Subject: Re: 4-bit words
Posted by David S. Foster/Admin on Thu, 21 Dec 1995 08:00:00 GMT
View Forum Message <> Reply to Message

David Foster <foster@bial6.ucsd.edu> wrote:

In the code that I sent previously, you have to add a byte
conversion to the lines that increment LO and HI; change the
lines to be something like:

```
  lo = lo + byte(2L ^ long(i))
```

If speed is critical then you could leave this out and convert
to byte at the end (probably the way to go).

Dave Foster
foster@bial6.ucsd.edu

Subject: Re: 4-bit words
Posted by David S. Foster/Admin on Thu, 21 Dec 1995 08:00:00 GMT
View Forum Message <> Reply to Message

orbach@rockvax.rockefeller.edu (Darren Orbach) wrote:
>
> I have a file that consists of a 256*256 array of 4-bit
> words created in another application, written as a binary
> file. I need to manipulate this array by shifting these
> 4-bit elements to the right by various amounts, and wrapping
> around to the other side of the array. However, since the
> smallest data type in WAVE or IDL is a full byte, I don't see
> a straightforward way to do this. Any suggestions?

I think I would write a routine that reads in the data two
elements (8 bits) at a time, then extract the two elements
from the byte, and puts these two bytes into a
corresponding 256x256 array of bytes; then manipulate
this byte array.

Given a byte of data stored in 'value':

```
 val = value
 lo = 0B
 for i = 0, 3 do begin
  higher_bit = 2L ^ long(i+1)
  rem = val mod higher_bit
  if (rem ne 0) then begin
```

```
   lo = lo + 2L ^ long(i)
   val = val - rem
  endif
 endfor

 hi = 0B
 for i = 4, 7 do begin
  higher_bit = 2L ^ long(i+1)
  rem = val mod higher_bit
  if (rem ne 0) then begin
   hi = hi + 2L ^ long(i-4)
   val = val - rem
  endif
 endfor
```

This *should* give the lower and upper halves of the original byte,
as two byte values, 'lo' and 'hi'. I've tested this and it seems
to work. But hey, my job description says that my programming
only has to be 90% accurate!

Wouldn't it be nice if IDL had bitwise operators!

Dave Foster
foster@bial6.ucsd.edu

Subject: Re: 4-bit words
Posted by rivers on Thu, 21 Dec 1995 08:00:00 GMT

View Forum Message <> Reply to Message

In article <DJw3nF.D8n@rockyd.rockefeller.edu>, orbach@rockvax.rockefeller.edu (Darren Orbach) writes:
> I have a file that consists of a 256*256 array of 4-bit
> words created in another application, written as a binary
> file. I need to manipulate this array by shifting these
> 4-bit elements to the right by various amounts, and wrapping
> around to the other side of the array. However, since the
> smallest data type in WAVE or IDL is a full byte, I don't see
> a straightforward way to do this. Any suggestions?

I am assuming that your data files have the 4-bit values packed together. If so
then the following should create the array you want:

```
; Make a 1-D byte array big enough to hold image from disk
IDL> temp = bytarr(2L^15)
; Read in the data
IDL> readu, lun, temp
; Make new array to hold decomposed data
```

```
IDL> data = bytarr(2, 2L^15)
; Low order 4 bits in even elements
IDL> data(0,*) = (temp and '00FF'X)
; High order 4 bits in odd elements
IDL> data(1,*) = (temp/16 and '00FF'X)
; Reform into 256x256 array
IDL> data = reform(data, 256, 256)
```

____

Mark Rivers	(312) 702-2279 (office)
CARS	(312) 702-9951 (secretary)
Univ. of Chicago	(312) 702-5454 (FAX)
5640 S. Ellis Ave.	(708) 922-0499 (home)
Chicago, IL 60637	rivers@cars3.uchicago.edu (Internet)

Subject: Re: 4-bit words
Posted by orbach on Fri, 22 Dec 1995 08:00:00 GMT

View Forum Message <> Reply to Message

In message <DJxwrH.ou@midway.uchicago.edu> - rivers@cars3.uchicago.edu (Mark Rivers) writes:

```
:>
:>In article <DJw3nF.D8n@rockyd.rockefeller.edu>, orbach@rockvax.rockefeller.edu (Darren Orbach) writes:
:>>I have a file that consists of a 256*256 array of 4-bit
:>>words created in another application, written as a binary
:>>file.  I need to manipulate this array by shifting these
:>>4-bit elements to the right by various amounts, and wrapping
:>>around to the other side of the array.  However, since the
:>>smallest data type in WAVE or IDL is a full byte, I don't see
:>>a straightforward way to do this.  Any suggestions?
:>
:>I am assuming that your data files have the 4-bit values packed together. If so
:>then the following should create the array you want:
:>
:>; Make a 1-D byte array big enough to hold image from disk
:>IDL> temp = bytarr(2L^15)
:>; Read in the data
:>IDL> readu, lun, temp
:>; Make new array to hold decomposed data
:>IDL> data = bytarr(2, 2L^15)
:>; Low order 4 bits in even elements
:>IDL> data(0,*) = (temp and '00FF'X)
:>; High order 4 bits in odd elements
:>IDL> data(1,*) = (temp/16 and '00FF'X)
:>; Reform into 256x256 array
:>IDL> data = reform(data, 256, 256)
```

```
:>
:> ___________________________________________________________
:>Mark Rivers                    (312) 702-2279 (office)
:>CARS                          (312) 702-9951 (secretary)
:>Univ. of Chicago              (312) 702-5454 (FAX)
:>5640 S. Ellis Ave.            (708) 922-0499 (home)
:>Chicago, IL 60637             rivers@cars3.uchicago.edu (Internet)
:>
```

Mark's suggestion worked nicely, with the proviso that the hex
number to use is '000F'x. The rest of the procedure involved
bit-shifting to the right, keeping proper account of where
high bits are so formed & moving them separately, and then
re-packing the data(2, 2L^15) array back into a temp(1, 2L^15)
array with the operation temp = (data(0,*) or data(1,*)).
Thanks to all.

-Darren Orbach