8),iand(ichar(a(4)),255))

```

```

iexponent=iand(ishft(i,-23),255)-ioffset
mantissa=iand(i,Mask23)

```

```

if(i.eq.0)then
  FromIEEEER4=0
else
  mantissa=ior(ishft(1,23),mantissa)
  if(i.gt.0)then
    FromIEEEER4= mantissa*2.**iexponent
  else
    FromIEEEER4=-mantissa*2.**iexponent
  endif
endif
end

```

C-----

```

function FromRIEEER4(x)

```

c Function to convert IEEE real\*4 number to local floating point.  
c Assumes number written on a "least significant byte first" machine like  
c a PC.

c Cannot handle NaNs or numbers which are too small or too large.

c By mitchell r grunes.

```
integer*4 x,y,i      ! Really IEEE real*4--but
                     ! must be kept in integers
                     ! so won't be "normalized".
```

```
character*1 a(4)
equivalence (y,a)
parameter (Mask23=2**23-1)
parameter (ioffset=128+22)
```

```
y=x
i=      iand(ichar(a(4)),255)
i=ior(ishft(i,8),iand(ichar(a(3)),255))
i=ior(ishft(i,8),iand(ichar(a(2)),255))
i=ior(ishft(i,8),iand(ichar(a(1)),255))
```

```
ie exponent=iand(ishft(i,-23),255)-ioffset
mantissa=iand(i,Mask23)
```

```
if(i.eq.0)then
  FromRIEER4=0
else
  mantissa=ior(ishft(1,23),mantissa)
  if(i.gt.0)then
    FromRIEER4= mantissa*2.**ie exponent
  else
    FromRIEER4=-mantissa*2.**ie exponent
  endif
endif
end
```

---

Subject: Re: converting ieee-float-format to pw-wave-float-format

Posted by [thompson](#) on Fri, 26 Aug 1994 14:44:35 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

grunes@imsy1.nrl.navy.mil (Mitchell R Grunes) writes:

> No. IEEE does not define byte order--at least it is in oposite order  
> on PCs from the most common useage on most Unix Workstations (Suns,  
> SGIs,... -- but watch out for Dec Alpha, which someone told me use  
> backwards (least significant byte first, like the PC) byte order.

I've often seen the byte order with the most significant byte first, i.e.  
that use on most Unix workstations, referred to as the "network" byte order.  
Most interchange formats, such as FITS, which are meant to be transportable  
from machine to machine, use that byte order.

> One thing puzzles me: Most floating point numbers translate between  
> Suns and PCs simply through BYTEORDER. But NaN values seem to  
> behave differently. Does anyone know why?

If both machines follow the IEEE standard, then both should recognize the same NaN values. However, many bit patterns can represent NaN. Perhaps the two machines use a different \*default\* NaN value.

Bill Thompson

---