WCT Snapshot Data Format

Wave Computation Tech.

Sept., 2014

Data Type

1. Original data saved by GUI
2. Exported freq. data
3. Exported transient data

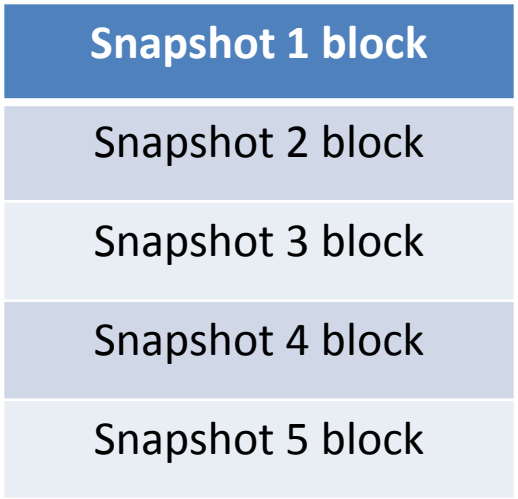
Original Data Saved by GUI

- Snapshot data file position:
  - Project\project_res\snapshot\project_snaps.dat

(Note: all snapshots share 1 data file)

- The data file use binary format, mix with ***char***, ***int*** and ***floating numbers***.

Data file basic structure



Data structure for a Snapshot Data Block

Data Type	Length in Byte	Meaning
int	4	Length of snapshot name. We can define it as ***N***
char	following ***N*** bytes	snapshot name
int	4	How many sampling positions in this snapshot We can define it as ***M***
		Data frame for sampling position 1
		Data frame for sampling position 2
		…..
		Data frame for sampling position M

Data structure for a Snapshot sampling position

Data Type	Length in Byte	Meaning
double	4	Position x
double	4	Position y
double	4	Position z
int	4	Data length of a component. Unit is bytes. For example, ex frame has length of 10. This value is sizeof(float)*10=40 We can define it as ***NN***
	NN	Ex frame, if ex need to be recorded
	NN	Ey frame, if ex need to be recorded
	NN	Ez frame, if ex need to be recorded
	NN	Hx frame, if ex need to be recorded
	NN	Hy frame, if ex need to be recorded
	NN	Hz frame, if ex need to be recorded

Example Matlab code to Load a snapshot file

assume it record *Ex, Ey & Hz* components only

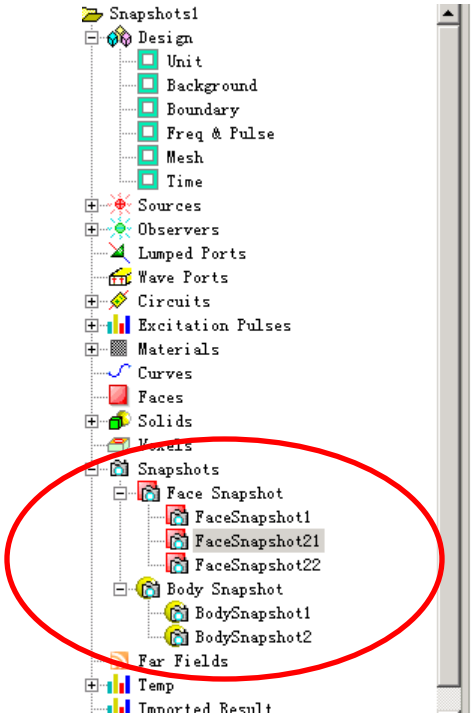
```
fid = fopen( 'xxxxx', 'r' );    % open data file xxxxx
            % note: matlab code is similar to C++ code, only the flag is different.

% for 1st snapshot
nNameLength = fread( fid, 1, 'int32' );          % length of name string
strName = fread( fid, nNameLength, 'uchar' );  % name
nPos = fread( fid, 1, 'int32' );                  % how many sampling positions
for j = 1:nPos,                                   % for each sampling position
    pos= fread( fid, 3, 'double' );   % pos(0) , (1), (2) is x, y, z
    nLen = fread( fid, 1, 'int32' );   % BYTE length of each recording component
    nLen = nLen / 4;                  % convert it from BYTE length to FLOAT length
    ex_pos(j) = fread( fid, nLen, 'float32');     % ex component for position j
    ey_pos(j) = fread( fid, nLen, 'float32');     % ey component for position j
    hz_pos(j) = fread( fid, nLen, 'float32');     % hz component for position j
end;

% for 2nd snapshot, just repeat above code
……
```

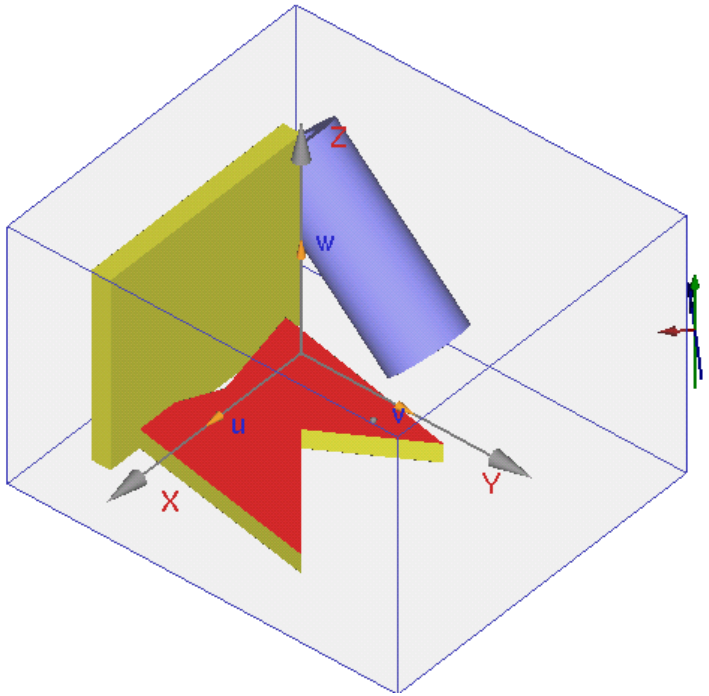
Exported Freq. Data

The steps to generate freq. snapshot data file in WCT

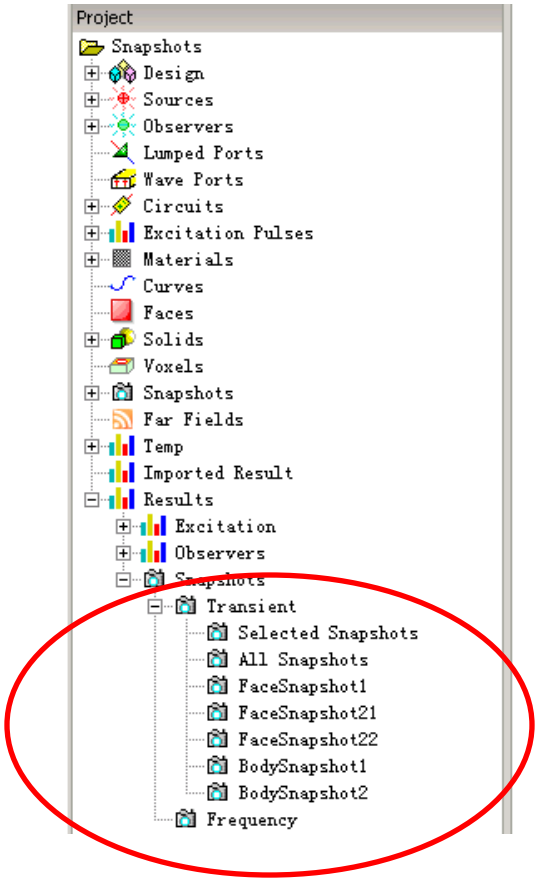
1. Let's use WCT tutorial case:
Tutorial-2012-08\Electromagnetic\Functionalities\Snapshots\Snapshots.wnt
(if you use other version tutorial, just try to find: ???\Functionalities\Snapshots\Snapshots.wnt)

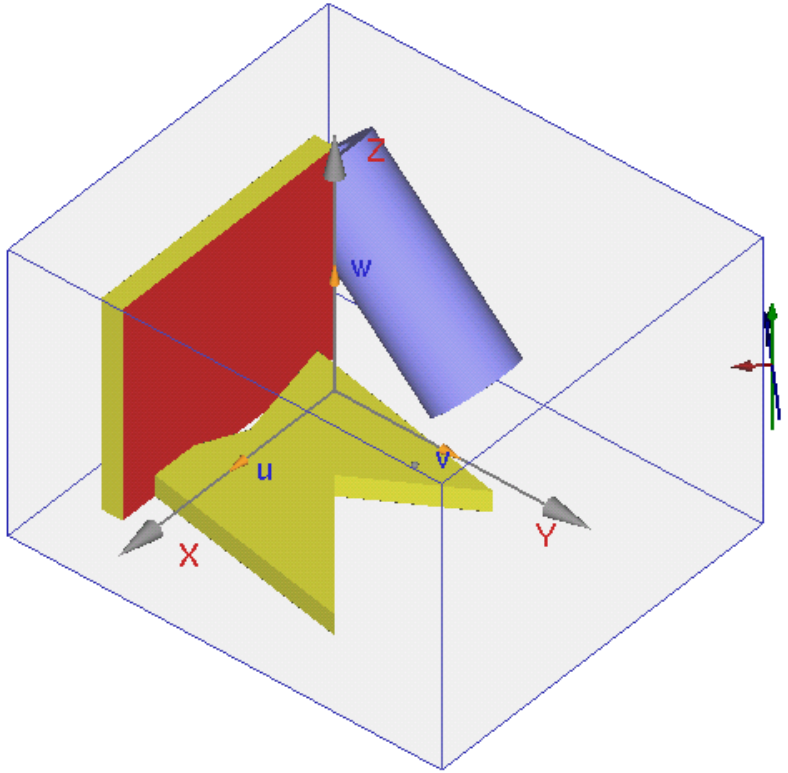
2. This case already has several snapshots as following:



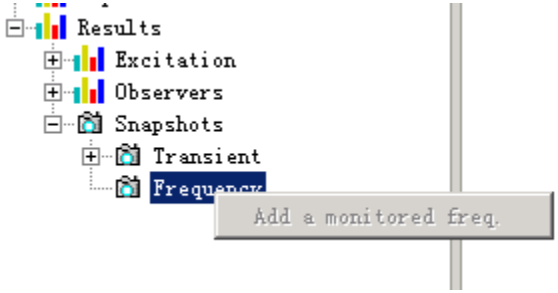


3. Run the simulation to get the ***transient*** result.



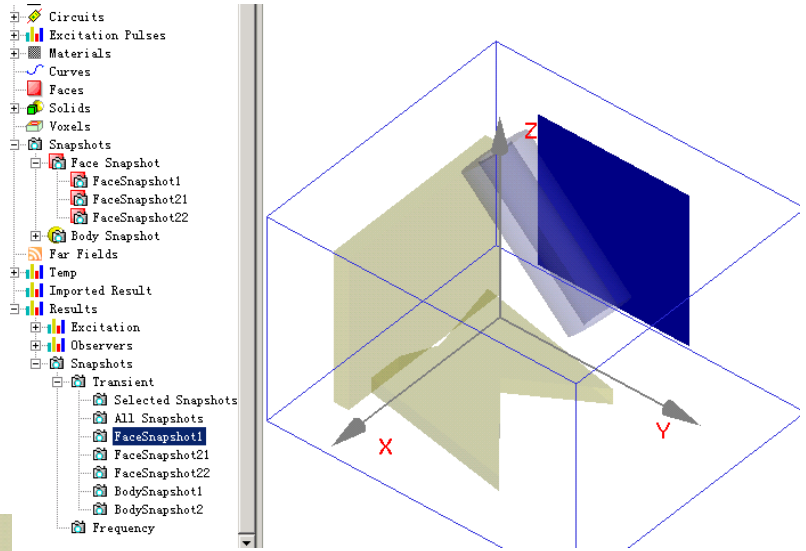


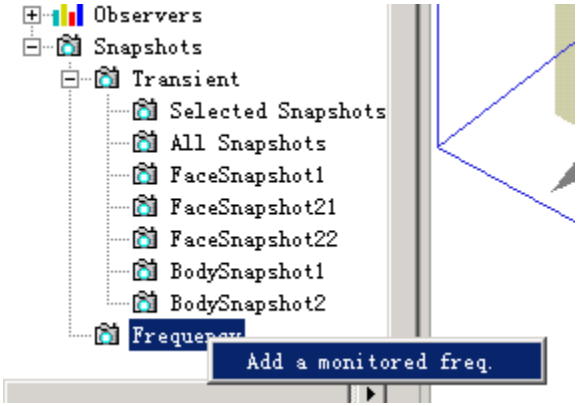
4. Define which freq. you want to observe for the snapshots, for example, 1 GHz & 0.5 GHz



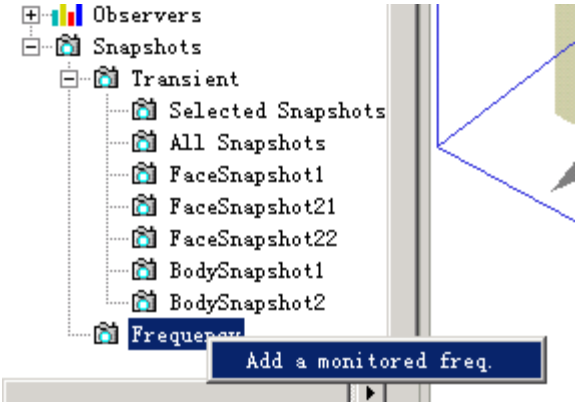
Note: in normal state, the freq. snapshot definition is disabled, you need to switch to “showing snapshot” mode to enable it.

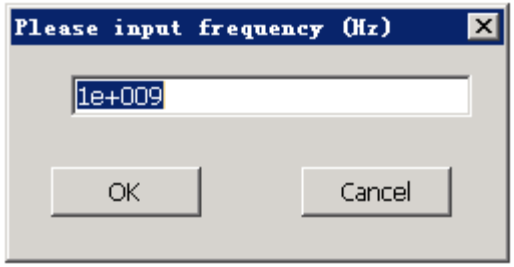
For example, show the transient “Facesnapshot1” firstly, then the menu “Add a monitored freq.” will be enabled.

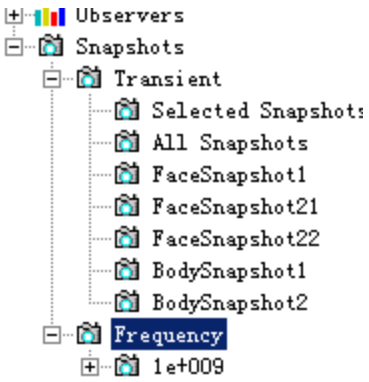




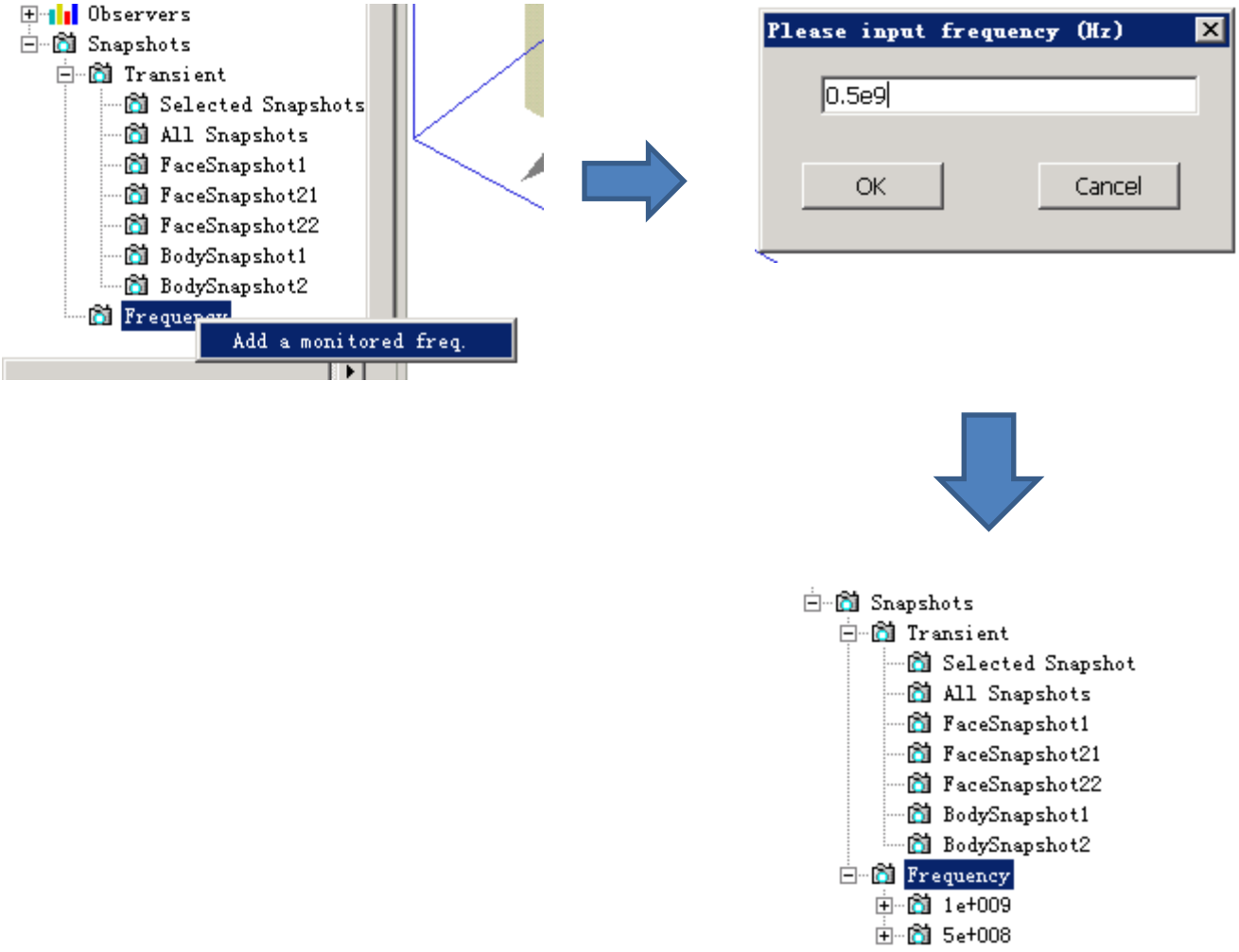
4.1 Define freq. snapshot at 1 GHz.



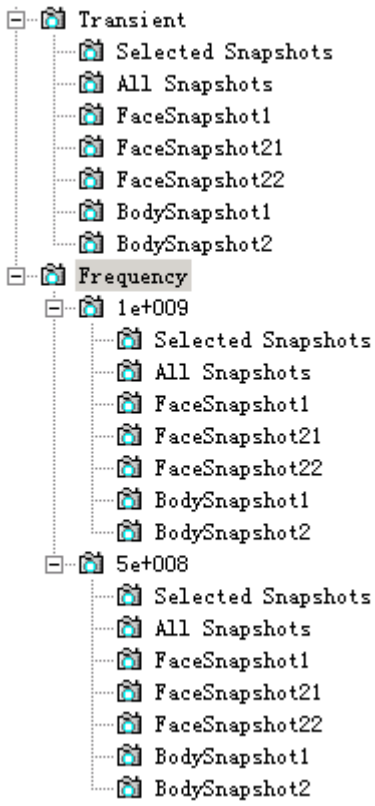




4.2 Define freq. snapshot at 0.5 GHz.

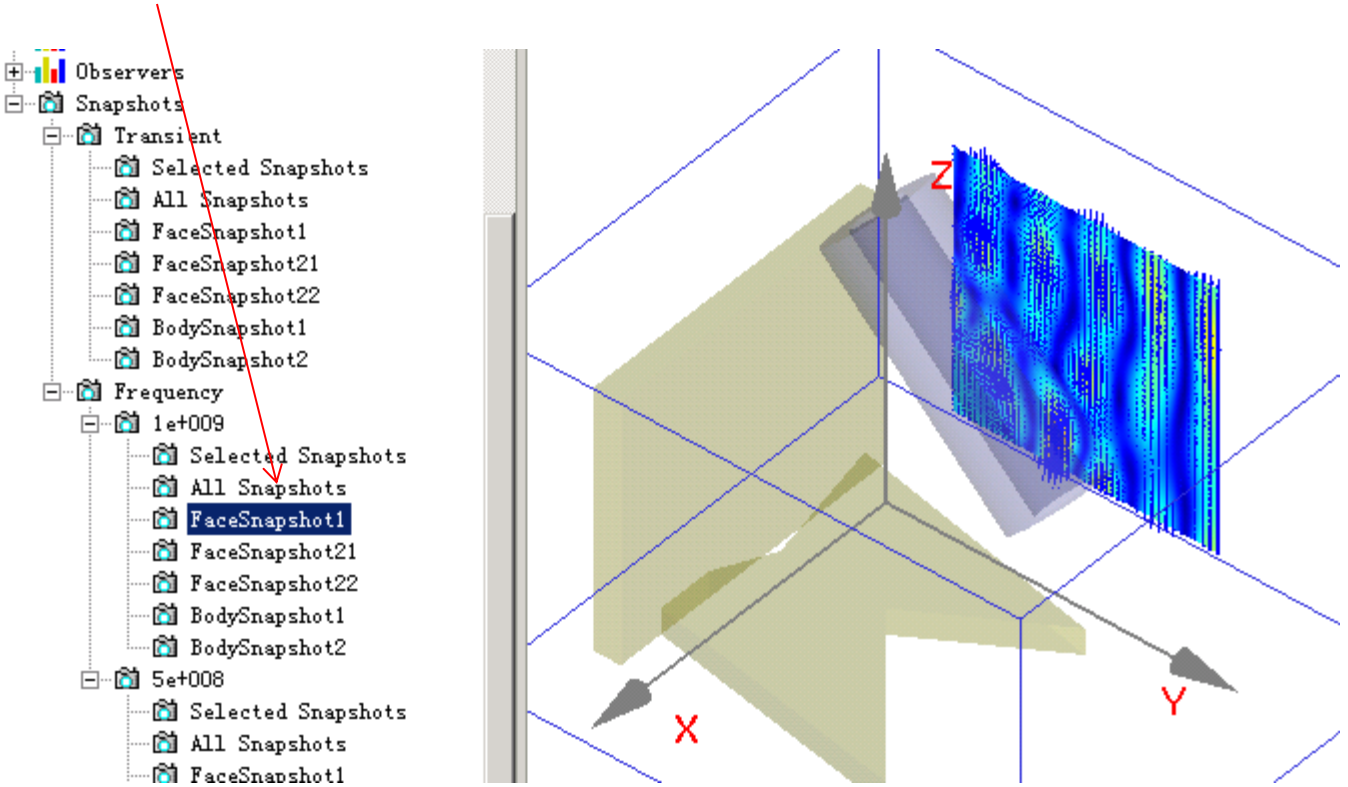


Finally, we get snapshots at 1 GHz & 0.5 GHz

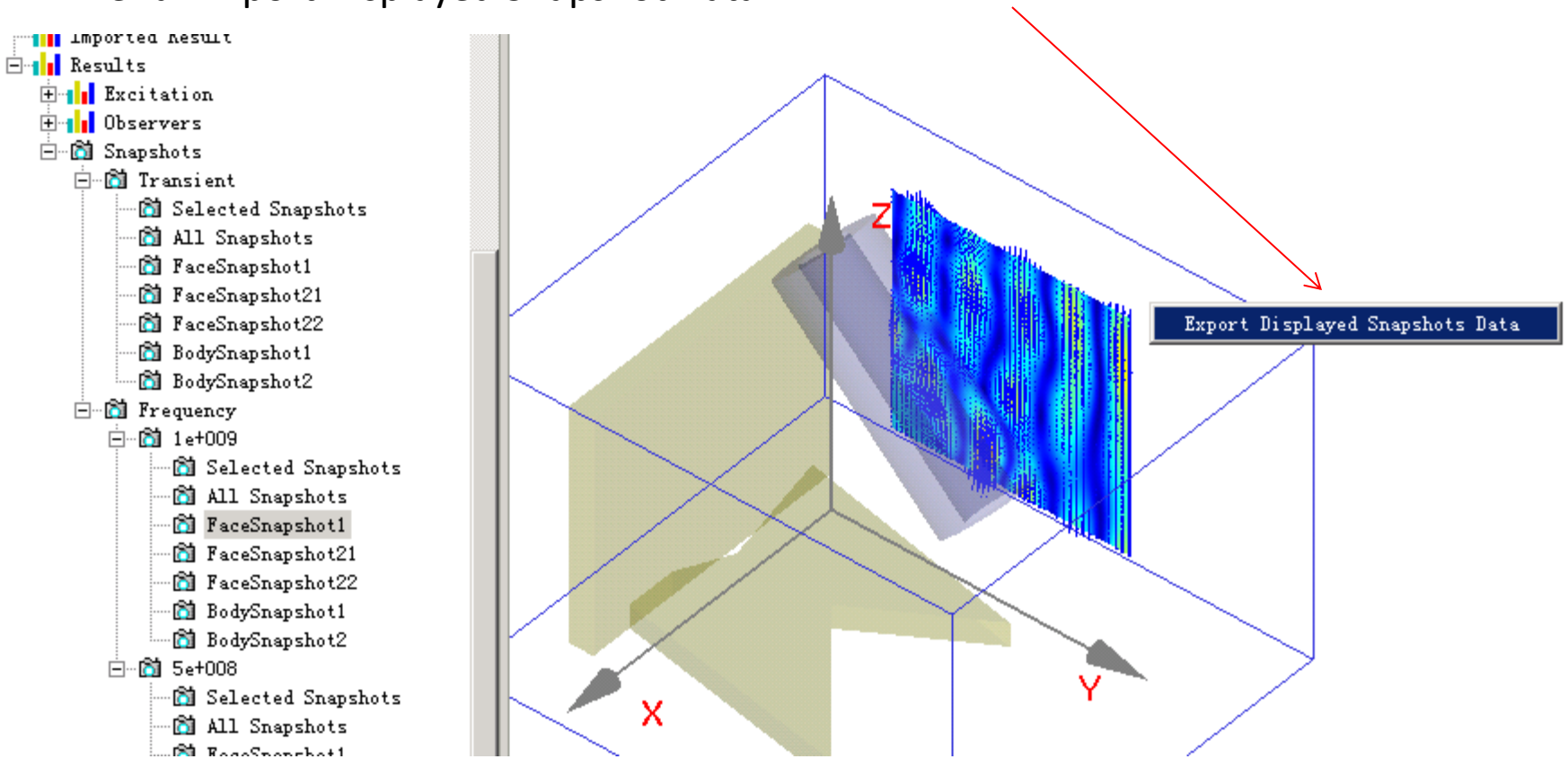


Assuming that we will export the “**Facesnapshot1**” at ***1*** GHz and “**Facesnapshot21**” at ***0.5*** GHz, separately.

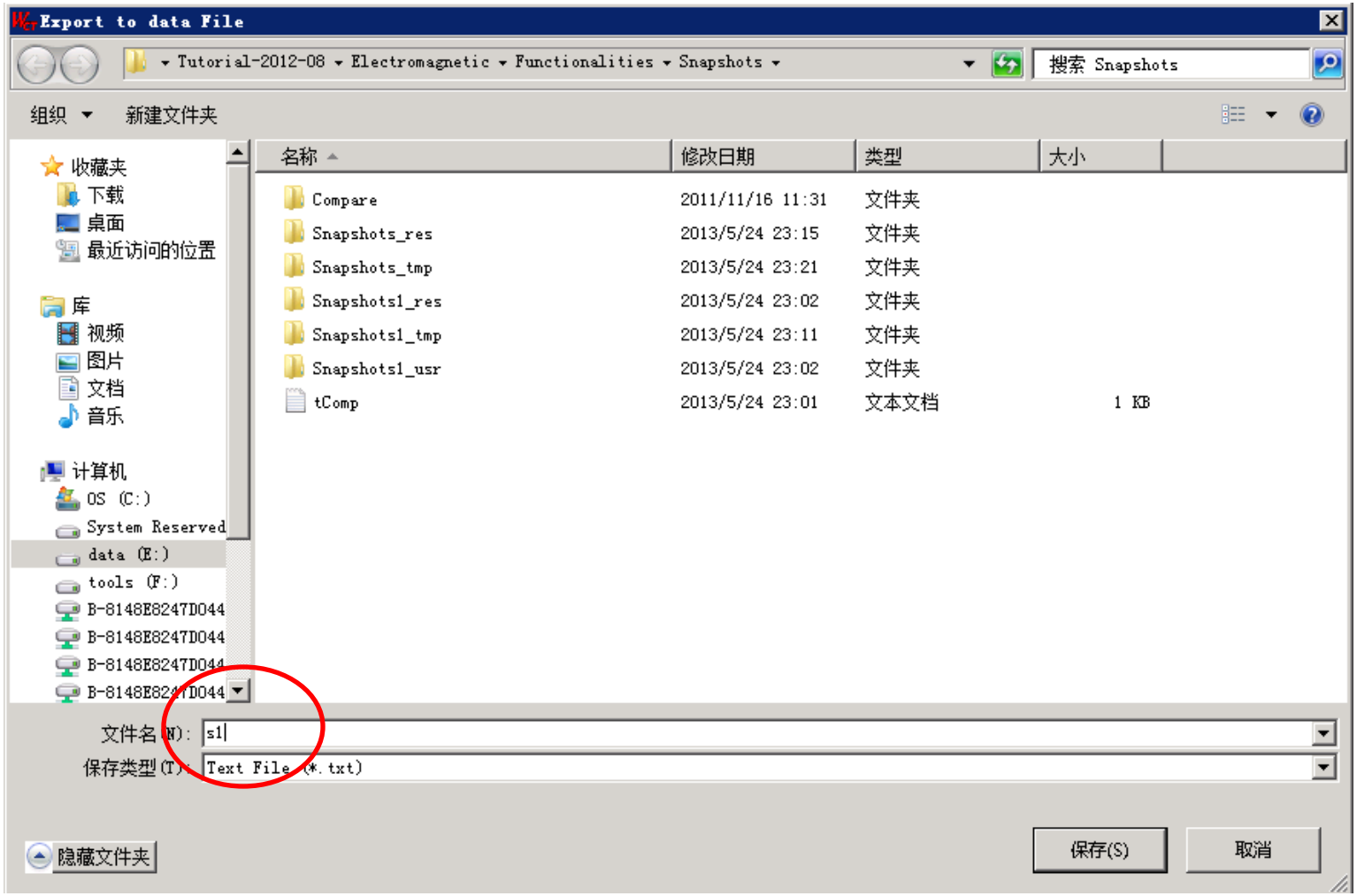
5.1. Double click this node to let “Facesnapshot1” at 1 GHz is shown.



Move mouse to main 3D canvas and click right mouse button to popup menu “Export Displayed Snapshot Data”



Let’s say the data file is ***s1.txt***



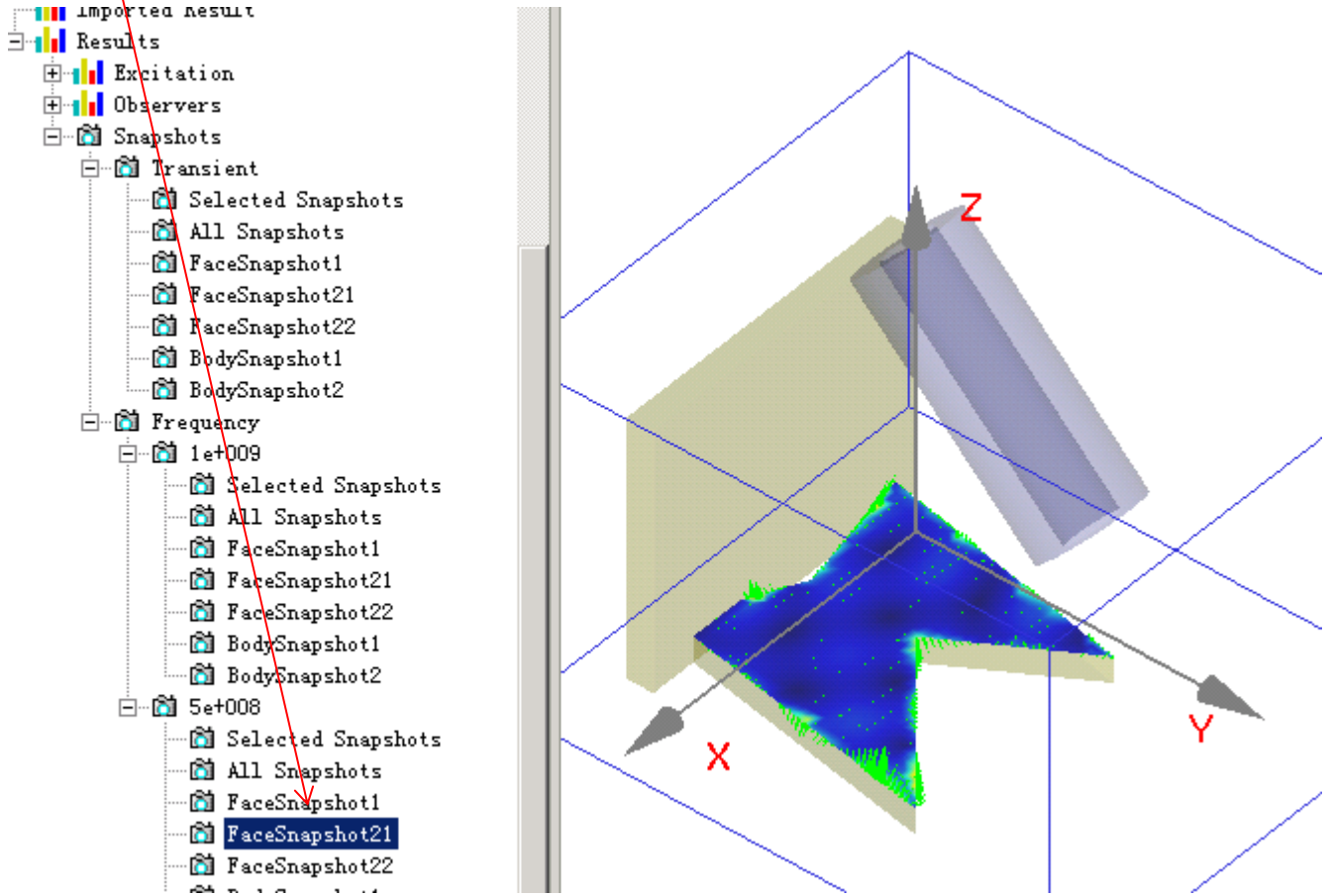
After the data file ***S1.txt*** has been exported, we can check the data in the file.

```
%Wave Computation Technology Snapshot Freq Data
%Version 1.0.0
%Note: the output H field value is [ 377.0 * real_H_field ]
%
%FaceSnapshot1::::1e+009
%x              y              z              [Ex Re]                    [Ex Im]
                [Ey Re]                       [Ey Im]                    [Ez Re]
                [Ez Im]                       [Hx Re]                    [Hx Im]
                [Hy Re]                       [Hy Im]                    [Hz Re]
                [Hy Im]

-0.9            -0.5           -0.5           -0.16645                   -0.0423294
                -0.0632416                    -0.160976                  0.527346
                0.595204                      0                  0
```

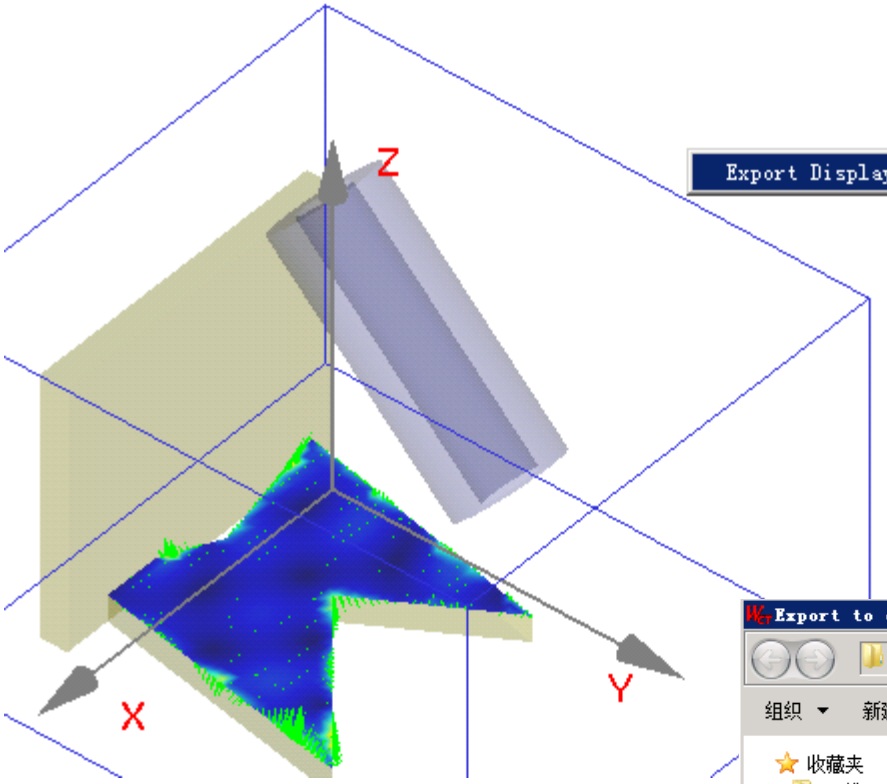
You can see the content of this file is for FaceSnapshot1 at 1e+009 Hz.

Next, we will export data for “Facesnashop21” at 0.5 GHz.

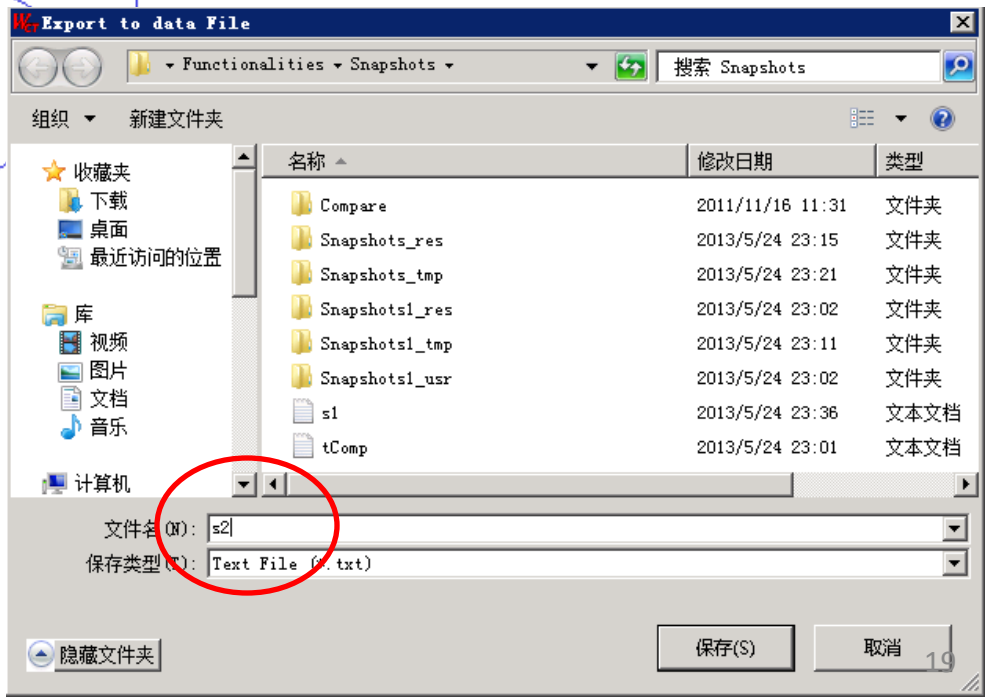
5.2. Double click this node to let “Facesnapshot21” at 0.5 GHz is shown.



Use menu “Export Displayed Snasphot Data” to export data to ***s2.txt***.



Export Displayed Snapshots Data



After the data file ***S2.txt*** has been exported, we can check the data in the file.

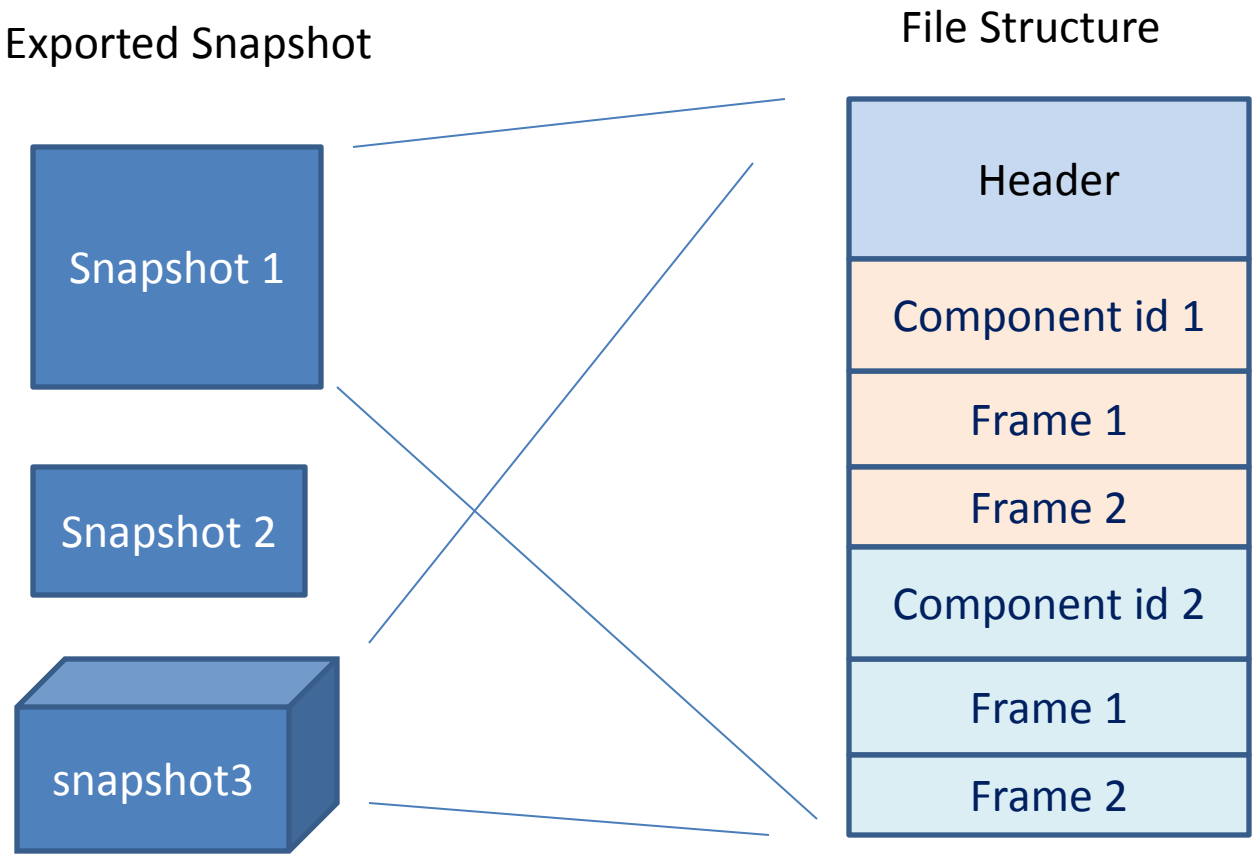
```
%Wave Computation Technology Snapshot Freq Data
%Version 1.0.0
%Note: the output H field value is [ 377.0 * real_H_field ]
%
%FaceSnapshot21::::5e+008
%x              y              z              [Ex Re]                [Ex Im]
                [Ey Re]                       [Ey Im]                [Ez Re]
                [Ez Im]                       [Hx Re]                [Hx Im]
                [Hy Re]                       [Hy Im]                [Hz Re]
                [Hy Im]

0.563333        0.433333       -0.4           0                      0
                0                     0       0                      0
                -1.01123                      0.0780922              0.519358
                0.562987                      0.494926               0.0275544
```

You can see the content of this file is for FaceSnapshot21 at 500 MHz.

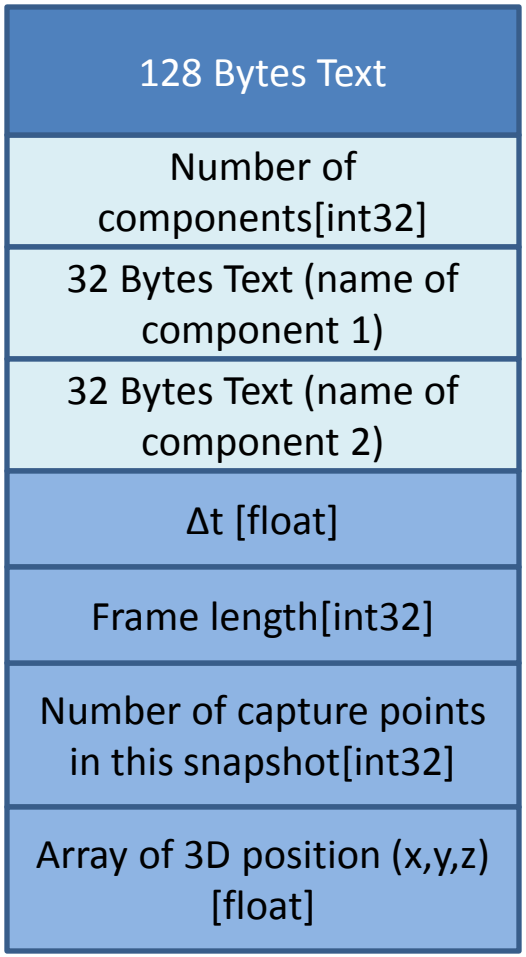
Exported Transient Data

Type I: Frame by Frame

Rule: one snapshot one file

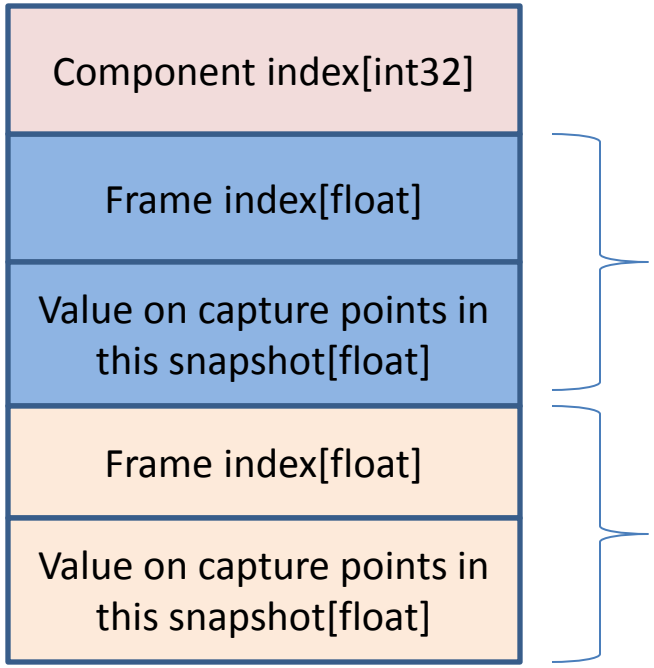


Header



Note: for surface current, we denote it as H components. So, in this case, Hx is the current in X direction.

Frame Data


Component index[int32]
Frame index[float]
Value on capture points in this snapshot[float]
Frame index[float]
Value on capture points in this snapshot[float]

Snapshot Loading Matlab Code 1/3

```
clear all;
% open file
fid = fopen( 'E:\wct_src\test cases\Tutorial-2012-08\Electromagnetic\Functionalities\Snapshots\a1.bin', 'rb' );
% target file
if( fid == -1 )
   return;
end;
%% read 128 file header info
info = fread( fid, 128, '*char' );
%% number of component
nComp = fread( fid, 1, '*int' );
%% components name
sCompName = cell(nComp,1);
for k = 1 : nComp,
   tmp = fread( fid, 32, '*char' );
   sCompName{k, 1} = tmp;
end;
```

Snapshot Loading Matlab Code 2/3

```
%% dt
dt = fread( fid, 1, '*float32' );
%% frame number
nFrame = fread( fid, 1, '*int' );
%% number of capture points
nRecv = fread( fid, 1, '*int' );
%% position of recv.
vRecvPos = fread( fid, [3, double(nRecv)], '*float32' );
vRecvPos = vRecvPos'; %% transpos to make it looks better, but it is not must-be operation.
%% read component by components
data = zeros( double(nComp), double(nRecv + 1), double(nFrame) );
for j = 1 : nComp,
   idxComp = fread( fid, 1, '*int' );
   data(j, :, : ) = fread( fid, [double(nRecv + 1), double(nFrame)], '*float32' );  % +1 is for frame index
end;
```

Snapshot Loading Matlab Code 3/3

```
%%
fclose( fid );

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% verify data
% the first value of each frame is frame index
dispId = 3;    % component id

% recvId = 1;  % frame index
recvId = 2;  % 1st recv.

dispData = squeeze( data(dispId, recvId, :) );
figure;
plot( dispData );
```