bufr_filter Practicals II

More complex BUFR messages

Roberto Ribas

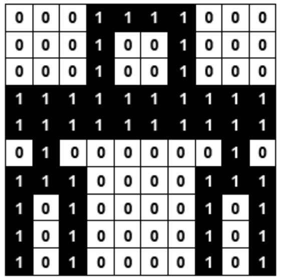
Forecast Department

Copernicus Production Section

Outline

In this session, we will see some examples of

- Data replication



- Quality information

- Data replication and bit maps

Replication

Replication is used in BUFR to repeat a sequence of descriptors a certain number of times(called replication factor). We may have two different types of replications:

- **Simple replication**, where the number of replications is included as a part of the descriptor sequence in Section 3.This may be used to encode series of parameters a fixed number of times in all reports. (for example blocks of location information)
- **Delayed replication**, where the number of replications must be found in the data section (Section 4). This may be useful for TEMP messages, where the number of levels is not know in advance.

Simple Replication

In this case, the number of replications is fixed and is part of the descriptors sequence. For example

1 04 002

0 08 002 0 20 011 0 20 012 0 20 013

In this case, 1 04 002 is a replication descriptor as starts with F=1. The 04 means take 4 following descriptors and repeat them 002 times.

The 0 08 002 is Table B Class 08 which means Significance qualifiers, in particular vertical significance.

So after expansion, we end up with:

0 08 002 0 20 011 0 20 012 0 20 013

0 08 002 0 20 011 0 20 012 0 20 013

Simple replication example

```
################################
# simple replication example
################################
set compressedData=0;
set unexpandedDescriptors={104002,1006,1008,2061,2062};
set #1#aircraftFlightNumber="BA486";
set #2#aircraftFlightNumber="BA485";
set #1#aircraftRegistrationNumberOrOtherIdentification="N12345";
set #2#aircraftRegistrationNumberOrOtherIdentification="N12345";
set #1#aircraftNavigationalSystem=0;
set #2#aircraftNavigationalSystem=0;
set #1#aircraftDataRelaySystemType=3;
set #2#aircraftDataRelaySystemType=3;

set pack=1;

write;
```

Simple replication example

```
104002,1006,1008,2061,2062
```

This is a simple replication descriptor,104002, which means repeat the 4 following descriptors two times (for the two legs of the flight), this number of repetitions is fixed and ”lives” in section 3 of the BUFR file.

Then we end up with

```
1006,1008,2061,2062
```

```
1006,1008,2061,2062
```

Delayed replication

In this case, the ***number of replications*** is **not** found in ***Section 3***, with the rest of the descriptors. We find the number of replications in ***Section 4***, in the Data Section.

For example the following sequence:

1 04 000	0 31 001	0 08 002 0 20 011 0 20 012 0 20 013

Now the replication descriptor 1 04 000 looks weird(000 replications??).

This indicates that is a ***delayed replication***, so we have to look at the next descriptor 0 31 001. This descriptor indicates that we have to read 8 bits from the data Section to retrieve the number of repetitions(in this case 8 bits, allowing up to 2^8-1=255 repetitions) but it may be 16 bits for an extended replication (up to 65535 replications).

Delayed replication

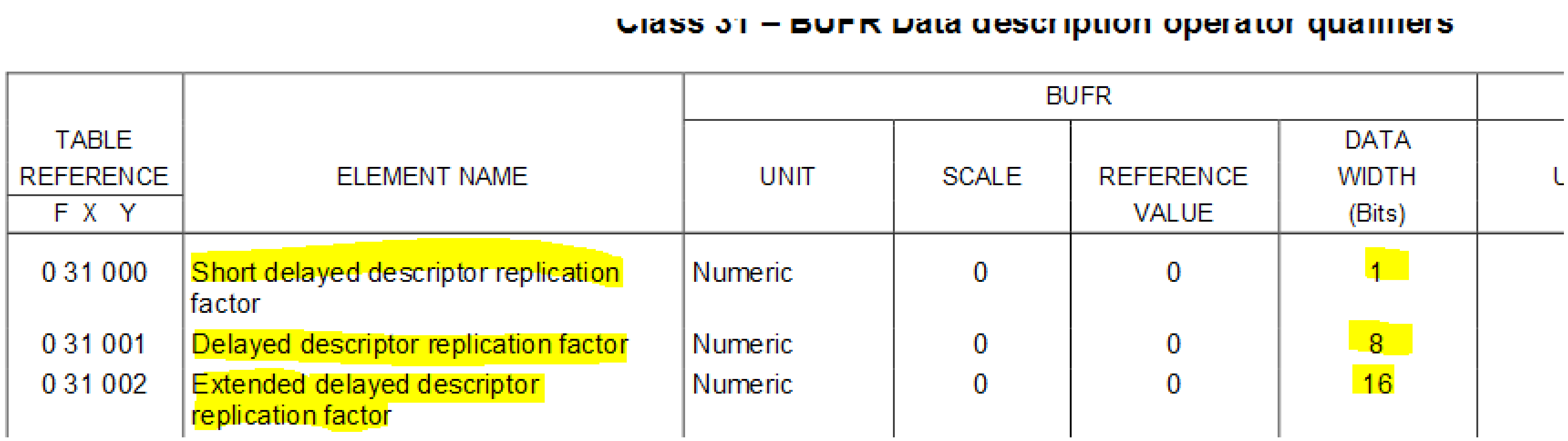
If we want to create a new message with a given number of delayed replication factors, some of the following **VECTOR** keys must be set with the required number of elements.

- **inputDelayedDescriptorReplicationFactor**
- **inputExtendedDelayedDescriptorReplicationFactor**
- **inputShortDelayedDescriptorReplicationFactor**

These keys must be set before setting unexpandedDescriptors. This is due to the fact that unexpandedDescriptors sets the Section 3 (structure of the BUFR file) and fills section 4 accordingly with MISSING_VALUES. The replication factors are set to 1 in the created BUFR message as well.

Delayed replication

- The different keys represent different number of bits to be read from the data section in order to know the number of repetitions.



Class 31 – BUFR Data description operator qualifiers

TABLE REFERENCE	ELEMENT NAME	BUFR			
F X Y		UNIT	SCALE	REFERENCE VALUE	DATA WIDTH (Bits)
0 31 000	Short delayed descriptor replication factor	Numeric	0	0	1
0 31 001	Delayed descriptor replication factor	Numeric	0	0	8
0 31 002	Extended delayed descriptor replication factor	Numeric	0	0	16

Delayed replication

When using **delayed replication** two possible cases arise:

- **Uncompressed data**, set the delayed descriptors keys to the appropriate number of values/repetitions. ***Different subsets may have different replications***.

- **Compressed data**. In this case, the replication factor **must be constant for all subsets**.

Creating a **uncompressed** BUFR message

UNCOMPRESSED message with delayed replication

Creating a **uncompressed** BUFR message

- We can create a new BUFR message with the descriptor 309052.

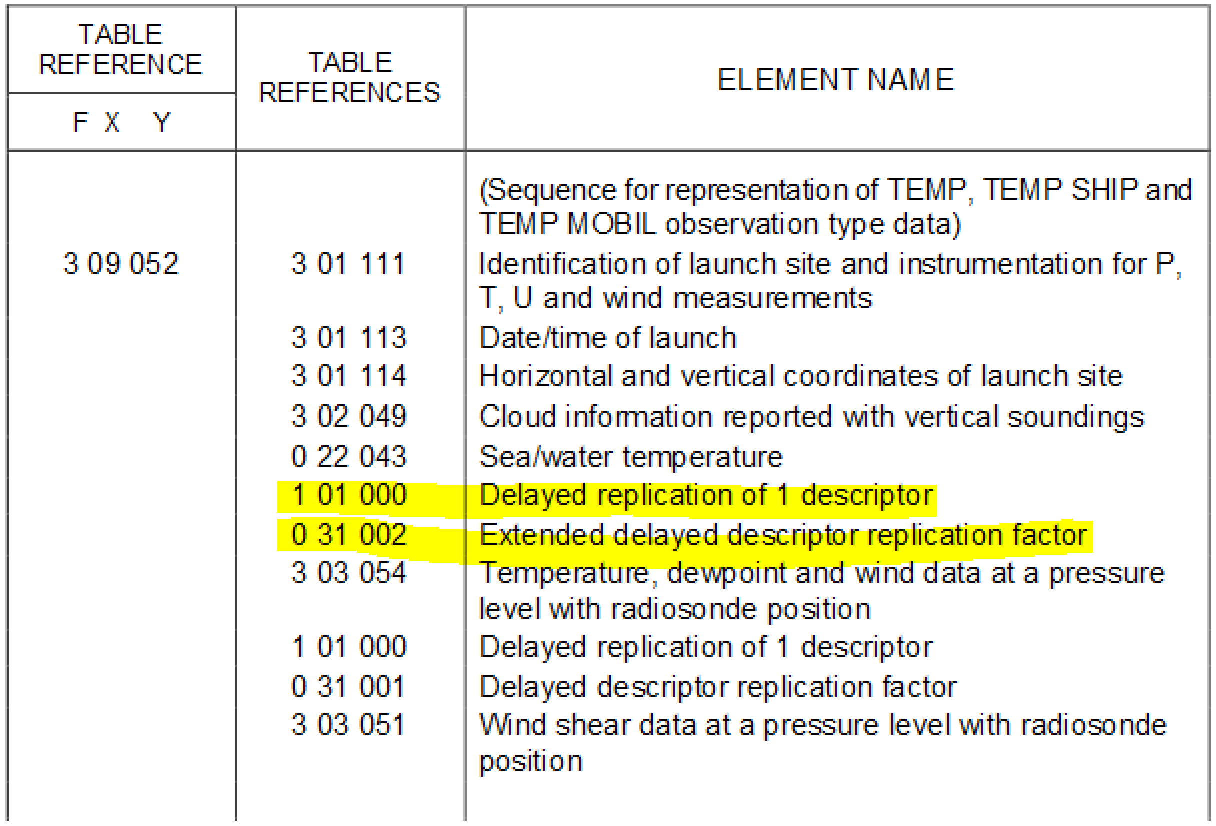


TABLE REFERENCE	TABLE REFERENCES	ELEMENT NAME
F X Y		
		(Sequence for representation of TEMP, TEMP SHIP and TEMP MOBIL observation type data)
3 09 052	3 01 111	Identification of launch site and instrumentation for P, T, U and wind measurements
	3 01 113	Date/time of launch
	3 01 114	Horizontal and vertical coordinates of launch site
	3 02 049	Cloud information reported with vertical soundings
	0 22 043	Sea/water temperature
	1 01 000	Delayed replication of 1 descriptor
	0 31 002	Extended delayed descriptor replication factor
	3 03 054	Temperature, dewpoint and wind data at a pressure level with radiosonde position
	1 01 000	Delayed replication of 1 descriptor
	0 31 001	Delayed descriptor replication factor
	3 03 051	Wind shear data at a pressure level with radiosonde position

Create an uncompressed+replication

In this case, the number of repetitions may change from subset to subset.

Start creating a BUFR message by setting the unexpandedDescriptors to 309052

```
set compressedData=0;
set unexpandedDescriptors={309052};
write;
```

To run this filter

```
bufr_filter –o out.b  theFilter.flt synop.bufr4
```

Create an uncompressed+replication

If we do a

```
bufr_dump –p out.b
```

we see

```
inputDelayedDescriptorReplicationFactor= {1}
inputExtendedDelayedDescriptorReplicationFactor= {1}
```

So when the message is created, the replication factors are set to 1, and the data section is filled with MISSING values.

Create an uncompressed+replication

- If we print the **expandedDescriptors** and the **expandedAbbreviations** the replication stands out.

```
print "[expandedDescriptors!1] ";
print;
print "[expandedAbbreviations!1]";
```

Uncompressed+replication

The extended and the delayed replication factors appear as numbers as well.

110000 --- we decide to replicate 4 times this part

extendedDelayedDescriptorReplicationFactor
timePeriod
extendedVerticalSoundingSignificance
→**pressure**
nonCoordinateGeopotentialHeight
latitudeDisplacement
longitudeDisplacement
airTemperature
dewpointTemperature
→**windDirection**
→**windSpeed**

Appear only here

Appears twice

107000 --- we decide to replicate 2 times this part

delayedDescriptorReplicationFactor
timePeriod
extendedVerticalSoundingSignificance
→**pressure**
latitudeDisplacement
longitudeDisplacement
absoluteWindShearIn1KmLayerBelow
absoluteWindShearIn1KmLayerAbove

Uncompressed +replication

- Now we set the number of subsets to 1 and delayed Replication keys.

```
set numberOfSubsets=1;
set inputDelayedDescriptorReplicationFactor={2};
set inputExtendedDelayedDescriptorReplicationFactor={4};
set compressedData=0;
set unexpandedDescriptors={309052};
write;
```

- At this point we have one subset and the delayed descriptor replication factors. **It is critical to set these keys before setting the unexpandedDescriptors**.

- No **`set pack=1`** is needed here as we are not setting values in the data section, so far we are only working on section 3(message structure).

Uncompressed+replication

- So with one subset we have to provide the appropriate number of values to match the replications

```
set numberOfSubsets=1; # numberOfSubsets is 1

set inputDelayedDescriptorReplicationFactor={2};
set inputExtendedDelayedDescriptorReplicationFactor={4};
set compressedData=0;
set unexpandedDescriptors={309052};
# now we have to provide 2+4=6 values to match the
# replications delayed+extended
# for pressure ( appears in both replications)

set pressure={102400,50000,60000,70000,89000,80000};

# and 4 values for windDirection as it appears only
# in the extendedReplication
set windDirection={211,212,213,214};

set pack=1;
write;
```

Uncompressed +replication

If we use **`bufr_dump –p`** to see the output file

```
bufr_dump -p out.b |grep "windDirection"
#1#windDirection=211
#2#windDirection=212
#3#windDirection=213
#4#windDirection=214
```

```
bufr_dump -p out.b |grep "pressure"
#1#pressure=102400
#2#pressure=50000
#3#pressure=60000
#4#pressure=7000
#5#pressure=89000
#6#pressure=80000
```

Uncompressed +replication

Now we can add more subsets and more data values (remember they have to match with the number of replications and extended replications).

```
set numberOfSubsets=2; # numberOfSubsets is 2

set inputDelayedDescriptorReplicationFactor={2,4};
set inputExtendedDelayedDescriptorReplicationFactor={3,5};
set compressedData=0;
set unexpandedDescriptors={309052};
# now we have to provide 2+3=5 values for the first subsets
# and 4+5=9 values for the second subset  to match the replications
# for pressure ( appears in both replications)
set pressure={102400,50000,60000,70000,
                        89000,80000,102300,53000,
                        54000,61000,68000,72000,75000,85000};
#and 3+5=8 values for windDirection as it appears only
# in the extendedReplication
set windDirection={211,212,213,214,301,302,250,260};
set pack=1;
write;
```

Creating an uncompressed BUFR message

Exercise 11. After seeing the BUFR table D for 309052.

Create a new BUFR uncompressed message with 2 subsets

1. Set the number of subsets to 2.
2. Set the keys:

inputDelayedDescriptorReplicationFactor
inputExtendedDelayedDescriptorReplicationFactor

for the first subset: set 3 delayed replications and 2 extended delayed replications. For the second subset: set 4 delayed replications and 3 extended delayed replications

3. Set unexpandedDescriptors=309052. Set some values for the key **pressure**. How many values do we have to set? They have to match the information of step 2.
4. Check your output BUFR file with **bufr_dump** and **bufr_dump –p** .

Solution ex 11

```
###############################################
#  creates a temp message with replications
#############################################
set numberOfSubsets=2;
# we must set the delayed replications first
set inputDelayedDescriptorReplicationFactor={3,4};
set inputExtendedDelayedDescriptorReplicationFactor={2,3};
# after setting the replication we can set the unexpandedDescriptors
set compressedData=0;
set unexpandedDescriptors={309052};

# now we can see the replications for subset #1 and #2
print "/subsetNumber=1/delayedDescriptorReplicationFactor=
       [/subsetNumber=1/delayedDescriptorReplicationFactor]";
print "/subsetNumber=1/extendedDelayedDescriptorReplicationFactor=
       [/subsetNumber=1/extendedDelayedDescriptorReplicationFactor]";
print "/subsetNumber=2/delayedDescriptorReplicationFactor=
       [/subsetNumber=2/delayedDescriptorReplicationFactor]";
print "/subsetNumber=2/extendedDelayedDescriptorReplicationFactor=
       [/subsetNumber=2/extendedDelayedDescriptorReplicationFactor]";

# at this point we have the proper replications and the template
# the number of elements of the pressure array must be the same
# as the total number of replications 3+2 for the first subset
# and 4+3 for the second totalling 12 elements
set
pressure={102400,50000,40000,30000,20000,15000,102400,50000,40000,30000,20000,15000};
set pack=1;
print "pressure=[pressure!12',']";
write;
```

Create a compressed BUFR message

Create a compressed BUFR message

Create a compressed BUFR message

- In this case, the replication factor must be **constant** for the different subsets.

		(ASCAT level 1b and level 2 data)
3 12 061	3 12 058	ASCAT level 1b data
	3 12 060	Scatterometer soil moisture data
	3 12 059	Scatterometer wind data

		(Scatterometer wind data)
3 12 059	3 12 056	Scatterometer wind cell information
	1 01 000	Delayed replication of 1 descriptor
	0 31 001	Delayed descriptor replication factor
	3 12 057	Ambiguous wind data

Create a compressed BUFR message

We can create a compressed BUFR message by using this filter.

```
set compressedData=1;
set unexpandedDescriptors={312061};
write;
```

All the ***replication factors are 1*** in this case, as the message starts brand new. If we want to change the number of delayed replications, we need to set them *before* setting the **unexpandedDescriptors** key.

The number of delayed replications **has to be the same for all subsets.**

Create a compressed BUFR message

We can show the **expandedDescriptors**, **expandedNames** and **expandedAbbreviations**.

```
set unpack=1;# needed to see the delayedDescriptors
print "unexpandedDescriptors=[unexpandedDescriptors!1]";
print "delayedDescriptorReplicationFactor=[delayedDescriptorReplicationFactor!1]";
print "expandedDescriptors=[expandedDescriptors!1]";
print "expandedNames=[expandedNames!1]";
print "expandedAbbreviations=[expandedAbbreviations!1]";
```

Create a compressed BUFR message

We see the delayed replication descriptor that repeats the following sequence. This sequence contains both **windSpeedAt10M** and **windDirectionAt10M**. Then we should provide enough values to fill these keys according with the ***numberOfSubsets*** and the number of replications we want.

We want numberOfSubsets=2 and

```
104000 == we will set this to 5
        delayedDescriptorReplicationFactor
        windSpeedAt10M
        windDirectionAt10M
        backscatterDistance
        likelihoodComputedForSolution
```

Create a compressed BUFR message

Exercise 12. By looking at the Table D descriptor 312061 we can create a compressed BUFR message.

Create a new compressed BUFR message with **unexpandedDescriptors** =312061 with 5 delayed descriptor Replications. Don't forget to set the proper **delayedDescriptors** key before setting the **unexpandedDescriptors**.

Fill the corresponding **windSpeedAt10M**, how many windSpeedAt10M keys do we have? how many values should be set for these keys?

Compare your result with scatterometer.bufr.

Solution ex 12

```
############################################
# file create_compressed_scatterometer.filter
# creates a scatterometer message and sets some data
############################################
set inputDelayedDescriptorReplicationFactor=5;
#
set compressedData=1;

set numberOfSubsets=2;

set unexpandedDescriptors={312061};
# 5 replications
set #1#windSpeedAt10M={10,20};# 2 subsets
set #2#windSpeedAt10M={30,40};
set #3#windSpeedAt10M={50,60};
set #4#windSpeedAt10M={70,80};
set #5#windSpeedAt10M={90,100};
set pack=1;
write;
```

- If we try to set 3 values for #5#windSpeedAt10M then we get an error as ecCodes expects 2 values and we provide 3. If I provide only one it works.

Quality information

Quality information

Quality information

To include data quality information, BUFR can use bit map operator from table D combined with descriptors in table B 08 (significance of qualifiers) to convey information about the precision/confidence of a measurement.

A bit map is a mask made of 1 and 0 according to the following convention:

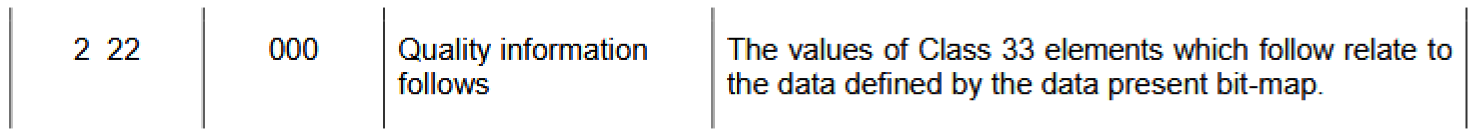
1 → not present

0 → present

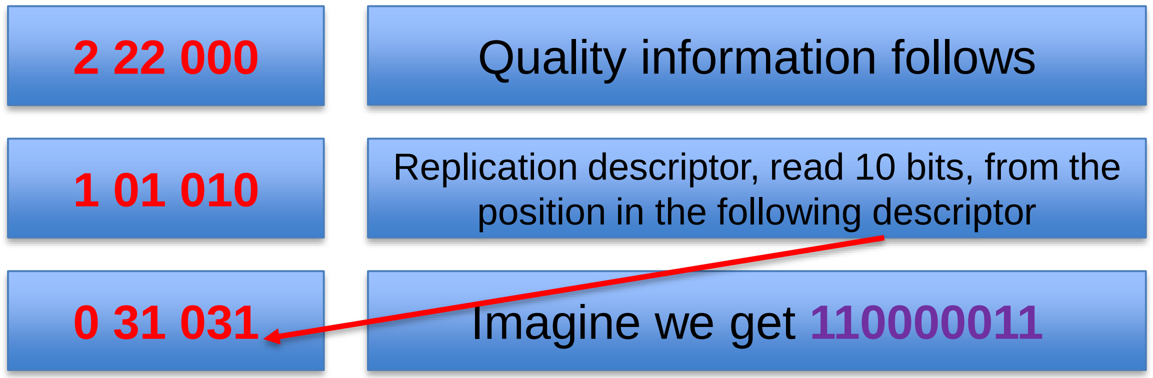
The BUFR key **inputDataPresentIndicator** is a vector key that allows us to define the bitmap.

Quality information

- For example, the Table D operator 222000(class 33 element relates present bitmap) adds quality information (descriptor of class 33) to the keys we decide through a bitmap.



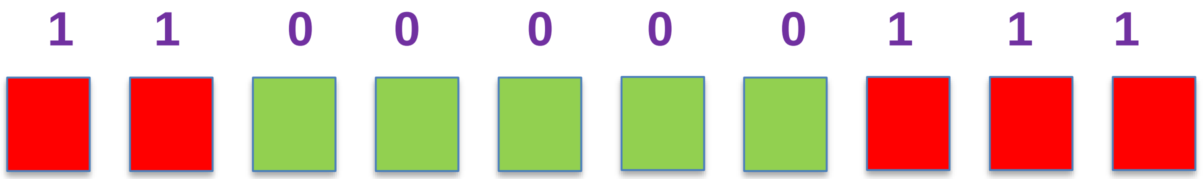
2 22	000	Quality information follows	The values of Class 33 elements which follow relate to the data defined by the data present bit-map.



2 22 000
Quality information follows
1 01 010
Replication descriptor, read 10 bits, from the position in the following descriptor
0 31 031
Imagine we get 110000011

Quality information

Once we have the 10 bit mask, we apply the mask to the data and we end up only with the data values for which the mask contains 0s.



So we only use the data values in green (the ones that have 0 bit map) to store first order statistics (errors). The type of first order statistics is defined by the Table B descriptor 0 08 023.

Quality information(exercise 13)

Create a BUFR message with the same **unexpandedDescriptors** as the file **synop_with_confidence.bufr**.

Print the key **dataPresentIndicator** for this file (**synop_with_confidence.bufr**).

Modify the bitmap so we only have the attribute ***percentConfidence*** for **dewpointTemperature** and **airTemperatureAt2M**.

Hint: The option –ja of **bufr_dump** may be helpful to find the indexes for **airTemperature2M** and **dewpointTemperature**. Then in the bitmap (**inputDataPresentIndicator** array) these indexes must be set to 0 for the bitmap to affect the corresponding variables.

Quality information

```
########################################
# file create_complex_bitmap.filter
# creates a message with a bit map, the indexes for the
0s in the bitmap
# can be found using bufr_dump-ja
#########################################################
############
set inputDataPresentIndicator={1,1,1,1,1,1,1,1,
1,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1
,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1};
set unexpandedDescriptors={307005,
13023,13013,222000,101049,31031,1031,1032,101049,33007};
write;
```

Bitmap + delayed replication

Bit map + delayed replication

BUFR messages can be complex. We may combine delayed replication with bit maps.

```
set compressedData=1;
# we have to use local (ECMWF) tables
set masterTablesVersionNumber=13;
set localTablesVersionNumber=101;
set inputDelayedDescriptorReplicationFactor={1,4,2};
set inputDataPresentIndicator={1, 1, 0, 0};
set numberOfSubsets=5;
set unexpandedDescriptors={310022,301011,301013,
301021,7024,7025,27080,27193,20079, 13040, 20010, 2071,
2171, 8043, 15024, 8043, 8045, 15024, 22094, 8043, 104000,
31001, 2071, 2171,15024, 15042, 224000, 236000, 101000,
31001, 31031, 1033, 1032, 8023, 101000, 31001, 224255};
write;
```

Bit map +delayed replication (exercise 14)

- Run **bufr_filter** with the given rules to create a new message.
- Print the following keys:

delayedDescriptorReplicationFactor
dataPresentIndicators

they should have these values

delayedDescriptorReplicationFactor={1,4,2}

dataPresentIndicator={1, 1, 0, 0}

Check with **bufr_dump -ja** the **dataPresentIndicator** and **delayedDescriptorReplicationFactor** for the message created.

Bit map + delayed replication

Looking at the unexpanded Descriptors

{310022,301011,301013,301021,7024,7025,27080,27193, 20079, 13040, 20010, 2071, 2171, 8043, 15024, 8043, 8045, 15024, 22094, 8043, **104000, 31001, 2071, 2171,15024, 15042**, **224000, 236000, 101000, 31001, 31031, 1033, 1032,** 8023, **101000, 31001, 224255**};

We identify:

- Delayed replication **104000.**
- The first order statistics sequence **224000/224255**.
- Define data present bit map (descriptor **236000**)
- Delayed replication **101000 31001 31031** read 8 bits (**31001** ***delayedDescriptorReplicationFactor***) of the data Section. The bit map is produced with **031031**, a bit set to 0 means present, set to 1 means not present.

Bitmap + replication

If we look at the ***expandedNames*** along with the ***expandedDescriptors***, we can identify the sequences of replications

104000	delayed replication of the 4 following descriptors
31001	Go to section 4 to find how many times (1 time)
2071	**spectrographicWavelength**
2171	**instrumentSerialNumberForWaterTemper..**
15024	**opticalDepth**
15042	**reflectance**

Bear in mind that OpticalDepth and reflectance appear twice before the replication

Bitmap+replication

If we count the number of opticalDepth keys

```
bufr_dump pmap_1.bufr  |grep -c "opticalDepth"
```

We get 4. These 4 come from

2 **opticalDepth** appear before the replication

1 **opticalDepth** appears after the replication and is affected by First Order Statistics Value (FOS) that follows in the sequence.

1 **opticalDepth** appears at the end (not affected by FOS)

If we look at the output of

```
bufr_dump   pmap_1.bufr  |grep –C12  "opticalDepth"
```

We see that only some keys are affected by the FOS. Which ones? The bitmap is the place to look for this information.

Bitmap + replication

The bitmap is encoded as a delayed replication itself

101000 **delayed replication of 1 descriptor**

31001 Go to section 4 to find out how many times (4 times)

31031 dataPresentIndicator (bitmap)

This is the bitmap that applies first order statistical information to keys. Which keys are affected and what is the first order statistical information?

Bitmap+replication

104000,

delayedDescriptorReplicationFactor,

spectrographicWavelength,

instrumentSerialNumberForWaterTemperatureProfile,

opticalDepth,

reflectance,

    firstOrderStatisticalValuesFollow

    defineDataPresentBitmap

101000

delayedDescriptorReplicationFactor

dataPresentIndicator

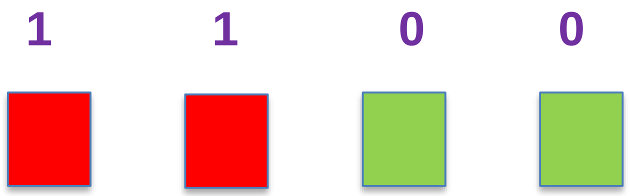
- The *opticalDepth* and *reflectance* are affected by the FOS as it appears right after these keys and is “applied” with the last 2 replications 224255.

Bitmap + replications

```
 104000 104000

    31001 DELAYED DESCRIPTOR REPLICATION FACTOR ( the first replication 1)

     2071 SPECTROGRAPHIC WAVELENGTH

     2171 (TBP) MODIS AEROSOL ALGORITHM

    15024 OPTICAL DEPTH

    15042 (VAL) REFLECTANCE   these are the two keys affected by the following bitmap.

   224000 The statistical values which follow relate to the data defined by the data present bit-
map

   236000 This operator defines the data present bit-map which follows for possible re-use; only
one data present bitmap may be defined between this operator and the cancel use defined data
present bit-map operator

   101000 101000

    31001 DELAYED DESCRIPTOR REPLICATION FACTOR ( second replication 4 opens the bitmap )

    31031 DATA PRESENT INDICATOR

     1033 IDENTIFICATION OF ORIGINATING/GENERATING CENTRE

     1032 GENERATING APPLICATION

      8023 FIRST ORDER STATISTICS

   101000 101000

    31001 DELAYED DESCRIPTOR REPLICATION FACTOR ( last replication 2 ( closes the bitmap )

   224255 224255
```

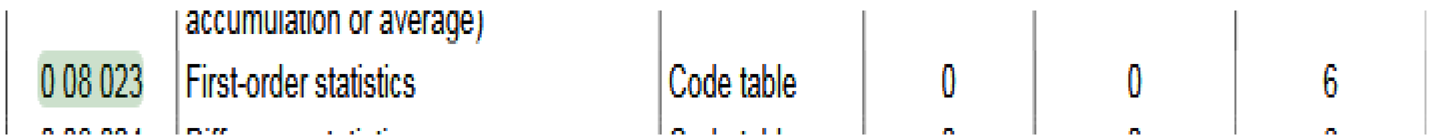
Bit map + delayed replication

- In our case, the bit map is



Which means that affects the last two descriptors. The FOS is given by the descriptor 8023 which is 9 in the bufr file. If we look at the code table, 9 means *best estimate of standard deviation.*

https://software.ecmwf.int/wiki/display/ECC/WMO%3D13+code-flag+table#WMO=13code-flagtable-CL_8



	accumulation or average)				
0 08 023	First-order statistics	Code table	0	0	6

ECMWF EUROPEAN CENTRE FOR MEDIUM-RANGE WEATHER FORECASTS

Bitmap+replication

224000 First orderStatisticsFollow refers to the bitmap
236000 introduces the bitmap for possible reuse
101000 delayed replication of 1 descriptor
31001 This is the delayedReplication factor 4 times
31031 This is the bitmap dataPresentIndicator
1033
1032
8023 Introduces the FirstOrderStatistics type
101000 delayed replication of 1 descriptor
31001 delayedReplicationFactor (2 times)
224255 First order statistical values marker operator

This operator shall signify a data item containing a first order statistical value of the type indicated by the preceding 0 08 023 element descriptor

Bitmap + replication

In this case, the first order statistics 8023 affects the last two of the block created by the 104000. The first two keys are unaffected.

```
#2#spectrographicWavelength=5.5e-07
#2#instrumentSerialNumberForWaterTemperatureProfile=MISSING
#3#opticalDepth={
      0.0377, 0.0314, 0.062, 0.0582, 0.0628, 0.0506, 0.049,
0.088}
#3#opticalDepth->firstOrderStatisticalValue = {
      0.0222, 0.0206, 0.0259, 0.0256, 0.0264, 0.0251, 0.0251,
0.0309}
#1#reflectance=MISSING
#1#reflectance->firstOrderStatisticalValue = MISSING
```

Bitmap+replication

In the filter rules given, modify the first replication factor replacing 1 by 3. What is the effect on the resulting BUFR file?

If we use bufr_dump –ja out.b we find

Three blocks of opticalDepth reflectance but only the last one is affected by the FOS as is the preceding block after the FOS block definition

Tables and references

Latest version of the BUFR tables

http://www.wmo.int/pages/prog/www/WMOCodes/WMO306_vI2/LatestVERSION/LatestVERSION.html

Guide to BUFR format

http://www.wmo.int/pages/prog/www/WMOCodes.html

ecCodes

https://software.ecmwf.int/wiki/display/ECC/ecCodes+Home

The end

Thank you