
Subject: IDL DICOM writer

Posted by [Bhautik Joshi](#) on Tue, 08 Jan 2002 05:01:03 GMT

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

Hi all,

A little while back, Marc O'Brien (marcus@icr.ac.uk) posted his very nifty TIFF_to_DICOM.c utility. Just on its own, this piece of code is pretty legendary and I'd like to thank Marc for sharing it with all of us.

However, I needed a solution that I could quickly port between platforms, so I set out upon the task of writing a DICOM writer that uses 100% IDL code, loosely based on Marc's program, and, now I have dicom_writer v0.1, which should be attached to this newsgroup message :)

Please find attached two files - dicom_writer.pro and dicom_example.pro. Using dicom_writer is a snap:

```
> rows = 20
> cols = 20
> bpp = 1
> image = BYTESCL(indgen(rows,cols))
> dicom_writer, 'test.dcm', rows, cols, bpp, image
```

This would create a boring DICOM image called test.dcm in the current directory. To test if it works, use dicom_example to load the file. Look at the source for better instructions and explanations.

At the moment dicom_writer is extremely basic - it'll only write a single slice and I've filled in most of the important fields with dummy data. However, the code itself is very simple and can be easily customised. I'm still working on it, but if you make any improvements I'd love to hear about it :)

Next on the list is a better and more comprehensive way to fill in the fields in the DICOM header, and perhaps part 10 DICOM compliance (!!). Remember, its pretty raw at the moment and has little in the way of error checking; if it bites your pet or causes hallucinations then you probably need to wait for the next release :P

Anyway, hope some of you find it useful, and I'd love to hear if you do get it working successfully :)

Cheers,
Bhautik

--

/-----(____)----- \

```

| nbj@imag.wsahs.nsw.gov.au | phone: 0404032617 |..|--\ -moo |
| ICQ #: 2464537           | http://cow.mooh.org | |--|   |
|-----+-----\OO//| -----/
| international           |
| roast. my sanity has gone |
| its lost forever         |
\-----/
;
;
; NAME:
; DICOM_WRITER
;
;
; VERSION:
; 0.1
;
;
; PURPOSE:
; Generate a dicom file from within RSI IDL
;
;
; AUTHOR:
; Bhautik Joshi
;
;
; EMAIL:
; bjoshi@geocities.com
;
;
; HOMEPAGE:
; http://cow.mooh.org
;
;
; USE:
; DICOM_WRITER, filename rows, cols, bpp, image
;
;
; INPUT:
; filename - string containing name of dicom file to be written to
; rows - number of rows in image
; cols - number of columns in image
; bpp - number of _bytes_ per pixel in the image
; image - byte formatted version of image
;
;
; NOTES ON USAGE:
; * At the moment the program only writes to a single slice
; * The input image must be squashed into a 1D array of bytes before
;   it can be used in dicom_writer
; * bpp specifies the number of bytes (not bits!!) per pixel
; * Extra dicom tags can be easily added (see body of program)
; * There is little to no error-checking at the moment, so be
;   careful!
;
;
; EXAMPLE:
; Create a horrendously boring byte image and store it in a
; dicom file, ~/test.dcm :

```

```

;
; > rows = 20
; > cols = 20
; > bpp = 1
; > image = BYTESCL(indgen(rows,cols))
; > dicom_writer, ~/test.dcm, rows, cols, bpp, image
;
; HISTORY:
; Based on Marc O'Briens (marcus@icr.ac.uk) TIFF_to_DICOM.c
; version 0.1 08-01-2002 - first working version produced
;
; TODO:
; * Allow for more robust dicom writing
; * Expand the number of tags written (using DICOM data
;   dictionary)
; * Part 10 compliance (!!!!!!!!!!!!!)
;
; DISCLAIMER:
;
; Permission to use, copy, modify, and distribute this software and its
; documentation for any purpose and without fee is hereby granted,
; provided that the above copyright notice appear in all copies and that
; both that copyright notice and this permission notice appear in
; supporting documentation.
;
; This file is provided AS IS with no warranties of any kind. The author
; shall have no liability with respect to the infringement of copyrights,
; trade secrets or any patents by this file or any part thereof. In no
; event will the author be liable for any lost revenue or profits or
; other special, indirect and consequential damages.
;
; The author accepts no responsibility for any action arising from use of
; this package. The software is not guaranteed to write compliant DICOM
; files. If it causes damage to you or your system, you have been warned -
; this is a work in progress. If it bites your dog, its not my fault. If
; it causes you to curl up on the floor in the foetal position muttering
; about pixies and mushrooms, its not my fault. If it causes you or someone
; else to spontaneously burst into song and dance, its not my fault but
; I'd like to hear about it. You have been warned.
;
;
pro dicom_writer, filename, rows, cols, bpp, image

;dummy fill-in variables.
random= '123456'
SOPClass = '1.2.840.10008.5.1.4.1.1.20'
SOPInstance = '1.2.840.10008.5.1.4.1.1.20.1'
StudyID = '1.2.3.4'

```

```
StudyInstanceUID = SOPInstance + random
SeriesInstanceUID = StudyInstanceUID
RelFrameOfReferenceUID = StudyInstanceUID
SeriesInstanceUID = SeriesInstanceUID + '.1'
RelFrameOfReferenceUID = RelFrameOfReferenceUID + '.2'
StudyID = StudyID + 'SIGNA '
```

```
;image variables
Seriesnum=0
Imagenum=0
thickness=1.0
spacing='1.0\\1.0'
```

```
OPENW, 1, filename
```

```
; DICOM tags - feel free to add more!
```

```
;0008 tags
```

```
;MR type
WRITEU, 1, BYTE(generate_stringtag('0008'x,'0008'x,'ORIGINAL\\PRIMARY\\ OTHER'))
;Instance date
WRITEU, 1, BYTE(generate_stringtag('0008'x,'0012'x,'20020108'))
;Instance time
WRITEU, 1, BYTE(generate_stringtag('0008'x,'0013'x,'000000.00000'))
;SOP class
WRITEU, 1, BYTE(generate_stringtag('0008'x,'0016'x,SOPClass))
;SOP instance
WRITEU, 1, BYTE(generate_stringtag('0008'x,'0018'x,SOPInstance))
;Modality
WRITEU, 1, BYTE(generate_stringtag('0008'x,'0060'x,'MR'))
;Manufacturer
WRITEU, 1, BYTE(generate_stringtag('0008'x,'0070'x,'GE'))
;Study Physicians Name
; WRITEU, 1, BYTE(generate_stringtag('0008'x,'0012'x,'chewbacca wookiee'))
```

```
;0010 tags
```

```
;Patient name
WRITEU, 1, BYTE(generate_stringtag('0010'x,'0010'x,'Jabba the Hutt'))
;Patient ID
WRITEU, 1, BYTE(generate_stringtag('0010'x,'0020'x,'TK247'))
;Patient birth date
WRITEU, 1, BYTE(generate_stringtag('0010'x,'0030'x,'20010820'))
;Patient sex
WRITEU, 1, BYTE(generate_stringtag('0010'x,'0040'x,'M'))
```

;0018 tags

;Acquisition type

WRITEU, 1, BYTE(generate_stringtag('0018'x,'0023'x,'2D'))

;Slice thickness

WRITEU, 1, BYTE(generate_stringtag('0018'x,'0050'x,STRING(thickness)))

;0020 tags

;Study instance

WRITEU, 1, BYTE(generate_stringtag('0020'x,'000D'x,StudyInstanceUID))

;Series instance UID

WRITEU, 1, BYTE(generate_stringtag('0020'x,'000E'x,SeriesInstanceUID))

;StudyID

WRITEU, 1, BYTE(generate_stringtag('0020'x,'0010'x,StudyID))

;Series number

WRITEU, 1, BYTE(generate_stringtag('0020'x,'0011'x,STRING(seriesnum)))

;Image number

WRITEU, 1, BYTE(generate_stringtag('0020'x,'0013'x,STRING(imagenum)))

;0028 tags

;samples per pixel

WRITEU, 1, BYTE(generate_UStag('0028'x,'0002'x,1))

;Photometric interpretation

WRITEU, 1, BYTE(generate_stringtag('0028'x,'0004'x,'MONOCHROME2'))

;Rows in image

WRITEU, 1, BYTE(generate_UStag('0028'x,'0010'x,rows))

;Columns in image

WRITEU, 1, BYTE(generate_UStag('0028'x,'0011'x,rows))

;pixel spacing

WRITEU, 1, BYTE(generate_stringtag('0028'x,'0030'x,spacing))

;bits allocated per sample

WRITEU, 1, BYTE(generate_UStag('0028'x,'0100'x,bpp*8))

;bits stored per sample

WRITEU, 1, BYTE(generate_UStag('0028'x,'0101'x,bpp*8))

;high bit

WRITEU, 1, BYTE(generate_UStag('0028'x,'0102'x,(bpp*8)-1))

;pixel representation

WRITEU, 1, BYTE(generate_UStag('0028'x,'0103'x,'0000'x))

;write image data

imsize=rows*cols*bpp

WRITEU, 1, BYTE(generate_pixeltag('7FE0'x,'0010'x,imsize))

WRITEU, 1, BYTE(image)

```
CLOSE, 1
end
```

```
;write any tag
function generate_anytag, group, element, data
```

```
rs=[getbytes(group,2),getbytes(element,2)]
```

```
;correct to even length if necessary
```

```
bs=BYTE(data)
```

```
nl=n_elements(bs)
```

```
if ((nl mod 2) ne 0) then begin
```

```
bs=[bs,BYTE(0)]
```

```
nl=nl+1
```

```
end
```

```
;size of field
```

```
rs=[rs,getbytes(nl,2)]
```

```
;padding
```

```
rs=[rs,[0,0]]
```

```
;string itself
```

```
rs=[rs,bs]
```

```
return, rs
```

```
end
```

```
;generate string tag
```

```
function generate_stringtag, group, element, string
```

```
return, generate_anytag(group, element, BYTE(string))
```

```
end
```

```
;generate binary element (unsigned short) tag
```

```
function generate_UStag, group, element, val
```

```
param=getbytes(val,2)
```

```
return, generate_anytag(group,element,param)
```

```
end
```

```
;generate pixel tag
```

```
function generate_pixeltag, group, element, val
```

```
return, [getbytes(group,2),getbytes(element,2), getbytes(val,4)]
```

```
end
```

```
;generate unsigned long tag
```

```
function generate_ULtag, group, element, val
```

```
param=getbytes(val,4)
```

```
return, generate_anytag(group,element,param)
```

```
end
```

```

;convert a value, val, that is num bytes long, into
;a series of ordered bytes
function getbytes, val, num
  ret=BYTARR(num)
  offset=0
;work in big endian ONLY
;val=swap_endian(val)
byteorder,val,/SWAP_IF_LITTLE_ENDIAN
for i=0,(num-1) do begin
  tmpres=BYTE(ISHFT(val, offset) AND 255)
  ret[i]=tmpres
  offset=offset-8
endfor

return, ret
end

```

PRO Dicom_Example

```

filename=dialog_pickfile()
object = Obj_New('IDLffDicom')
ok = object->Read(filename)
IF NOT ok THEN BEGIN
  Print, 'File: ' + filename + ' cannot be read. Returning...'
  RETURN
ENDIF

```

```

name = object->GetValue('0010'x, '0010'x)
IF Ptr_Valid(name[0]) THEN Print, 'name: ', *name[0]

```

```

image = object->GetValue('7Fe0'x, '0010'x)
;IF Ptr_Valid(image[0]) THEN tvscl, BytScl(*image[0])

```

```

im_type = object->GetValue('0008'x, '0008'x)
IF Ptr_Valid(im_type[0]) THEN PRINT, 'im_type: ', *im_type[0]

```

```

modality = object->GetValue('0008'x, '0060'x)
IF Ptr_Valid(modality[0]) THEN PRINT, 'modality: ', *modality[0]

```

```

slice_spacing = object->GetValue('0018'x, '0088'x)
IF Ptr_Valid(slice_spacing[0]) THEN PRINT, 'slice_spacing: ', *slice_spacing[0]

```

```

image_number = object->GetValue('0020'x, '0013'x)
IF Ptr_Valid(image_number[0]) THEN PRINT, 'image_number: ', *image_number[0]

```

```

rows = object->GetValue('0028'x, '0010'x)
IF Ptr_Valid(rows[0]) THEN PRINT, 'rows: ', *rows[0]

```

```

cols = object->GetValue('0028'x,'0011'x)
IF Ptr_Valid(cols[0]) THEN PRINT, 'cols: ',*cols[0]

spatial_res = object->GetValue('0018'x,'1050'x)
IF Ptr_Valid(spatial_res[0]) THEN PRINT, 'spatial_res: ',*spatial_res[0]

aspect_ratio = object ->GetValue('0028'x,'0034'x)
IF Ptr_Valid(aspect_ratio[0]) THEN PRINT, 'aspect_ratio: ',*aspect_ratio[0]

rows = object->GetValue('0028'x,'0010'x)
IF Ptr_Valid(rows[0]) THEN PRINT, 'rows: ',*rows[0]

cols = object->GetValue('0028'x,'0011'x)
IF Ptr_Valid(cols[0]) THEN PRINT, 'cols: ',*cols[0]

xys=object->GetValue('0028'x,'0030'x)
IF Ptr_Valid(xys[0]) THEN PRINT, 'xys: ',*xys[0]

zs = object->GetValue('0018'x,'0050'x)
IF Ptr_Valid(zs[0]) THEN PRINT,'zs: ',*zs[0]

xys=*xys[0]
zs=*zs[0]
sp=strpos(xys,'\')
voxelsizes=[FLOAT(strmid(xys,0,sp)), FLOAT(strmid(xys,sp+1)),FLOAT(zs)]

xy=[FLOAT(*rows[0]),FLOAT(*cols[0])]

print, voxelsizes
print, xy
zzzz=*image[0]

Obj_Destroy, object

Ptr_Free, name
Ptr_Free, image

END

```

File Attachments

-
- 1) [dicom_writer.pro](#), downloaded 123 times
 - 2) [dicom_example.pro](#), downloaded 116 times
-

Subject: Re: IDL DICOM writer
Posted by [Bhautik Joshi](#) on Wed, 09 Jan 2002 01:01:25 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi all,

There were a couple of nasty little bugs in the first version of the dicom writer, so I've patched it up.

Now, it *should*:

- allocate a file unit to write to properly
- work on both big- and little-endian platforms
- work straight away without having to pre-compile it first

If you have any problems with it, or have any suggestions, please let me know and I'll patch it up as soon as I can :)

cheers,
Bhautik

```
--
/-----(\_)------ \
| nbj@imag.wsahs.nsw.gov.au | phone: 0404032617 |..|--\ -moo |
| ICQ #: 2464537           | http://cow.mooh.org | |--|   |
|-----+-----\OO//| -----/
| international           |
| roast. my sanity has gone |
| its lost forever         |
\-----/
;
;
; NAME:
; DICOM_WRITER
;
;
; VERSION:
; 0.11
;
; PURPOSE:
; Generate a dicom file from within RSI IDL
;
; AUTHOR:
; Bhautik Joshi
;
; EMAIL:
; bjoshi@geocities.com
;
; HOMEPAGE:
; http://cow.mooh.org
;
; USE:
; DICOM_WRITER, filename, bpp, image
;
; INPUT:
```

```

; filename - string containing name of dicom file to be written to
; bpp - number of _bytes_ per pixel in the image
; image - byte formatted version of image
;
; NOTES ON USAGE:
; * At the moment the program only writes to a single slice
; * The input image must be squashed into a 1D array of bytes before
;   it can be used in dicom_writer
; * bpp specifies the number of bytes (not bits!!) per pixel
; * Extra dicom tags can be easily added (see body of program)
; * There is little to no error-checking at the moment, so be
;   careful!
; * Analyse seems to need a minimum image size of somewhere around
;   100x100
;
; EXAMPLE:
; Create a horrendously boring byte image and store it in a
; dicom file, test.dcm :
;
; > rows = 200
; > cols = 200
; > bpp = 1
; > image = BYTESCL(indgen(rows,cols))
; > dicom_writer, 'test.dcm', bpp, image
;
; HISTORY:
; Based on Marc O'Briens (marcus@icr.ac.uk) TIFF_to_DICOM.c
; version 0.1 08-01-2002 - first working version produced
; version 0.11 09-01-2002 - fixed endian-ness issue & added get_lun
;   functionality
;
; TODO:
; * Allow for more robust dicom writing
; * Expand the number of tags written (using DICOM data
;   dictionary)
; * Part 10 compliance (!!!!!!!!!!!)
;
; DISCLAIMER:
;
; Permission to use, copy, modify, and distribute this software and its
; documentation for any purpose and without fee is hereby granted,
; provided that the above copyright notice appear in all copies and that
; both that copyright notice and this permission notice appear in
; supporting documentation.
;
; This file is provided AS IS with no warranties of any kind. The author
; shall have no liability with respect to the infringement of copyrights,
; trade secrets or any patents by this file or any part thereof. In no

```

```

; event will the author be liable for any lost revenue or profits or
; other special, indirect and consequential damages.
;
; The author accepts no responsibility for any action arising from use of
; this package. The software is not guaranteed to write compliant DICOM
; files. If it causes damage to you or your system, you have been warned -
; this is a work in progress. If it bites your dog, its not my fault. If
; it causes you to curl up on the floor in the foetal position muttering
; about pixies and mushrooms, its not my fault. If it causes you or someone
; else to spontaneously burst into song and dance, its not my fault but
; I'd like to hear about it. You have been warned.
;
;

```

```

;convert a value, val, that is num bytes long, into
;a series of ordered bytes

```

```

function getbytes, val, num
  ret=BYTARR(num)
  offset=0
;work in big endian ONLY
;val=swap_endian(val)
if (!version.arch eq 'x86') then begin
  byteorder, val, /SWAP_IF_BIG_ENDIAN
endif else begin
  byteorder, val, /SWAP_IF_LITTLE_ENDIAN
endelse
for i=0,(num-1) do begin
  tmpres=BYTE(ISHFT(val, offset) AND 255)
  ret[i]=tmpres
  offset=offset-8
endfor

  return, ret
end

```

```

;write any tag
function generate_anytag, group, element, data

```

```

  rs=[getbytes(group,2),getbytes(element,2)]

```

```

;correct to even length if necessary

```

```

  bs=BYTE(data)
  nl=n_elements(bs)
  if ((nl mod 2) ne 0) then begin
    bs=[bs,BYTE(0)]
    nl=nl+1
  end

```

```

;size of field

```

```

  rs=[rs,getbytes(nl,2)]

```

```

;padding
rs=[rs,[0,0]]
;string itself
rs=[rs,bs]

return, rs
end

;generate string tag
function generate_stringtag, group, element, string
return, generate_anytag(group, element, BYTE(string))
end

;generate binary element (unsigned short) tag
function generate_UStag, group, element, val
param=getbytes(val,2)
return, generate_anytag(group,element,param)
end

;generate pixel tag
function generate_pixeltag, group, element, val
return, [getbytes(group,2),getbytes(element,2), getbytes(val,4)]
end

;generate unsigned long tag
function generate_ULtag, group, element, val
param=getbytes(val,4)
return, generate_anytag(group,element,param)
end

pro dicom_writer, filename, bpp, image

;dummy fill-in variables.
random= '123456'
SOPClass = '1.2.840.10008.5.1.4.1.1.20'
SOPInstance = '1.2.840.10008.5.1.4.1.1.20.1'
StudyID = '1.2.3.4'
StudyInstanceUID = SOPInstance + random
SeriesInstanceUID = StudyInstanceUID
RelFrameOfReferenceUID = StudyInstanceUID
SeriesInstanceUID = SeriesInstanceUID + '.1'
RelFrameOfReferenceUID = RelFrameOfReferenceUID + '.2'
StudyID = StudyID + 'SIGNA '

;image variables
Seriesnum=0
Imagenum=0

```

```
thickness=1.0
spacing='1.0\1.0'
sz=size(image)
rows=sz[1]
cols=sz[2]
```

```
GET_LUN, U
```

```
OPENW, U, filename
```

```
; DICOM tags - feel free to add more!
```

```
;0008 tags
```

```
;MR type
```

```
WRITEU, U, BYTE(generate_stringtag('0008'x,'0008'x,'ORIGINAL\PRIMARY\ OTHER'))
```

```
;Instance date
```

```
WRITEU, U, BYTE(generate_stringtag('0008'x,'0012'x,'20020108'))
```

```
;Instance time
```

```
WRITEU, U, BYTE(generate_stringtag('0008'x,'0013'x,'000000.00000'))
```

```
;SOP class
```

```
WRITEU, U, BYTE(generate_stringtag('0008'x,'0016'x,SOPClass))
```

```
;SOP instance
```

```
WRITEU, U, BYTE(generate_stringtag('0008'x,'0018'x,SOPInstance))
```

```
;Modality
```

```
WRITEU, U, BYTE(generate_stringtag('0008'x,'0060'x,'MR'))
```

```
;Manufacturer
```

```
WRITEU, U, BYTE(generate_stringtag('0008'x,'0070'x,'GE'))
```

```
;Study Physicians Name
```

```
; WRITEU, U, BYTE(generate_stringtag('0008'x,'0012'x,'chewbacca wookiee'))
```

```
;0010 tags
```

```
;Patient name
```

```
WRITEU, U, BYTE(generate_stringtag('0010'x,'0010'x,'Jabba the Hutt'))
```

```
;Patient ID
```

```
WRITEU, U, BYTE(generate_stringtag('0010'x,'0020'x,'TK247'))
```

```
;Patient birth date
```

```
WRITEU, U, BYTE(generate_stringtag('0010'x,'0030'x,'20010820'))
```

```
;Patient sex
```

```
WRITEU, U, BYTE(generate_stringtag('0010'x,'0040'x,'M'))
```

```
;0018 tags
```

```
;Acquisition type
```

```
WRITEU, U, BYTE(generate_stringtag('0018'x,'0023'x,'2D'))
```

```
;Slice thickness
```

```

WRITEU, U, BYTE(generate_stringtag('0018'x,'0050'x,STRING(thickness)))

;0020 tags

;Study instance
WRITEU, U, BYTE(generate_stringtag('0020'x,'000D'x,StudyInstanceUID))
;Series instance UID
WRITEU, U, BYTE(generate_stringtag('0020'x,'000E'x,SeriesInstanceUID))
;StudyID
WRITEU, U, BYTE(generate_stringtag('0020'x,'0010'x,StudyID))
;Series number
WRITEU, U, BYTE(generate_stringtag('0020'x,'0011'x,STRING(seriesnum)))
;Image number
WRITEU, U, BYTE(generate_stringtag('0020'x,'0013'x,STRING(imagenum)))

;0028 tags

;samples per pixel
WRITEU, U, BYTE(generate_UStag('0028'x,'0002'x,1))
;Photometric interpretation
WRITEU, U, BYTE(generate_stringtag('0028'x,'0004'x,'MONOCHROME2'))
;Rows in image
WRITEU, U, BYTE(generate_UStag('0028'x,'0010'x,rows))
;Columns in image
WRITEU, U, BYTE(generate_UStag('0028'x,'0011'x,cols))
;pixel spacing
WRITEU, U, BYTE(generate_stringtag('0028'x,'0030'x,spacing))
;bits allocated per sample
WRITEU, U, BYTE(generate_UStag('0028'x,'0100'x,bpp*8))
;bits stored per sample
WRITEU, U, BYTE(generate_UStag('0028'x,'0101'x,bpp*8))
;high bit
WRITEU, U, BYTE(generate_UStag('0028'x,'0102'x,(bpp*8)-1))
;pixel representation
WRITEU, U, BYTE(generate_UStag('0028'x,'0103'x,'0000'x))

;write image data
imsize=rows*cols*bpp

WRITEU, U, BYTE(generate_pixeltag('7FE0'x,'0010'x,imsize))

WRITEU, U, BYTE(image)

CLOSE, U
FREE_LUN, U
end

```

File Attachments

1) [dicom_writer.pro](#), downloaded 137 times

Subject: Re: IDL DICOM writer
Posted by [gogosgogos](#) on Wed, 09 Jan 2002 23:45:45 GMT
[View Forum Message](#) <> [Reply to Message](#)

excellent!
good job :)

Subject: Re: IDL DICOM writer
Posted by [m.lowry](#) on Thu, 10 Jan 2002 10:21:47 GMT
[View Forum Message](#) <> [Reply to Message](#)

Great job, just what I've been looking for!

Cheers!

Subject: Re: IDL DICOM writer
Posted by [Marcus O'Brien](#) on Fri, 11 Jan 2002 15:51:06 GMT
[View Forum Message](#) <> [Reply to Message](#)

Hi Bhautik,

Thanks for all the nice comments about TIFF_TO_DICOM.c, if I'd realized anyone would appreciate it I'd have made certain the file comments actually reflected its current usage, DUH! Oh and that I had my current e-mail address on it (m.obrien@sghms.ac.uk).

Love the DICOM_WRITER for IDL, took a little bit of playing for ctn dicom software to open the created files. I believe that the minimum bits per pixel supported by the standard is 16. I've attached a slightly hacked version that sets bpp internally irrespective of the image type coming in. Had played with the idea of allowing floats and setting pixel representation according to image array type, but settled on bytscl and a conversion to uint for now. See what you think.

Trivial bit, physicians name tag is 0008,0090 (some readers will drop the file if the tags are out of order).

I'm sure all the med image people out there that use IDL love you :)

Thanks for an excellent bit of porting, I'd been putting it off for ages.

Marc

--

Tel: 0208 725 2857

Fax: 0208 725 2992

St George's Hospital Medical School
Department of Biochemistry and Immunology
Cranmer Terrace
Tooting
London SW17 0RE

;

; NAME:

; DICOM_WRITER

;

; VERSION:

; 0.11

;

; PURPOSE:

; Generate a dicom file from within RSI IDL

;

; AUTHOR:

; Bhautik Joshi

;

; EMAIL:

; bjoshi@geocities.com

;

; HOMEPAGE:

; <http://cow.mooh.org>

;

; USE:

; DICOM_WRITER, filename, image

;

; INPUT:

; filename - string containing name of dicom file to be written to

; bpp - number of `_bytes_` per pixel in the image

; image - byte formatted version of image

;

; NOTES ON USAGE:

; * At the moment the program only writes to a single slice

; * The input image must be squashed into a 1D array of bytes before

; it can be used in `dicom_writer`

; * `bpp` specifies the number of bytes (not bits!!) per pixel

; * Extra dicom tags can be easily added (see body of program)

; * There is little to no error-checking at the moment, so be

; careful!

; * Analyse seems to need a minimum image size of somewhere around

; 100x100

```

;
; EXAMPLE:
; Create a horrendously boring byte image and store it in a
; dicom file, test.dcm :
;
; > rows = 200
; > cols = 200
; > bpp = 1
; > image = BYTESCL(indgen(rows,cols))
; > dicom_writer, 'test.dcm', bpp, image
;
; HISTORY:
; Based on Marc O'Briens (marcus@sghms.ac.uk) TIFF_to_DICOM.c
; version 0.1 08-01-2002 - first working version produced
; version 0.11 09-01-2002 - fixed endian-ness issue & added get_lun
;     functionality
;
; TODO:
; * Allow for more robust dicom writing
; * Expand the number of tags written (using DICOM data
;   dictionary)
; * Part 10 compliance (!!!!!!!!!!!!!)
;
; DISCLAIMER:
;
; Permission to use, copy, modify, and distribute this software and its
; documentation for any purpose and without fee is hereby granted,
; provided that the above copyright notice appear in all copies and that
; both that copyright notice and this permission notice appear in
; supporting documentation.
;
; This file is provided AS IS with no warranties of any kind. The author
; shall have no liability with respect to the infringement of copyrights,
; trade secrets or any patents by this file or any part thereof. In no
; event will the author be liable for any lost revenue or profits or
; other special, indirect and consequential damages.
;
; The author accepts no responsibility for any action arising from use of
; this package. The software is not guaranteed to write compliant DICOM
; files. If it causes damage to you or your system, you have been warned -
; this is a work in progress. If it bites your dog, its not my fault. If
; it causes you to curl up on the floor in the foetal position muttering
; about pixies and mushrooms, its not my fault. If it causes you or someone
; else to spontaneously burst into song and dance, its not my fault but
; I'd like to hear about it. You have been warned.
;
;convert a value, val, that is num bytes long, into

```

```

;a series of ordered bytes
function getbytes, val, num
  ret=BYTARR(num)
  offset=0
;work in big endian ONLY
;val=swap_endian(val)
if (!version.arch eq 'x86') then begin
  byteorder,val,/SWAP_IF_BIG_ENDIAN
endif else begin
  byteorder,val,/SWAP_IF_LITTLE_ENDIAN
endelse
for i=0,(num-1) do begin
  tmpres=BYTE(ISHFT(val, offset) AND 255)
  ret[i]=tmpres
  offset=offset-8
endfor

return, ret
end

;write any tag
function generate_anytag, group, element, data

  rs=[getbytes(group,2),getbytes(element,2)]

;correct to even length if necessary
bs=BYTE(data)
nl=n_elements(bs)
if ((nl mod 2) ne 0) then begin
  bs=[bs,BYTE(0)]
  nl=nl+1
end
;size of field
rs=[rs,getbytes(nl,2)]
;padding
rs=[rs,[0,0]]
;string itself
rs=[rs,bs]

return, rs
end

;generate string tag
function generate_stringtag, group, element, string
return, generate_anytag(group, element, BYTE(string))
end

;generate binary element (unsigned short) tag

```

```

function generate_UStag, group, element, val
  param=getbytes(val,2)
  return, generate_anytag(group,element,param)
end

;generate pixel tag
function generate_pixeltag, group, element, val
  return, [getbytes(group,2),getbytes(element,2), getbytes(val,4)]
end

;generate unsigned long tag
function generate_ULtag, group, element, val
  param=getbytes(val,4)
  return, generate_anytag(group,element,param)
end

pro dicom_writer, filename, image
; bytescl and convert input image to 16bit unsigned integer
sz=size(image)
image = reform(UINT(BYTSCCL(image)),sz[1],sz[2])

;dummy fill-in variables.
random= '123456'
SOPClass = '1.2.840.10008.5.1.4.1.1.20'
SOPInstance = '1.2.840.10008.5.1.4.1.1.20.1'
StudyID = '1.2.3.4'
StudyInstanceUID = SOPInstance + random
SeriesInstanceUID = StudyInstanceUID
RelFrameOfReferenceUID = StudyInstanceUID
SeriesInstanceUID = SeriesInstanceUID + '.1'
RelFrameOfReferenceUID = RelFrameOfReferenceUID + '.2'
StudyID = StudyID + 'SIGNA '

;image variables
Seriesnum=0
Imagenum=0
thickness=1.0
spacing='1.0\\1.0'
rows=sz[1]
cols=sz[2]
bpp=2

GET_LUN, U

OPENW, U, filename

; DICOM tags - feel free to add more!

```

;0008 tags

;MR type

WRITEU, U, BYTE(generate_stringtag('0008'x,'0008'x,'ORIGINAL\\PRIMARY\\ OTHER'))

;Instance date

WRITEU, U, BYTE(generate_stringtag('0008'x,'0012'x,'20020108'))

;Instance time

WRITEU, U, BYTE(generate_stringtag('0008'x,'0013'x,'000000.00000'))

;SOP class

WRITEU, U, BYTE(generate_stringtag('0008'x,'0016'x,SOPClass))

;SOP instance

WRITEU, U, BYTE(generate_stringtag('0008'x,'0018'x,SOPInstance))

;Modality

WRITEU, U, BYTE(generate_stringtag('0008'x,'0060'x,'MR'))

;Manufacturer

WRITEU, U, BYTE(generate_stringtag('0008'x,'0070'x,'GE'))

;Study Physicians Name

WRITEU, U, BYTE(generate_stringtag('0008'x,'0090'x,'fizzy'))

;0010 tags

;Patient name

WRITEU, U, BYTE(generate_stringtag('0010'x,'0010'x,'A Patient'))

;Patient ID

WRITEU, U, BYTE(generate_stringtag('0010'x,'0020'x,'TK247'))

;Patient birth date

WRITEU, U, BYTE(generate_stringtag('0010'x,'0030'x,'20020111'))

;Patient sex

WRITEU, U, BYTE(generate_stringtag('0010'x,'0040'x,'F'))

;0018 tags

;Acquisition type

WRITEU, U, BYTE(generate_stringtag('0018'x,'0023'x,'2D'))

;Slice thickness

WRITEU, U, BYTE(generate_stringtag('0018'x,'0050'x,STRING(thickness)))

;0020 tags

;Study instance

WRITEU, U, BYTE(generate_stringtag('0020'x,'000D'x,StudyInstanceUID))

;Series instance UID

WRITEU, U, BYTE(generate_stringtag('0020'x,'000E'x,SeriesInstanceUID))

;StudyID

WRITEU, U, BYTE(generate_stringtag('0020'x,'0010'x,StudyID))

;Series number

```

WRITEU, U, BYTE(generate_stringtag('0020'x,'0011'x,STRING(seriesnum)))
;Image number
WRITEU, U, BYTE(generate_stringtag('0020'x,'0013'x,STRING(imagenum)))

;0028 tags

;samples per pixel
WRITEU, U, BYTE(generate_UStag('0028'x,'0002'x,1))
;Photometric interpretation
WRITEU, U, BYTE(generate_stringtag('0028'x,'0004'x,'MONOCHROME2'))
;Rows in image
WRITEU, U, BYTE(generate_UStag('0028'x,'0010'x,rows))
;Columns in image
WRITEU, U, BYTE(generate_UStag('0028'x,'0011'x,cols))
;pixel spacing
WRITEU, U, BYTE(generate_stringtag('0028'x,'0030'x,spacing))
;bits allocated per sample
WRITEU, U, BYTE(generate_UStag('0028'x,'0100'x,bpp*8))
;bits stored per sample
WRITEU, U, BYTE(generate_UStag('0028'x,'0101'x,bpp*8))
;high bit
WRITEU, U, BYTE(generate_UStag('0028'x,'0102'x,(bpp*8)-1))
;pixel representation
WRITEU, U, BYTE(generate_UStag('0028'x,'0103'x,'0000'x))

;write image data
imsize=rows*cols*bpp

WRITEU, U, BYTE(generate_pixeltag('7FE0'x,'0010'x,imsize))
WRITEU, U, image

CLOSE, U
FREE_LUN, U
end

```

File Attachments

1) [dicom_writer.pro](#), downloaded 116 times

Subject: IDL DICOM writer v0.2
 Posted by [Bhautik Joshi](#) on Mon, 14 Jan 2002 22:51:27 GMT
[View Forum Message](#) <> [Reply to Message](#)

G'day all,

I've updated the source code for my (open source (free!(as in 'free beer')) DICOM writer for RSI's IDL. Currently, its main features are:

- * Generates implicit VR DICOM files (little endian)
- * Handles most VR tags (except for PN and SQ)
- * Generates single slice images, from most integer (BYTE, FIX, UINT, LONG, ULONG) data types
- * Seems to handle endian-ness issues on tested platforms (win32 (little endian) and SPARC (big endian))
- * Has a quick & easy mode for generating really simple DICOM files with important tags filled in with dummy values
- * Relatively simple program structure means it can be easily extended to write more tags

It only writes basic single slice DICOM files (ie not part 10 files), and I can't guarantee DICOM compliance. However, as a simple DICOM writer that uses pure IDL code (ie no .DLL's or libraries or whatever) you might find it useful. The source code itself is fairly simple so it is also quite easy to extend it so that you can add more tags to be written.

It can be found here:

<http://cow.mooh.org/idl.shtml>

and soon it will be available on David Fannings (absolutely legendary) site:

<http://www.dfanning.com>

If you have any queries, comments or suggestions, please don't hesitate to contact me as I'd love to hear about it.

Cheers,
Bhautik Joshi

ps. thanks to Marc O'Brien (m.obrien@sghms.ac.uk) for his hints and source code, and David Fanning for hosting the files :)

```
--
/-----+-----(\_)- - - - -\
| bjoshi@nospam.geocities.com | phone: 0404032617 |..|--\ -moo |
| ICQ #: 2464537           | http://cow.mooh.org | |--| |
|-----+-----\OO/|| - - - - -/
| international           |
| roast. my sanity has gone |
| its lost forever         |
\-----/
```