BUFR/PrepBUFR File Processing

Ruifang Li* and Ming Hu**

*NCAR/MMM

**Developmental Testbed Center

Topics Covered

- Basic actions and steps of BUFR file processing
- Basic structure of BUFR/PrepBUFR files
- Community tools for BUFR/PrepBUFR users
  - 10 examples are provided in the community GSI release version 3 package
  - Detailed instructions to these examples are in the Chapter 2 of the BUFR User's Guide

BUFR Processing Actions and Steps

Actions and steps

BUFR Processing Actions

- **Encode**:
  - **Write** the observations into a new BUFR file
  - need to follow WMO defined BUFR format to save the data
- **Decode**:
  - **Read** the observations from a BUFR file
- **Append**:
  - **Add** the observations to the end of an existing BUFR file

BUFR Encoding Steps

Step 1: Write the Fortran Code "*encode_temperature.f90*"

```
Line 01:    program bufr_encode_temperature

Line 02:     implicit none
Line 03:     real(8) :: obs
Line 04:     integer :: iret

Line 05:     obs=10.15

Line 06:! encode
Line 07:     open(20,file='bufrtable.txt')
Line 08:     open(10,file='t.bufr',action='write',form='unformatted')
Line 09:     call openbf(10,'OUT',20)
Line 10:       call openmb(10, 'ADPUPA', 08120100)
Line 11:         call ufbint(10,obs,1,1,iret,'TOB')
Line 12:         call writsb(10)
Line 13:       call closmg(10)
Line 14:     call closbf(10)

Line 15:    end program
```

Step 2: Compile executable "*encodeT.exe* "

```
$ ifort -c encode_temperature.f90
$ ifort -o encodeT.exe encode_temperature.o -L../lib –lbufr_i4r8
```

Note: Need BUFR library under GSI directory ./lib/libbufr_i4r8.a
Use same compiler to compile BUFR lib and *encode_temperature.f90*

BUFR Encoding Steps (Continue)

Step 3: Prepare BUFR Table file "*bufrtable.txt*"

```
.-------------------------------------------------------------------------------.
| ------------   USER DEFINITIONS FOR TABLE-A TABLE-B TABLE D   -------------- |
|-------------------------------------------------------------------------------|
| MNEMONIC | NUMBER | DESCRIPTION                                              |
|----------|--------|----------------------------------------------------------|
| ADPUPA   | A48102 | UPPER-AIR (RAOB, PIBAL, RECCO, DROPS) REPORTS            |
| TOB      | 012245 | TEMPERATURE OBSERVATION
|          |        |                                                          |
|-------------------------------------------------------------------------------|
| MNEMONIC | SEQUENCE                                                              |
|----------|-----------------------------------------------------------------------|
| ADPUPA   | TOB                                                                   |
|          |                                                                       |
|-------------------------------------------------------------------------------|
| MNEMONIC | SCAL | REFERENCE   | BIT | UNITS                     |-------------|
|----------|------|-------------|-----|---------------------------|-------------|
| TOB      |    1 |       -2732 |  14 | DEG C                     |-------------|
`-------------------------------------------------------------------------------'
```

Step 4: Run *encodeT.exe* to generate BUFR file "*t.bufr*", BUFR Table is also written into *t.bufr*

```
$ encodeT.exe
```

BUFR Decoding Steps

Step 1: Write the Fortran Code "*decode_temperature.f90*"

```
Line 01:    program bufr_decode_temperature
Line 02:     implicit none
Line 03:     real(8) :: obs
Line 04:     character(8) subset
Line 05:     integer :: idate,iret

Line 06:! decode
Line 07:     open(10,file=' t.bufr',action='read',form='unformatted')
Line 08:     call openbf(10,'IN',10)
Line 09:       call readmg(10,subset,idate,iret)
Line 10:         call ireadsb(10,iret)
Line 11:         call ufbint(10,obs,1,10,iret,'TOB')
Line 12:         write(*,*) obs
Line 13:     call closbf(10)

Line 14:    end progr
```

Step 2: Compile executable "*decodeT.exe* "

```
$ ifort -c decode_temperature.f90
$ ifort -o decodeT.exe decode_temperature.o -L../lib –lbufr_i4r8
```

Step 3: Run *decodeT.exe* to read temperature from BUFR file "*t.bufr*"

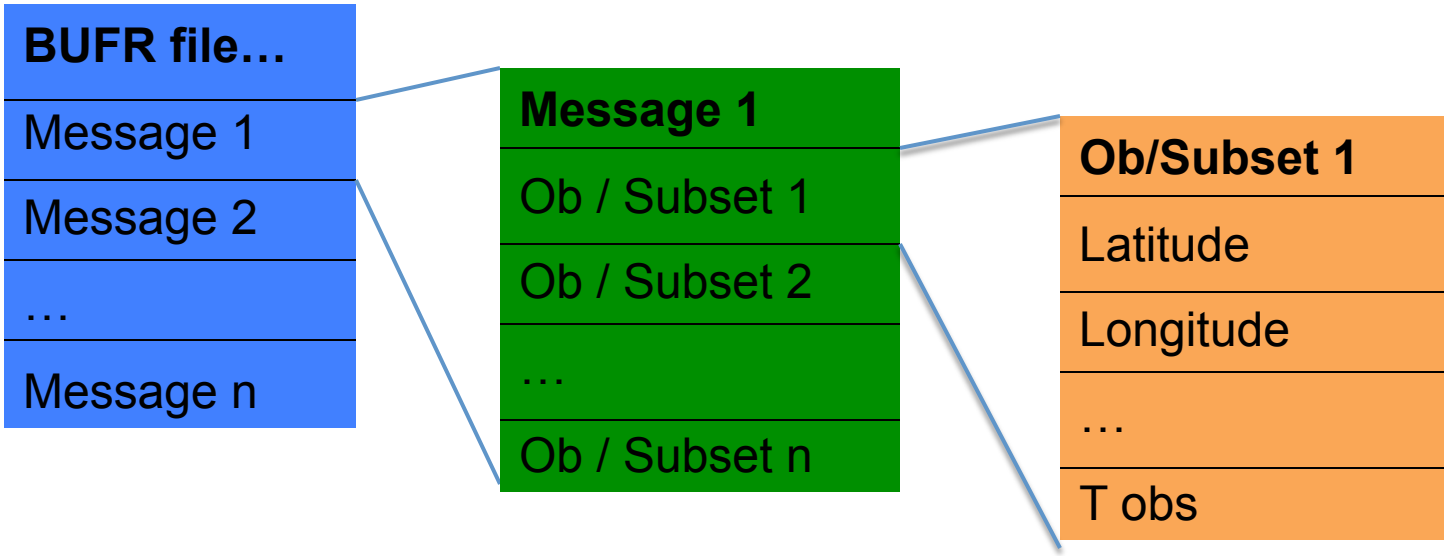
```
$ decodeT.exe
```

10.1000000000000

BUFR/PrepBUFR File Structure

BUFR/PrepBUFR file structure

A bit of terminology:
BUFR files (including “PrepBUFR” files) contain “messages”;
Each message contains “subsets”;
Each subset contains one meteorological observation report.



BUFR/PrepBUFR file structure

BUFR file example: *gdas1.t12z.prepbufr.nr*

Message 1: ADPSFC: Surface land (synoptic, metar) reports

Message 2: ADPUPA: Upper air (raob, pibal, recco, drops) reports

.

.

Message n:

Message 1	Lat	Lon	P	T	Q	U	V	Type	
Ob / Subset 1	52.1	12.5	984.4	10e10	10e10	1.7	4.7	281	wind report
Ob / Subset 2	52.1	12.5	984.4	23.1	12979.0	10e10	10e10	181	mass report
.									
.									
Ob / Subset n									

BUFR/PrepBUFR Processing Tools

- Encode, decode and append a simple BUFR file
- Encode, decode and append GSI conventional PrepBUFR files
- Decode GSI radiance BUFR files
- All tools are in GSI util/bufr_tools
- Detailed information for these tools are in BUFR user guide chapter 2.

NCEP BUFRLIB

- All tools are based on NCEP BUFRLIB
- BUFRLIB contains around 250 Fortran & C routines. No more than 10 to 20 of them are directly called by a user. The rest are used for underlying tasks.
- Excellent BUFRLIB on-line documentation from:

  http://www.nco.ncep.noaa.gov/sib/decoders/BUFRLIB/

- The BUFRLIB requires BUFR (or PrepBUFR) files to be FORTRAN-blocked before they are used by BUFRLIB
  - Almost always, any BUFR file is already blocked and you can use the BUFRLIB subroutines directly to read/write BUFR file.
  - If your BUFR file is not blocked, Use NCEP *cwordsh* utility to block it http://www.nco.ncep.noaa.gov/sib/decoders/BUFRLIB/toc/cwordsh/

Encode BUFR file

- Write the observation into a BUFR file

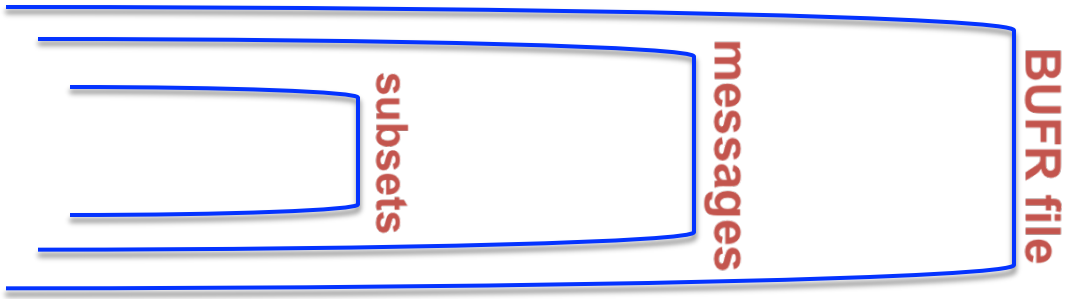
```
program bufr_encode_sample   (34 Lines)
!
!  example of writing one value into a bufr file
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write',form='unformatted')
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out,msgtype,idate)
      call ufbint(unit_out,hdr,3,1,iret,hdstr)
      call ufbint(unit_out,obs,1,1,iret,obstr)
      call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

"A BUFR file contains one or more BUFR messages, each containing one or more BUFR data subsets, each containing one or more BUFR data values

Define variables and assign obs values

Open BUFR tables file

Open BUFR file



BUFRLIB

```
program bufr_encode_sample   (34 Lines)
!  example of writing one value into a bufr file
 implicit none
 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write' ,form='unformatted')

 call datelen(10)
  call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out,msgtype,idate)
      call ufbint(unit_out,hdr,3,1,iret,hdstr)
      call ufbint(unit_out,obs,1,1,iret,obstr)
      call writsb(unit_out)
   call closmg(unit_out)
  call closbf(unit_out)

 end program
```

openbf, closbf, openmb, closmg, ufbint, writsb, datelen are the BUFRLIB subroutines and functions. They are very often used to read/write BUFR file. Understanding usage of them will be very helpful in users own application.

DX BUFR table

```
program bufr_encode_sample   (34 Lines)
!
!  example of writing one value into a bufr file
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write' ,form='unforma
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)
      call ufbint(unit_out,hdr,3,1,iret,hdstr)
      call ufbint(unit_out,obs,1,1,iret,obstr)
      call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

Fortran 'open' command to open the BUFR tables file.
(it is under GSI directory ./util/bufr_tools)

BUFR table defines the content and form for each of message types. It is embedded within the first few BUFR messages of the file itself.

```
program bufr_encode_sample   (34 Lines)
!
!  example of writing one value into a bufr file
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports
! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write' ,form='unformatted')
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)
      call ufbint(unit_out,hdr,3,1,iret,hdstr)
      call ufbint(unit_out,obs,1,1,iret,obstr)
      call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

Define mnemonics

DX BUFR table includes Table A, B, D mnemonic.
A mnemonic is a descriptive, alphanumeric name for an data value.

Table A mnemonic: refer to report types.
ADPUPA: UPPER-AIR (RAOB, PIBAL, RECCO, DROPS) REPORTS

Table B mnemonics: refer to basic data values.
XOB: Longtitude
YOB: Latitude
DHR: obs time – cycle time
TOB: temperature

Setup data

Data written to subset.

Array hdr/obstr is associated to mnemonics in string hdstr/obs.

BUFR file

```
program bufr_encode_sample   (34 Lines)
!
!  example of writing one value into a bufr file
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports
! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write' ,form='unformatted')
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)
     call ufbint(unit_out,hdr,3,1,iret,hdstr)
     call ufbint(unit_out,obs,1,1,iret,obstr)
     call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

Fortran 'open' command to open an unformatted binary file for write.

***OPENBF* (LUBFR, CIO, LUNDX)**
Purpose: Identifies to the BUFRLIB a BUFR file of logical unit *LUBFR.*
Input arguments:
LUBFR=INTEGER: Logical unit for BUFR file
LUNDX=INTEGER: Logical unit for BUFR tables
CIO = 'IN' or 'OUT' or 'APN'
'IN': reading/decoding BUFR
'OUT': writing/encoding BUFR
'APN': append to existing BUFR

***CLOSBF* (LUBFR)**
Purpose: severs the connection between logical unit *LUBFR* and the BUFRLIB.
Input arguments:
LUBFR=INTEGER: Logical unit for BUFR file

Note: ***CLOSBF*** actually executes a FORTRAN "CLOSE" on logical unit *LUBFR* before returning.

Set date

```
program bufr_encode_sample   (34 Lines)
!
!  example of writing one value into a bufr file
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write',form='unformatted')

 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)
     call ufbint(unit_out,hdr,3,1,iret,hdstr)
     call ufbint(unit_out,obs,1,1,iret,obstr)
     call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

***DATELEN* (LEN)**

Purpose: specify the format *IDATE.*

Input arguments:
LEN INTEGER
Length of Section 1 date-time values
8 = YYMMDDHH (2-digit year)
10 = YYYYMMDDHH (4-digit year)

Message

```
program bufr_encode_sample   (34 Lines)
!
!  example of writing one value into a bufr file
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write',form='unformatted')
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)
      call ufbint(unit_out,hdr,3,1,iret,hdstr)
     call ufbint(unit_out,obs,1,1,iret,obstr)
     call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

***OPENMB* (LUBFR, CSUBSET, IDATE)**

Purpose: Open and initialize, within internal arrays, a new BUFR message for eventual output to *LUBFR*, using *CSUBSET* as message type, *IDATE* as date.

Input arguments:

LUBFR=INTEGER: Logical unit for BUFR file

CSUBSET=CHAR*(*): Table A mnemonic for type of BUFR message to be opened

IDATE=INTEGER: Date-time to be stored within Section 1 of BUFR message

***CLOSMG* (LUBFR)**

Purpose: Close existing internal BUFR message (if any) and write it to output.

Input arguments:

LUBFR=INTEGER: Logical unit for BUFR file.

Data subsets

```
program bufr_encode_sample   (34 Lines)
!
!  example of writing one value into a bufr file
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write',form='unformatted')
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)

   call ufbint(unit_out,hdr,3,1,iret,hdstr)
   call ufbint(unit_out,obs,1,1,iret,obstr)
   call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

***UFBINT* (LUBFR, R8ARR, MXMN, MXLV, iret, CMNSTR)**

Purpose: writes or reads specified values to or from the current BUFR data subset within the internal arrays.

Input arguments:
CMNSTR=CHAR*(*): String of blank-separated mnemonics associated with R8ARR
MXMN INTEGER Size of first dimension of R8ARR
MXLV INTEGER Size of second dimension of R8ARR OR number of levels of data values to be written to data subset

Input or output argument:
R8ARR(MXMN, MXLV)=REAL*8
Data values written/read to/from data subset

Output argument:
iret = INTEGER Number of levels of data values written to data subset

hdstr= XOB	YOB	DHR	obstr= TOB
hdr = (1)75.	(2)30.	(3)-0.1	obs(1)= 17.15

Data subsets

```
program bufr_encode_sample   (34 Lines)
!  example of writing one value into a bufr file
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write',form='unformatted')
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)
      call ufbint(unit_out,hdr,3,1,iret,hdstr)
      call ufbint(unit_out,obs,1,1,iret,obstr)
      call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

***WRITSB* (LUBFR)**

Purpose: Indicates to BUFRLIB that the subset is ready to be encoded into the current message for the BUFR file.

Input argument:
LUBFR=INTEGER; Logical unit for BUFR file

Mnemonics and data array

```
program bufr_encode_sample   (34 Lines)
!
!  example of writing one value into a bufr file
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write',form='u
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)
   call ufbint(unit_out,hdr,3,1,iret,hdstr)
   call ufbint(unit_out,obs,1,1,iret,obstr)
   call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

hdstr= XOB YOB DHR obstr= TOB

hdr = (1)75. (2)30. (3)-0.1 obs(1)= 17.15

Write to bufr file sampe.bufr

Section 3	Section 4
XOB YOB DHR TOB	75. 30. -0.1 17.1

Decode BUFR file

- Read the observation from the BUFR file

```
program bufr_decode_sample
!
! example of reading observations from bufr
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,10)

 integer :: ireadmg,ireadsb
 character(8) msgtype
 integer :: unit_in=10
 integer :: idate,iret,num_message,num_subset

! decode
 open(unit_in,file='sample.bufr',action='read',form='unformatted')
 call openbf(unit_in,'IN',unit_in)
 call datelen(10)
  num_message=0
  msg_report: do while (ireadmg(unit_in,msgtype,idate) == 0)
   num_message=num_message+1
   num_subset = 0
   write(*,'(I10,I4,a10)') idate,num_message,msgtype
   sb_report: do while (ireadsb(unit_in) == 0)
    num_subset = num_subset+1
     call ufbint(unit_in,hdr,3,1 ,iret,hdstr)
     call ufbint(unit_in,obs,1,10,iret,obstr)
    write(*,'(2I5,4f8.1)') num_subset,iret,hdr,obs(1,1)
   enddo sb_report
  enddo msg_report
 call closbf(unit_in)

end program
```

"A BUFR file contains one or more BUFR messages, each containing one or more BUFR data subsets, each containing one or more BUFR data values

subsets

messages

BUFR file

Encode

```
program bufr_encode_sample
!
!  example of writing one value into a bufr
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write' &
            ,form='unformatted')
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
  call openmb(unit_out,msgtype,idate)
    call ufbint(unit_out,hdr,3,1,iret,hdstr)
    call ufbint(unit_out,obs,1,1,iret,obstr)
      call writsb(unit_out)
  call closmg(unit_out)
 call closbf(unit_out)

end program
```

Decode

```
rogram bufr_decode_sample
!
! example of reading observations from bufr
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,10)

 integer :: ireadmg,ireadsb
 character(8) msgtype
 integer :: unit_in=10
 integer :: idate,iret,num_message,num_subset

! decode
 open(unit_in,file='sample.bufr',action='read',form='unformatted')
 call openbf(unit_in,'IN',unit_in)
 call datelen(10)
   num_message=0
 msg_report: do while (ireadmg(unit_in,msgtype,idate)
== 0)
    num_message=num_message+1
    num_subset = 0
    write(*,'(I10,I4,a10)') idate,num_message,msgtype
   sb_report: do while (ireadsb(unit_in) == 0)
     num_subset = num_subset+1
     call ufbint(unit_in,hdr,3,1 ,iret,hdstr)
     call ufbint(unit_in,obs,1,10,iret,obstr)
     write(*,'(2I5,4f8.1)') num_subset,iret,hdr,obs(1,1)
    enddo sb_report
   enddo msg_report
 call closbf(unit_in)

end program
```

```
program bufr_decode_sample
!
! example of reading observations from bufr
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,10)

 integer :: ireadmg,ireadsb
 character(8) msgtype
 integer :: unit_in=10
 integer :: idate,iret,num_message,num_subset

! decode
 open(unit_in,file='sample.bufr',action='read',form='unformatted')
 call openbf(unit_in,'IN',unit_in)
 call datelen(10)
   num_message=0
msg_report: do while (ireadmg(unit_in,msgtype,idate) == 0)
     num_message=num_message+1
     num_subset = 0
     write(*,'(I10,I4,a10)') idate,num_message,msgtype
     sb_report: do while (ireadsb(unit_in) == 0)
       num_subset = num_subset+1
       call ufbint(unit_in,hdr,3,1 ,iret,hdstr)
       call ufbint(unit_in,obs,1,10,iret,obstr)
       write(*,'(2I5,4f8.1)') num_subset,iret,hdr,obs(1,1)
     enddo sb_report
   enddo msg_report
 call closbf(unit_in)

end program
```

Message

IRET = *IREADMG* (LUBFR, CSUBSET, IDATE)

Purpose: reads the next BUFR message from the given BUFR file pointed to by *LUBFR*.

Input argument:

LUBFR=INTEGER: Logical unit for BUFR file

Output arguments:

CSUBSET=CHAR*(*): Table A mnemonic for BUFR message

IDATE=INTEGER: Section 1 date-time for BUFR messsage

IRET=INTEGER: Return code:

0 = normal return

-1 = no more BUFR messages in BUFR file

Data Subset

```
program bufr_decode_sample
!
! example of reading observations from bufr
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,10)

 integer :: ireadmg,ireadsb
 character(8) msgtype
 integer :: unit_in=10
 integer :: idate,iret,num_message,num_subset

! decode
 open(unit_in,file='sample.bufr',action='read',form='unformatted')
 call openbf(unit_in,'IN',unit_in)
 call datelen(10)
  num_message=0
  msg_report: do while (ireadmg(unit_in,msgtype,idate) == 0)
    num_message=num_message+1
    num_subset = 0
    write(*,'(I10,I4,a10)') idate,num_message,msgtype
    sb_report: do while (ireadsb(unit_in) == 0)
      num_subset = num_subset+1
        call ufbint(unit_in,hdr,3,1 ,iret,hdstr)
        call ufbint(unit_in,obs,1,10,iret,obstr)
      write(*,'(2I5,4f8.1)') num_subset,iret,hdr,obs(1,1)
    enddo sb_report
  enddo msg_report
 call closbf(unit_in)

end program
```

IRET = *IREADSB* (LUBFR)

Purpose: reads a subset from that internal message.

Input argument:
LUBFR=INTEGER: Logical unit for BUFR file

Output arguments:
IRET INTEGER Return code:
0 = normal return
-1 = no more BUFR data subsets in current BUFR message

***UFBINT* (LUBFR, R8ARR, MXMN, MXLV, iret, CMNSTR)**
Write subset to hdr, obs array.

Note:
Decode BUFR file: MXLV contains the actual second dimension of *R8ARR*.
Encode BUFR File: MXLV contains the actual number of levels of data values that are to be written to the data subset.

Mnemonics and data array

```
program bufr_decode_sample
!
! example of reading observations from bufr
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,10)

 integer :: ireadmg,ireadsb
 character(8) msgtype
 integer :: unit_in=10
 integer :: idate,iret,num_message,num_subset

! decode
 open(unit_in,file='sample.bufr',action='read',form='unformatted')
 call openbf(unit_in,'IN',unit_in)
 call datelen(10)
  num_message=0
  msg_report: do while (ireadmg(unit_in,msgtype,idate) == 0)
    num_message=num_message+1
    num_subset = 0
    write(*,'(I10,I4,a10)') idate,num_message,msgtype
    sb_report: do while (ireadsb(unit_in) == 0)
      num_subset = num_subset+1
        call ufbint(unit_in,hdr,3,1 ,iret,hdstr)
        call ufbint(unit_in,obs,1,10,iret,obstr)
      write(*,'(2I5,4f8.1)') num_subset,iret,hdr,obs(1,1)
    enddo sb_report
  enddo msg_report
 call closbf(unit_in)

end program
```

hdstr= XOB YOB DHR obstr= TOB

Section 3	Section 4
XOB YOB DHR TOB	75. 30. -0.1 17.1

read from bufr file sampe.bufr

hdr = (1)75. (2)30. (3)-0.1 obs(1)= 17.1

Append to BUFR file

- Append the observation to an existing BUFR file

Encode

```
program bufr_encode_sample
!
!  example of writing one value into a bufr file
!
 implicit none

 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=75.;hdr(2)=30.;hdr(3)=-0.1
 obs(1,1)=17.15
 idate=2008120100  ! YYYYMMDDHH
 msgtype='ADPUPA'   ! upper-air reports

! encode
 open(unit_table,file='prepobs_prep.bufrtable')
 open(unit_out,file='sample.bufr',action='write' &
            ,form='unformatted')
 call datelen(10)
 call openbf(unit_out,'OUT',unit_table)
   call openmb(unit_out, msgtype,idate)
      call ufbint(unit_out,hdr,3,1,iret,hdstr)
      call ufbint(unit_out,obs,1,1,iret,obstr)
      call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)

end program
```

Append

```
program bufr_append_sample

!sample of appending one observation into bufr file
 implicit none
 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=85.0;hdr(2)=50.0;hdr(3)=0.2
 obs(1,1)=15.0
 idate=2008120101  ! YYYYMMDDHH
 msgtype='ADPSFC'   ! surface land reports

! get bufr table from existing bufr file
 open(unit_table,file='prepobs_prep_app.bufrtable')
 open(unit_out,file='sample.bufr',status='old',form='unformatted')
 call openbf(unit_out,'IN',unit_out)
 call dxdump(unit_out,unit_table)
 call closbf(unit_out)

! append
 open(unit_out,file='sample.bufr',status='old',form='unformatted')
 call datelen(10)
 call openbf(unit_out,'APN',unit_table)
   call openmb(unit_out, msgtype,idate)
      call ufbint(unit_out,hdr,3,1,iret,hdstr)
      call ufbint(unit_out,obs,1,1,iret,obstr)
      call writsb(unit_out)
   call closmg(unit_out)
 call closbf(unit_out)
end program
```

BUFR table

```
program bufr_append_sample
!
! sample of appending one observation into bufr file
 implicit none
 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=85.0;hdr(2)=50.0;hdr(3)=0.2
 obs(1,1)=15.0
 idate=2008120101  ! YYYYMMDDHH
 msgtype='ADPSFC'   ! surface land reports

! get bufr table from existing bufr file
 open(unit_table,file='prepobs_prep_app.bufrtable')
 open(unit_out,file='sample.bufr',status='old',form='unformatted')
 call openbf(unit_out,'IN',unit_out)
 call dxdump(unit_out,unit_table)
 call closbf(unit_out)

! append
 open(unit_out,file='sample.bufr',status='old',form='unformatted')
 call datelen(10)
 call openbf(unit_out,'APN',unit_table)
  call openmb(unit_out, msgtype,idate)
    call ufbint(unit_out,hdr,3,1,iret,hdstr)
    call ufbint(unit_out,obs,1,1,iret,obstr)
    call writsb(unit_out)
  call closmg(unit_out)
 call closbf(unit_out)
end program
```

Appending requires the report structure (BUFR table) of the new data subset fits the report structure in the existing file. So we use the following subroutine to retrieve BUFR table from the existing BUFR file:

***DXDUMP* (LUBFR, LDXOT)**

Purpose: reads the embedded tables information in the BUFR file and write it out to the ASCII format file.

Input arguments:

LUBFR: INTEGER Logical unit for BUFR file

LDXOT:INTEGER Logical unit for output BUFR tables file

```
program bufr_append_sample
! sample of appending one observation into bufr file
 implicit none
 character(80):: hdstr='XOB YOB DHR'
 character(80):: obstr='TOB'
 real(8) :: hdr(3),obs(1,1)

 character(8) msgtype
 integer :: unit_out=10,unit_table=20
 integer :: idate,iret

! set data values
 hdr(1)=85.0;hdr(2)=50.0;hdr(3)=0.2
 obs(1,1)=15.0
 idate=2008120101  ! YYYYMMDDHH
 msgtype='ADPSFC'   ! surface land reports

! get bufr table from existing bufr file
 open(unit_table,file='prepobs_prep_app.bufrtable')
 open(unit_out,file='sample.bufr',status='old',form='unformatted')
 call openbf(unit_out,'IN',unit_out)
 call dxdump(unit_out,unit_table)
 call closbf(unit_out)

! append
 open(unit_out,file='sample.bufr',status='old',form='unformatted')
 call datelen(10)
 call openbf(unit_out,'APN',unit_table)
 call openmb(unit_out, msgtype,idate)
    call ufbint(unit_out,hdr,3,1,iret,hdstr)
    call ufbint(unit_out,obs,1,1,iret,obstr)
    call writsb(unit_out)
 call closmg(unit_out)
 call closbf(unit_out)
end program
```

Write the new data subset to the existing BUFR file.

Test results (Basic Practice case 0):

./bufr_encode_sample.exe

This generates a new bufr file *sample.bufr*

./bufr_decode_sample.exe

This reads one observation from *sample.bufr*, and write result on screen:

```
2008120100    1  ADPUPA
    1     1     75.0     30.0     -0.1    17.1
```

./bufr_append_sample.exe

Now, append a new observation to *sample.bufr*.

./decode_sample.exe

Read *sample.bufr* and show two observations in it:

```
2008120100    1  ADPUPA
    1     1     75.0     30.0     -0.1    17.1
2008120101    2  ADPSFC
    1     1     85.0     50.0      0.2    30.0
```

Examples for GSI BUFR/PrepBUFR files

Code name	Illustrated process function
prepbufr_decode_all.f90	! read BUFR table from an existing prepbufr file ! read all observation information used by GSI analysis from an existing prepbufr file.
prepbufr_encode_surface.f90	! write a surface observation into a new prepbufr file
prepbufr_encode_upperair.f90	! write a upper air observation into a new prepbufr file
prepbufr_append_upperair.f90	! read BUFR table from an existing prepbufr file ! append a upper air observation into an existing prepbufr file
prepbufr_append_surface.f90	! read BUFR table from an existing prepbufr file ! append a surface observation into an existing prepbufr file.
prepbufr_append_retrieve.f90	! read BUFR table from an existing prepbufr file ! append retrieved data into an existing prepbufr file.
bufr_decode_radiance.f90	! read BUFR table from an existing radiance bufr file ! real radiance data from an existing radiance bufr file.

- These files have the same structure and call the same BUFRLIB subroutines/functions as those three simple examples
- The only difference is the mnemonic lists used in these files are much longer

MNEMONIC in prepbufr_decode_all.f90

In GSI, *read_prepbufr.f90* reads PrepBUFR file. The following mnemonic lists come from *read_prepbufr.f90* and are used in PrepBUFR sample code.

```
integer, parameter :: mxmn=35, mxlv=250
character(80):: hdstr='SID XOB YOB DHR TYP ELV SAID T29'
character(80):: obstr='POB QOB TOB ZOB UOB VOB PWO CAT PRSS'
character(80):: qcstr='PQM QQM TQM ZQM WQM NUL PWQ     '
character(80):: oestr='POE QOE TOE NUL WOE NUL PWE     '
real(8) :: hdr(mxmn),obs(mxmn,mxlv),qcf(mxmn,mxlv),oer(mxmn,mxlv)
```

```
call ufbint(unit_in,hdr,mxmn,1   ,iret,hdstr)
call ufbint(unit_in,obs,mxmn,mxlv,iret,obstr)
call ufbint(unit_in,oer,mxmn,mxlv,iret,oestr)
call ufbint(unit_in,qcf,mxmn,mxlv,iret,qcstr)
```

Questions?

gsi_help@ucar.edu