```
!-----------------------------------------------------------------------
! write_grads2nc.f
!
! This is the second module of the "grads2nc" program.  It converts a
! GrADS control file (with associated IEEE data file) into a netCDF file.
!
! Written by Hans Vahlenkamp
! Geophysical Fluid Dynamics Laboratory/NOAA
! Princeton University Forrestal Campus
! Last updated: 3/5/98
!-----------------------------------------------------------------------

#define MAXSTRLEN 256

      subroutine write_grads2nc(binfile, ncfile, gtitle, misval, xlen,
     &                          ylen, zlen, tlen, xvals, yvals, zvals,
     &                          tvals, nzdims, dimzlens, varzdims,
     &                          nvars, varzlens, varnames, vardescs,
     &                          valuesxy, varranges, byteswapped, yrev,
     &                          zrev, timeunits, calcrange, verbose,
     &                          quiet, mxdimout, ndims, dimvals,
     &                          lendims, cdimnam, cdimlnam, cdimunits,
     &                          cart_axis, ipositive, cvarnam, cvarlnam,
     &                          cvarunits, nbits,idimindx, nvdims,
     &                          maxnitems, validr,vspval, lspval,
     &                          istart, icount)

#include "netcdf.inc"

      character*(MAXSTRLEN) binfile, gtitle
      character*(MAXNCNAM) ncfile
      real misval
      integer xlen, ylen, zlen, tlen
      real xvals(xlen), yvals(ylen), zvals(zlen), tvals(tlen)
      integer nzdims, nvars
      integer dimzlens(nzdims), varzdims(nvars), varzlens(nvars)
      character*(MAXNCNAM) varnames(nvars)
      character*(MAXSTRLEN) vardescs(nvars)
      real valuesxy(xlen,ylen), value, varranges(2,nvars)
      integer byteswapped, calcrange, verbose, quiet, yrev, zrev
      character*MAXSTRLEN timeunits

      logical lclobber, lgspval
      real gspval
      integer icaltype, icalaux, iverbose
      integer mxdimout, ndims
      real dimvals(mxdimout,ndims)
      integer lendims(ndims)
      character*(MAXNCNAM) cdimnam(ndims)
      character*(MAXSTRLEN) cdimlnam(ndims), cdimunits(ndims)
      character*1 cart_axis(ndims)
      integer ipositive(ndims)
      character*(MAXNCNAM) cvarnam(nvars)
      character*(MAXSTRLEN) cvarlnam(nvars), cvarunits(nvars)
      integer nbits(nvars), idimindx(ndims,nvars)
      integer nvdims(nvars), maxnitems(nvars)
      real validr(2,nvars), vspval(nvars)
      logical lspval(nvars)
      integer istart(ndims), icount(ndims)
      real*8 time(1)

      integer yreorder, zreorder  ! Flip the data along the Y and Z axes
      integer yipos, zipos  ! "ipositive" attribute for the Y and Z axes
      integer zdn, tdn  ! Dimension index of the Y and Z axes
      integer i, j, k, l, x, y, y2  ! Loop variables
      integer recnum  ! Current record (horizontal level slice) number
      character*(20) zname  ! Name of a constructed Z dimension
      real*8 recval(1)  ! Record (time coordinate) value
      real minval, maxval, range(2)  ! Min and max variable values
      external byte_swap  ! C function for swapping bytes in a word

! Specify some general file info

      lclobber=.true.
      lgspval=.true.
      gspval=misval
      icaltype=3  ! GrADS assumes the Gregorian calendar
      icalaux=0
```

```
      iverbose=0

! Copy the coordinates for the X axis

      do i=1,xlen
        dimvals(i,1)=xvals(i)
      enddo

! Copy the coordinates for the Y axis

      if (ylen .eq. 1) then
        yipos=1
        dimvals(1,2)=yvals(1)
      else
        if (yvals(1) .lt. yvals(2)) then
          yipos=1
          do j=1,ylen
            dimvals(j,2)=yvals(j)
          enddo
          if (yrev .eq. 0) then
            yreorder=0
          else
            yreorder=1
          endif
        else
          yipos=-1
          do j=1,ylen
            dimvals(j,2)=yvals(ylen-j+1)
          enddo
          if (yrev .eq. 0) then
            yreorder=1
          else
            yreorder=0
          endif
        endif
      endif

! Copy the coordinates for the Z axes

      if (zlen .eq. 1) then
        zipos=1
        do kz=1,nzdims
          zdn=2+kz
          dimvals(1,zdn)=zvals(1)
        enddo
      else
        if (zvals(1) .lt. zvals(2)) then
          zipos=1
          do kz=1,nzdims
            zdn=2+kz
            do k=1,dimzlens(kz)
              dimvals(k,zdn)=zvals(k)
            enddo
          enddo
          if (zrev .eq. 0) then
            zreorder=0
          else
            zreorder=1
          endif
        else
          zipos=-1
          do kz=1,nzdims
            zdn=2+kz
            do k=1,dimzlens(kz)
              dimvals(k,zdn)=zvals(dimzlens(kz)-k+1)
            enddo
          enddo
          if (zrev .eq. 0) then
            zreorder=1
          else
            zreorder=0
          endif
        endif
      endif

! Specify the dimension information

      tdn=ndims
```

```
      lendims(1)=xlen
      lendims(2)=ylen
      lendims(tdn)=0
      cdimnam(1)='longitude'
      cdimnam(2)='latitude'
      cdimnam(tdn)='time'
      cdimlnam(1)='Longitude values'
      cdimlnam(2)='Latitudes values'
      cdimlnam(tdn)='Time coordinate values'
      cdimunits(1)='degrees_E'
      cdimunits(2)='degrees_N'
      cdimunits(tdn)=timeunits
      cart_axis(1)='X'
      cart_axis(2)='Y'
      cart_axis(tdn)='T'
      ipositive(1)=1
      ipositive(2)=yipos
      ipositive(tdn)=1
      do kz=1,nzdims
        zdn=2+kz
        lendims(zdn)=dimzlens(kz)
        write(zname,'(i3.3)') lendims(zdn)
        cdimnam(zdn)='level'//zname
        cdimlnam(zdn)='Level values'
        cdimunits(zdn)='level'
        cart_axis(zdn)='Z'
        ipositive(zdn)=zipos
      enddo

! Specify the variable information

      do i=1,nvars
        cvarnam(i)=varnames(i)
        cvarlnam(i)=vardescs(i)
        cvarunits(i)='unknown'
        nbits(i)=32
        idimindx(1,i)=1
        idimindx(2,i)=2
        if (varzlens(i) .eq. 0) then
          nvdims(i)=3
          idimindx(3,i)=tdn
        else
          nvdims(i)=4
          idimindx(3,i)=2+varzdims(i)
          idimindx(4,i)=tdn
        endif
        maxnitems(i)=0
        validr(1,i)=-1e+37
        validr(2,i)=1e+37
        lspval(i)=.true.
        vspval(i)=misval
      enddo

! Create the netCDF file and set up its structure

      call ncsetup(ncfile,lclobber,gtitle,lgspval,gspval,ndims,lendims,
     &             cdimnam,cdimlnam,cdimunits,cart_axis,ipositive,
     &             dimvals,mxdimout,time,icaltype,icalaux,nvars,cvarnam,
     &             cvarlnam,cvarunits,nbits,nvdims,idimindx,ndims,
     &             maxnitems,validr,lspval,vspval,iverbose)

! Open the IEEE data file

#ifdef HOST_T90
      call asnunit(10,'-F null -N ieee_32',ierror)
      open(unit=10,file=binfile,form='unformatted',access='sequential',
     &     status='old')
#else
      open(unit=10,file=binfile,form='unformatted',access='direct',
     &     status='old',recl=xlen*ylen)
#endif

! Read the records (horizontal slabs) and store them in the netCDF file

      recnum=1
      istart(1)=1
      istart(2)=1
      icount(1)=xlen
```

```
      icount(2)=ylen
      do l=1,tlen
        if (quiet .ne. 1) then
          write(*,'(a12,i5)') 'Time step = ',l
        endif
        recval(1)=tvals(l)
        do i=1,nvars
          if ((quiet .ne. 1) .and. (verbose .eq. 1)) then
            write(*,'(a24,a50)') '   Processing variable: ',varnames(i)
          endif
          if (varzlens(i) .gt. 0) then  ! Variable contains Z levels
            do k=1,varzlens(i)
              if ((quiet .ne. 1) .and. (verbose .eq. 1)) then
                write(*,'(a14,i5)') '      level = ',k
              endif
#ifdef HOST_T90
              read(10) valuesxy
#else
              read(10,rec=recnum) valuesxy
#endif
              if (byteswapped .eq. 1) then
                call byte_swap(valuesxy,xlen,ylen)
              endif
              if (yreorder .eq. 1) then
                do y=1,ylen/2
                  do x=1,xlen
                    value=valuesxy(x,y)
                    y2=ylen-y+1
                    valuesxy(x,y)=valuesxy(x,y2)
                    valuesxy(x,y2)=value
                  enddo
                enddo
              endif
              if (zreorder .eq. 0) then
                istart(3)=k
              else
                istart(3)=varzlens(i)-k+1
              endif
              istart(4)=l
              icount(3)=1
              icount(4)=1
              call ncstore(i,istart,icount,valuesxy,recval,iverbose)
              if (calcrange .eq. 1) then
                if ((l .eq. 1) .and. (k .eq. 1)) then
                  varranges(1,i)=valuesxy(1,1)
                  varranges(2,i)=valuesxy(1,1)
                  do y=1,ylen
                    do x=1,xlen
                      if (valuesxy(x,y) .ne. misval) then
                        varranges(1,i)=valuesxy(x,y)
                        varranges(2,i)=valuesxy(x,y)
                        goto 100
                      endif
                    enddo
                  enddo
100               continue
                endif
                do y=1,ylen
                  do x=1,xlen
                    if (valuesxy(x,y) .ne. misval) then
                      if (valuesxy(x,y) .lt. varranges(1,i)) then
                        varranges(1,i)=valuesxy(x,y)
                      endif
                      if (valuesxy(x,y) .gt. varranges(2,i)) then
                        varranges(2,i)=valuesxy(x,y)
                      endif
                    endif
                  enddo
                enddo
              endif
              recnum=recnum+1
            enddo
          else  ! Variable has no Z levels
            if ((quiet .ne. 1) .and. (verbose .eq. 1)) then
              write(*,'(a19)') '      level =      0'
            endif
#ifdef HOST_T90
            read(10) valuesxy
```

```
#else
            read(10,rec=recnum) valuesxy
#endif
            if (byteswapped .eq. 1) then
              call byte_swap(valuesxy,xlen,ylen)
            endif
            if (yreorder .eq. 1) then
              do y=1,ylen/2
                do x=1,xlen
                  value=valuesxy(x,y)
                  y2=ylen-y+1
                  valuesxy(x,y)=valuesxy(x,y2)
                  valuesxy(x,y2)=value
                enddo
              enddo
            endif
            istart(3)=l
            icount(1)=xlen
            icount(2)=ylen
            icount(3)=1
            call ncstore(i,istart,icount,valuesxy,recval,iverbose)
            if (calcrange .eq. 1) then
              if (l .eq. 1) then
                varranges(1,i)=valuesxy(1,1)
                varranges(2,i)=valuesxy(1,1)
                do y=1,ylen
                  do x=1,xlen
                    if (valuesxy(x,y) .ne. misval) then
                      varranges(1,i)=valuesxy(x,y)
                      varranges(2,i)=valuesxy(x,y)
                      goto 200
                    endif
                  enddo
                enddo
200             continue
              endif
              do y=1,ylen
                do x=1,xlen
                  if (valuesxy(x,y) .ne. misval) then
                    if (valuesxy(x,y) .lt. varranges(1,i)) then
                      varranges(1,i)=valuesxy(x,y)
                    endif
                    if (valuesxy(x,y) .gt. varranges(2,i)) then
                      varranges(2,i)=valuesxy(x,y)
                    endif
                  endif
                enddo
              enddo
            endif
            recnum=recnum+1
          endif
        enddo
      enddo

! Set the actual "valid_range" of the variables if necessary

      if (calcrange .eq. 1) then
        do i=1,nvars
          range(1)=varranges(1,i)
          range(2)=varranges(2,i)
          call ncaddatt(2,i,'valid_range',5,2,range)
        enddo
      endif

! Done

      close(10)
      call ncdone

      end
```