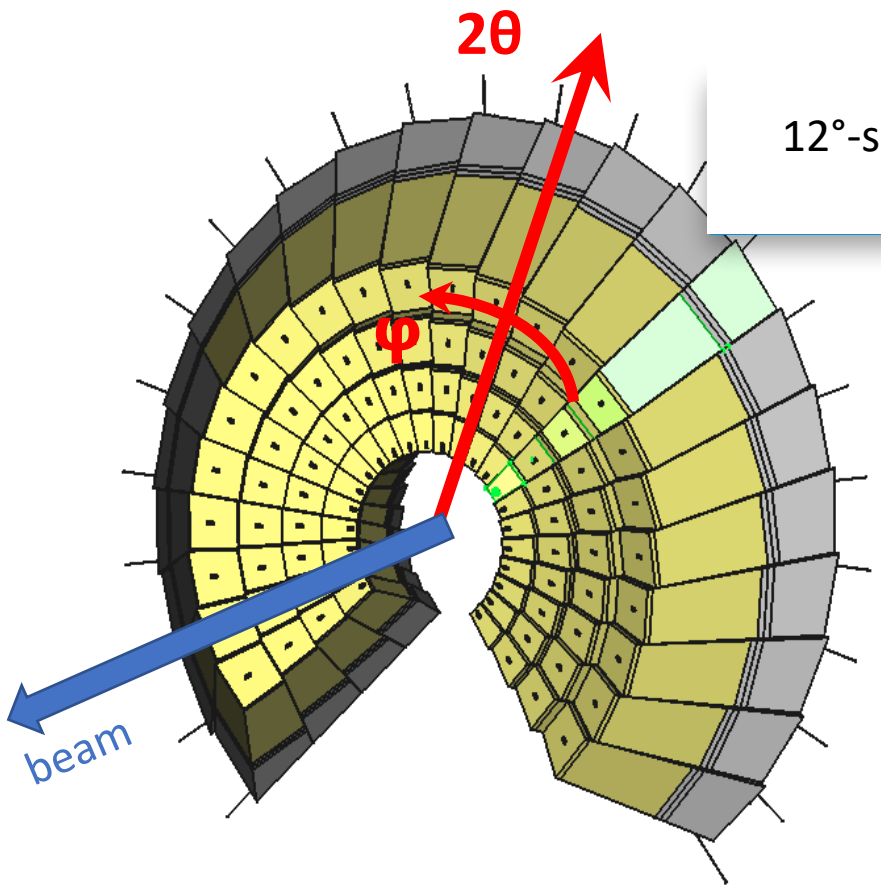
Background information on the channel mapping for the DREAM EndCap data collected in the V20 test performed in July 2019

Written by Irina Stefanescu, ESS DG

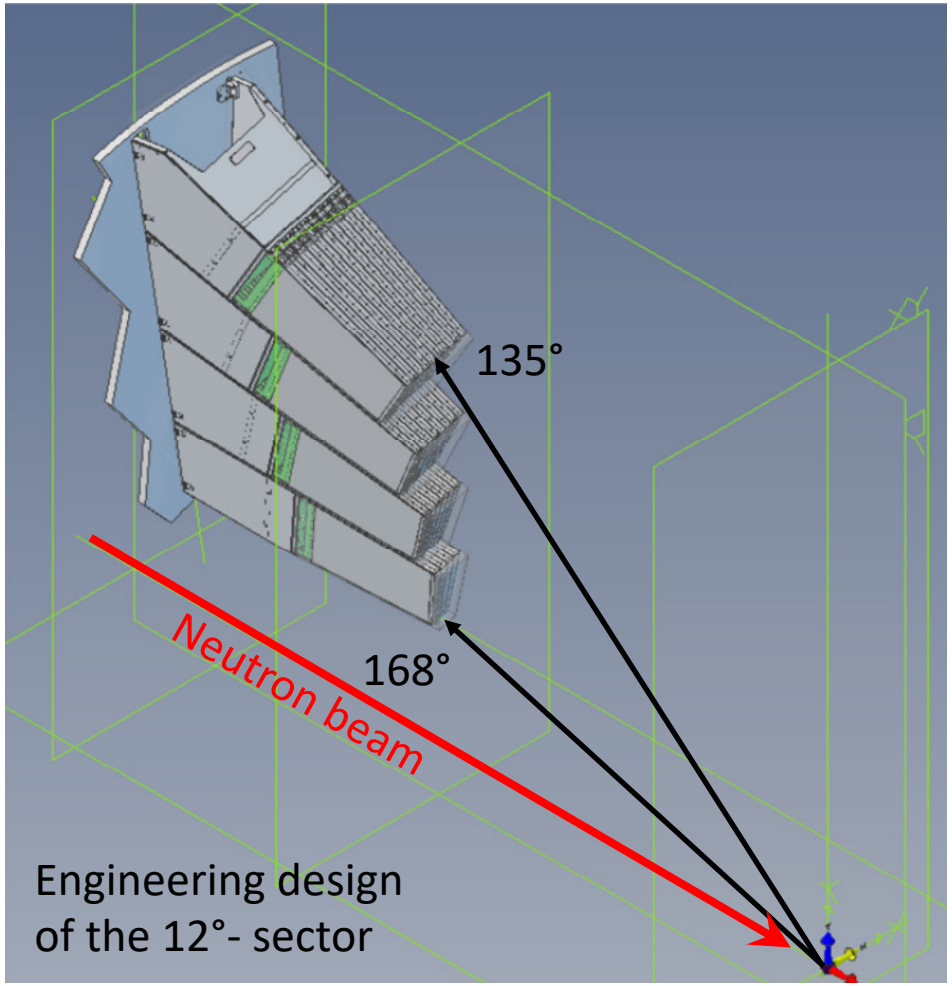
July 2020

Some background information on DREAM EndCap

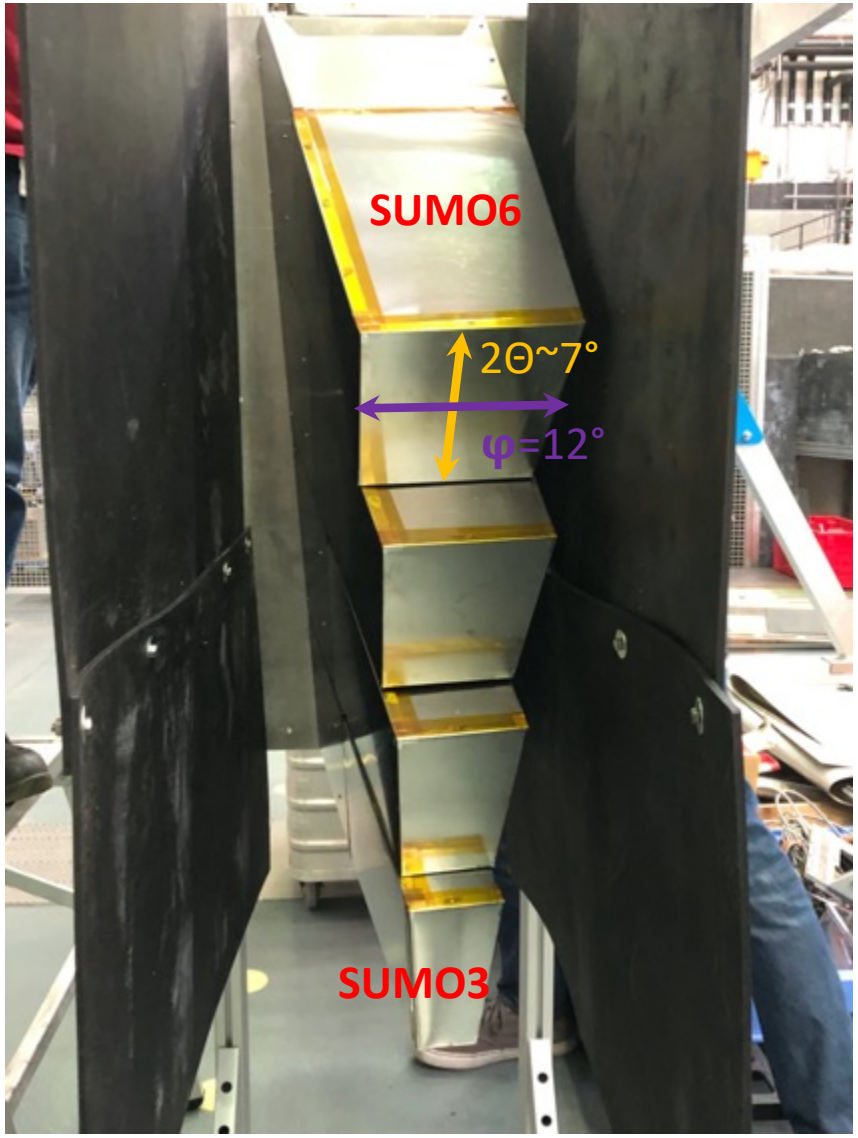


12°-sector

EndCap = assembly of 12°-sectors, each sector consisting of 4 detector modules (SUMOs)

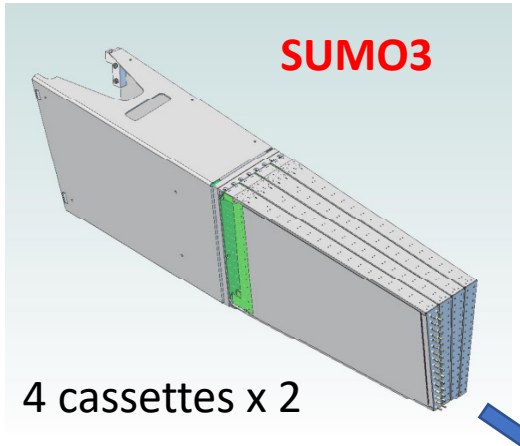


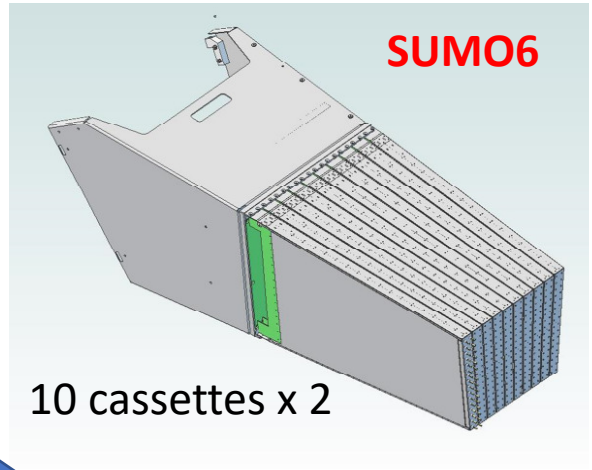
Engineering design
of the 12°- sector

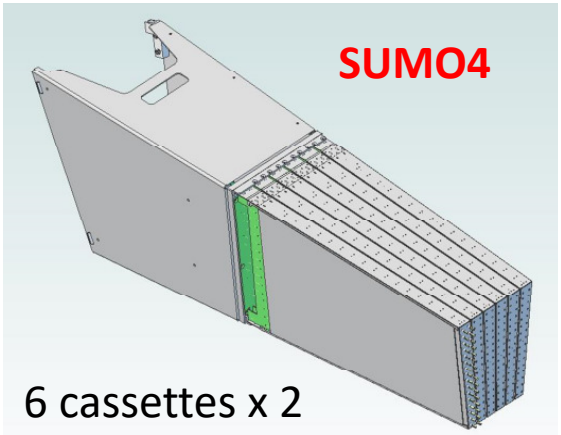


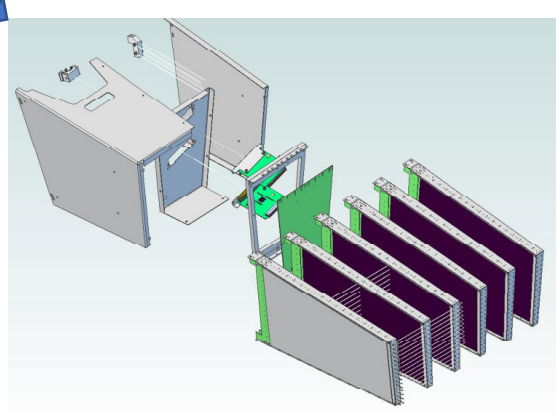
The DREAM 12°- sector @V20

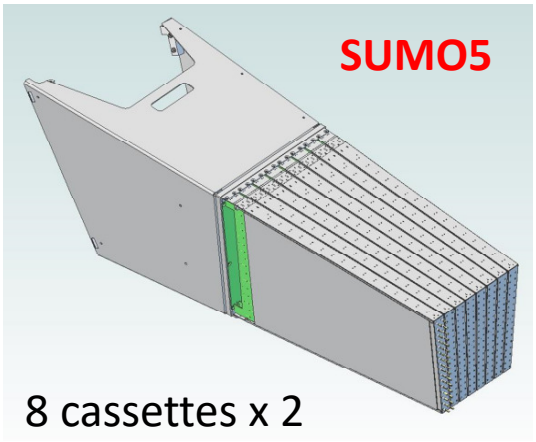
4 SUMO modules (SUper MOdules) mounted on the same backplate = mounting unit





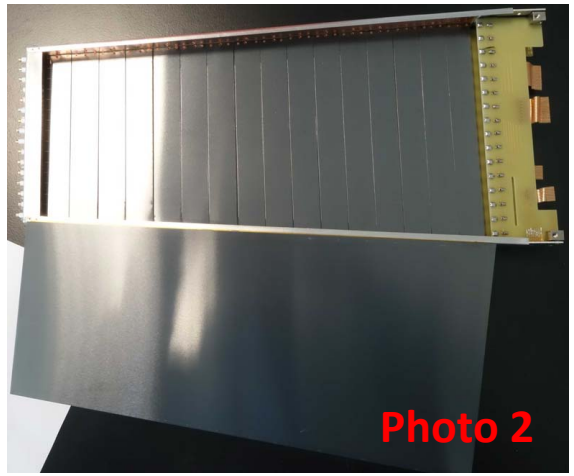


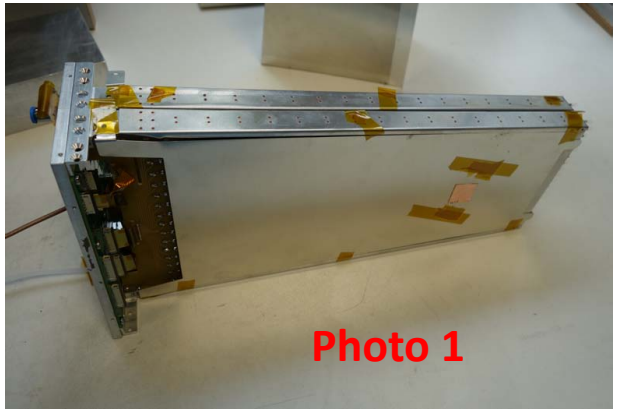


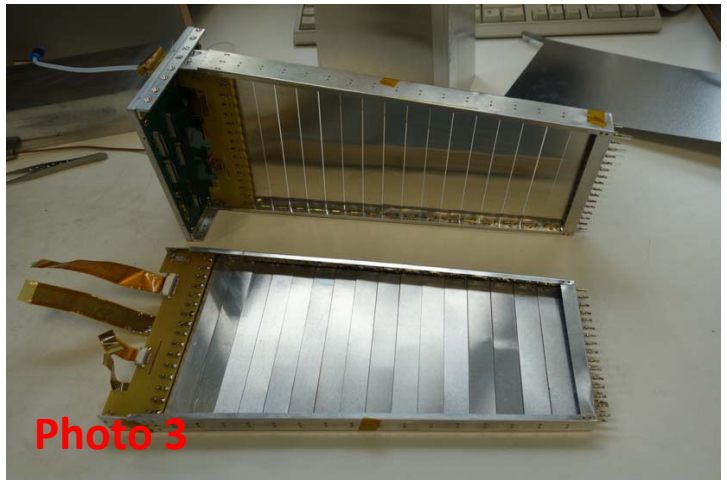


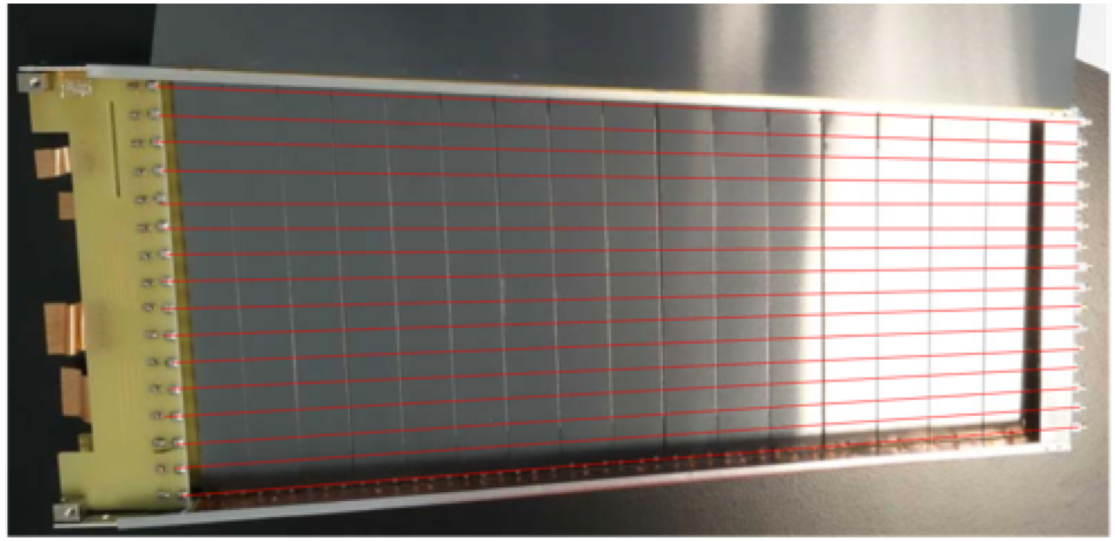
16 wires

16 equal strips



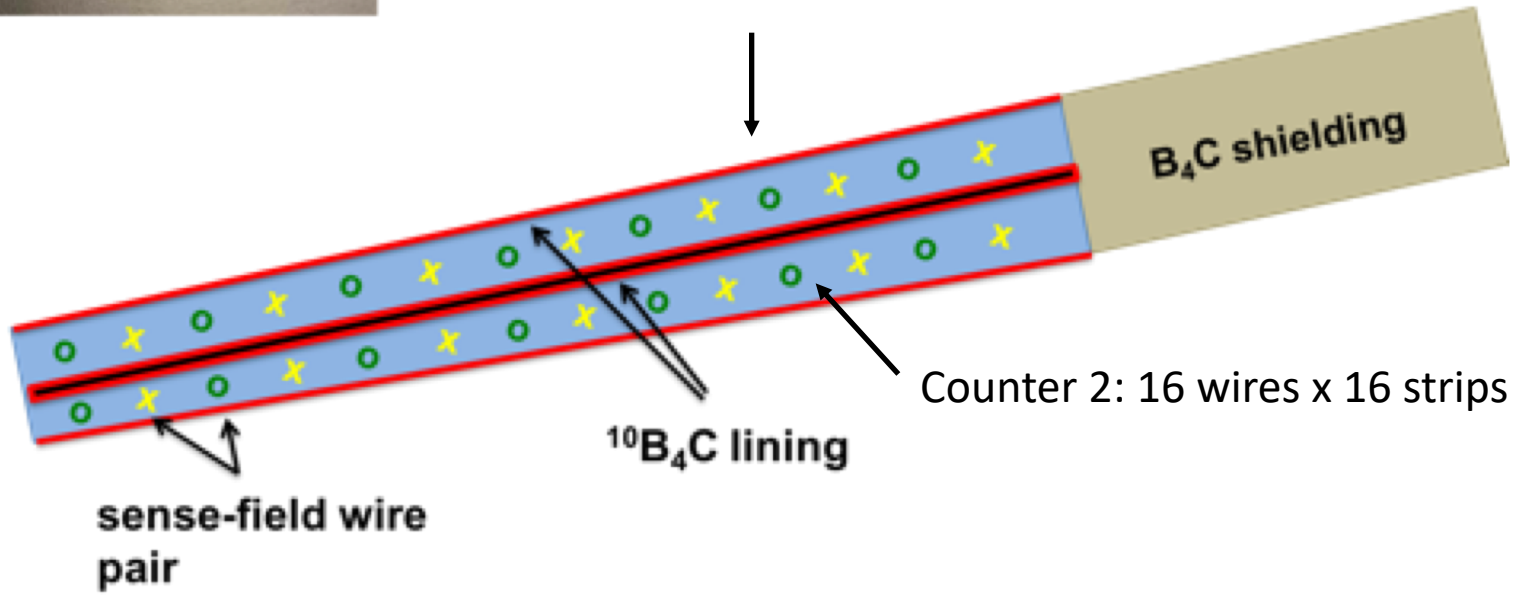






Each segment consists of 2 independent wire counters sharing a common segmented cathode.

Counter 1: 16 wires x 16 strips



EndCap data analysis, channel mapping

ASIC channel# for the wire =‘anode’ events in the data file

ASIC channel# for the cathode strip = ‘cathode’ events in the data file

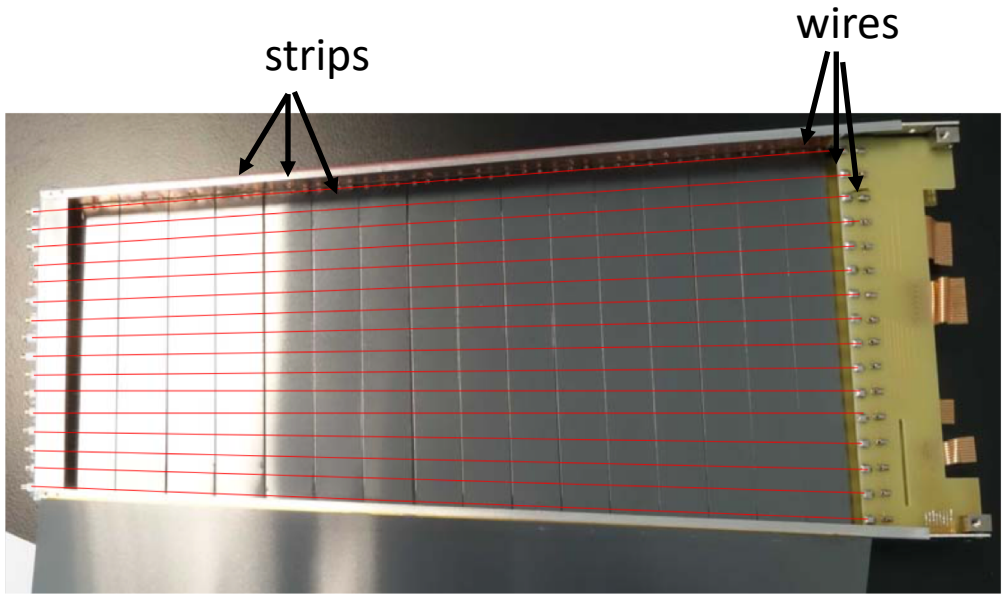
sumo_voxel_map_20190711

sumoType	wireLayer	anode	cathode	absoluteXPosition	absoluteYPosition
3	0	0	0	0	0
3	0	0	1	0	1
3	0	0	2	0	2
3	0	0	3	0	3
3	0	0	4	0	4
3	0	0	5	0	5
3	0	0	6	0	6
3	0	0	7	0	7
3	0	0	8	0	8
3	0	0	9	0	9
3	0	0	10	0	10
3	0	0	11	0	11
3	0	0	12	0	12
3	0	0	13	0	13
3	0	0	14	0	14
3	0	0	15	0	15
3	0	1	0	1	0
3	0	1	1	1	1
3	0	1	2	1	2

Use these 3 columns to get the plots shown in the next two slides

Information on the wiring of the detector included in the CSV file sent by CDT in summer 2019:

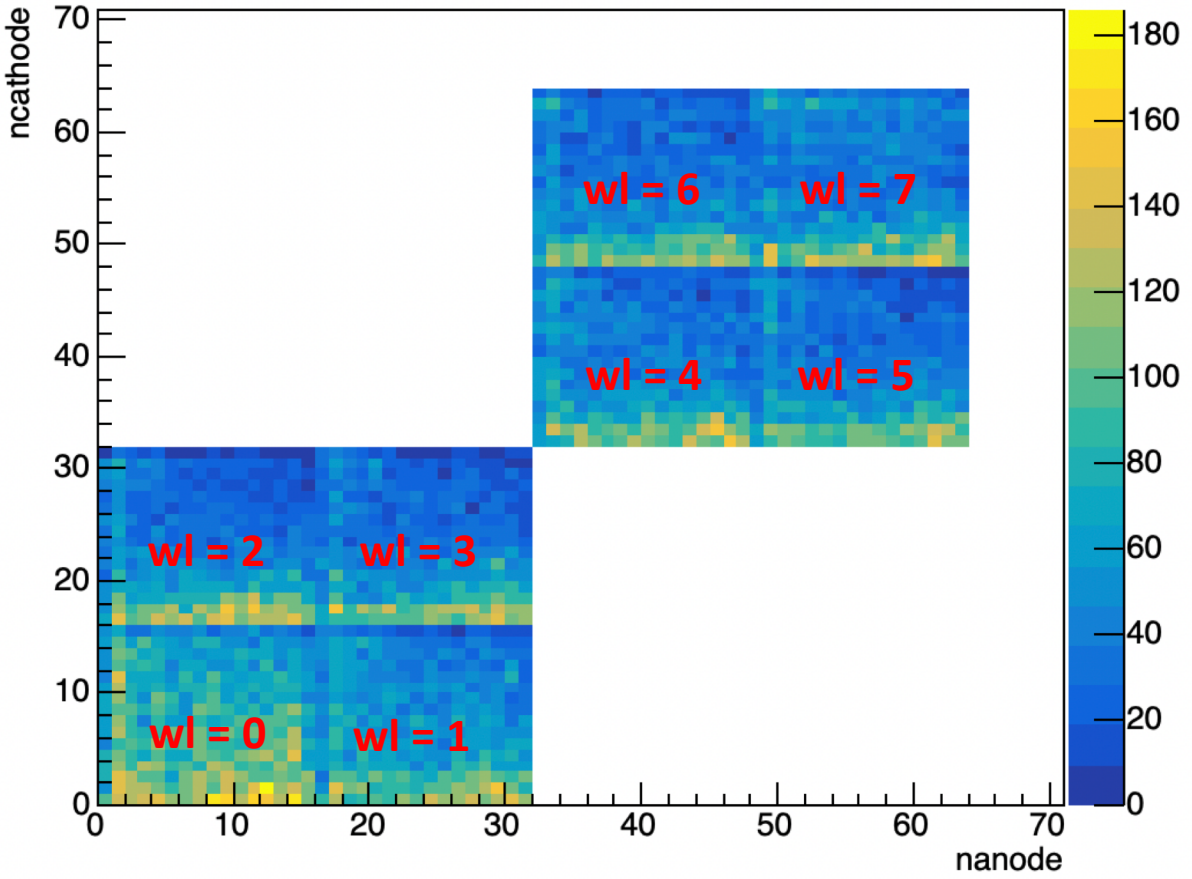
sumo_voxel_map_20190711.csv



There are 16x2 wires and 16 strips in each segment, so each segment is readout by 32 + 16 = 48 ASIC channels.

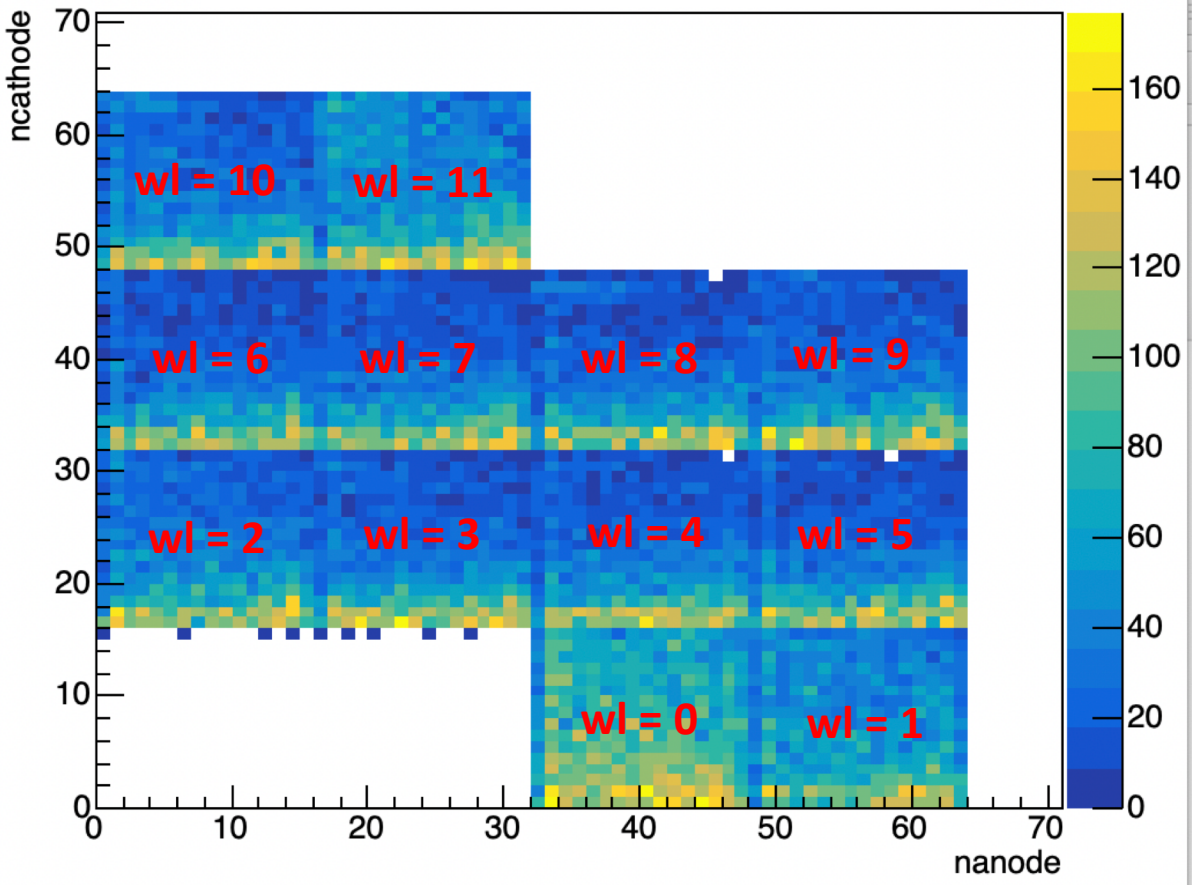
Plot of the events in the V20 data file as 2D histograms anode:cathode

SUMO3



4 segments x 2 counters/segment = 8 wire layers

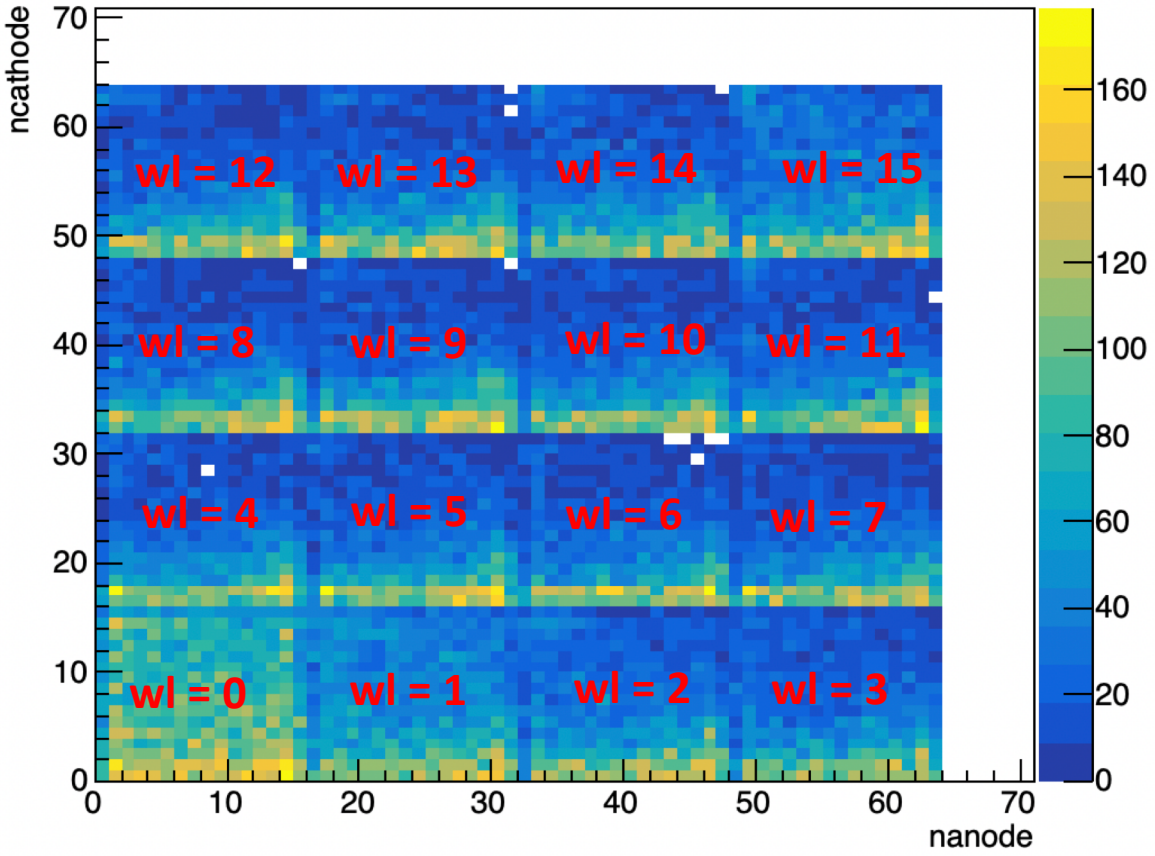
SUMO4



6 segments x 2 counters/segment = 12 wire layers

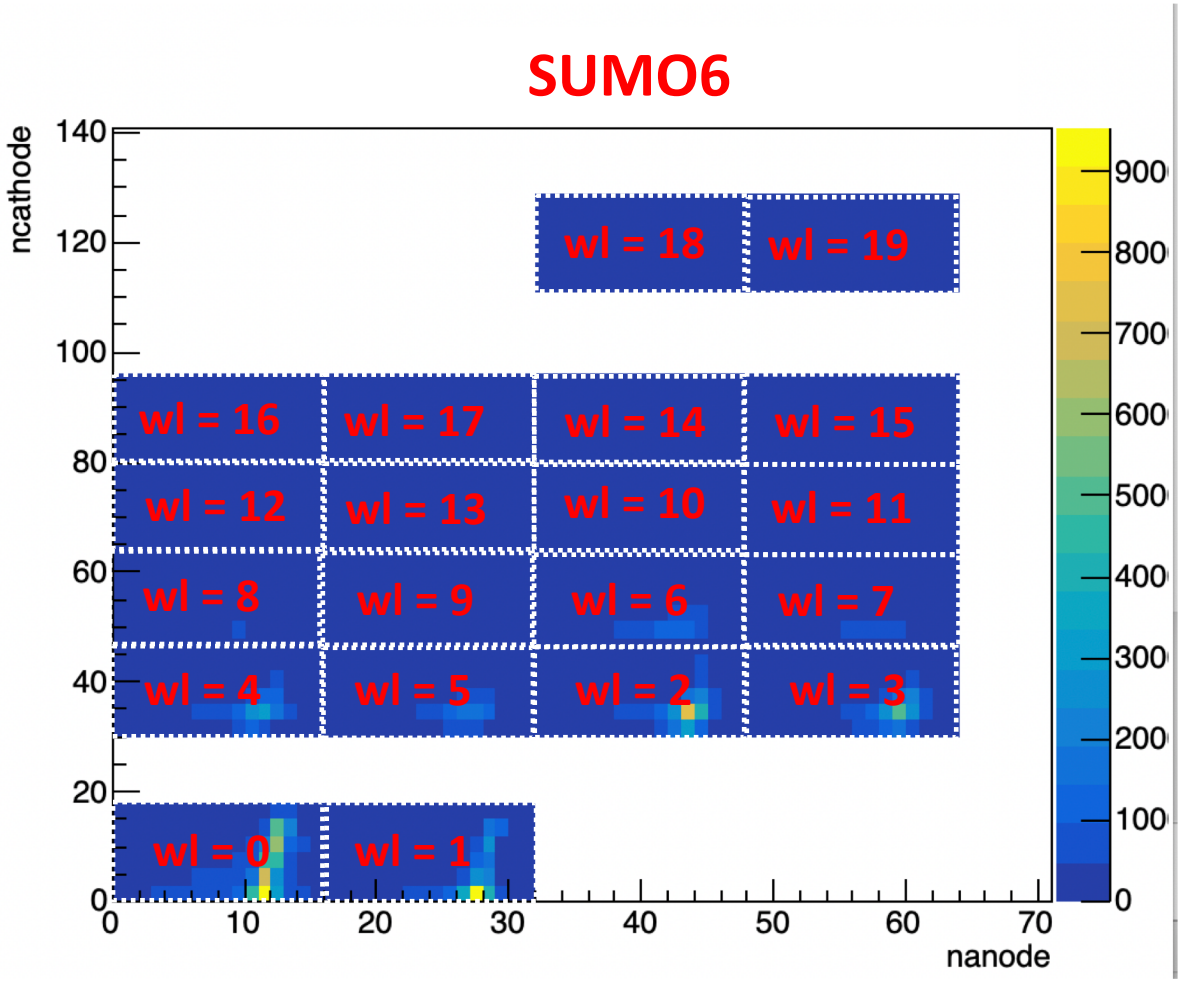
X-axis = absoluteXPosition (5th column in the CDT csv file), Y-axis = absoluteYPosition (6th column in the CDT csv file)
Squares correspond to the wire layer (wl), 2nd column in the CDT cvs file

SUMO5



8 segments x 2 counters/segment = 16 wire layers

SUMO6



10 segments x 2 counters/segment = 16 wire layers

This is how I use this information in my code (analysis_cdt.C) to convert from ASIC channels and wire layers to user-defined ‘location’ of the neutron event in the detector volume.

The info shown graphically in the last two slides is captured in the code like this:

```
int s3[4][4] = {{0,2,-1,-1}, {1,3,-1,-1}, {-1,-1,4,6}, {-1,-1,5,7}};  //sumo3
int s4[4][4] = {{-1,2,6,10}, {-1,3,7,11}, {0,4,8,-1}, {1,5,9,-1}};  //sumo4
int s5[4][4] = {{0,4,8,12}, {1,5,9,13}, {2,6,10,14}, {3,7,11,15}};  //sumo5
int s6[4][8] = {{0,-1,4,8,12,16,-1,-1},{1,-1,5,9,13,17,-1,-1},{-1,-1,2,6,10,14,-1,18},{-1,-1,3,7,11,15,-1,19}};  //sumo6
```

Wire layer vectors

For-loop through the neutron events

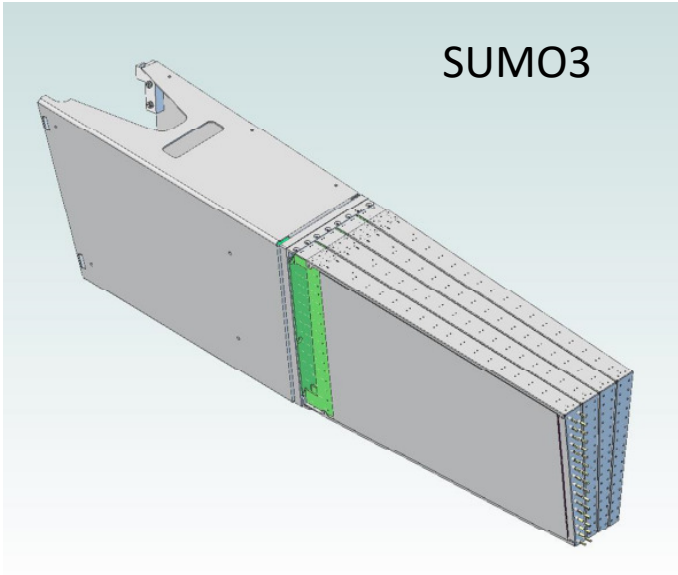
```
factor_a = floor(ncathode/16);
factor_b = floor(nanode/16);

switch (nsumo){

    case 3:
        nw_layer = s3[factor_b][factor_a];
        break;

    case 4:
        nw_layer = s4[factor_b][factor_a];
        break;

    case 5:
        nw_layer = s5[factor_b][factor_a];
        break;

    case 6:
        nw_layer = s6[factor_b][factor_a];
        break;

}

nsegment = floor(nw_layer/2) + 1;
ncounter = nw_layer % 2 + 1;

nwire = nanode - 16*factor_b + 1;
nstrip = ncathode - 16*factor_a + 1;
```

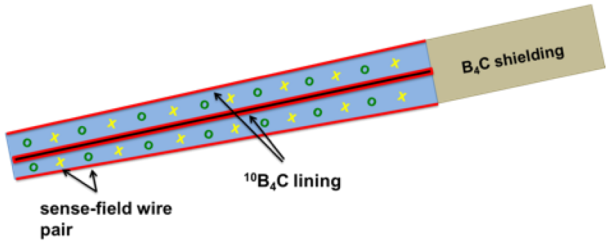
In my analysis code the location of the neutron event in the detector volume is specified by:

- The module (12-deg sector) that was hit
- The SUMO of the module that was hit
- The segment number of the SUMO that was hit
- The counter number of the segment that was hit
- The wire number that collected the charge
- The strip number that collected the charge

Example for SUMO3 which consists of 4 segments and each segment consists of 2 counters.

```
factor_a = floor(ncathode/16);
factor_b = floor(nanode/16);

switch (nsumo){

    case 3:
        nw_layer = s3[factor_b][factor_a];
        break;

    case 4:
        nw_layer = s4[factor_b][factor_a];
        break;

    case 5:
        nw_layer = s5[factor_b][factor_a];
        break;

    case 6:
        nw_layer = s6[factor_b][factor_a];
        break;

}

nsegment = floor(nw_layer/2) + 1;
ncounter = nw_layer % 2 + 1;

nwire = nanode - 16*factor_b + 1;
nstrip = ncathode - 16*factor_a + 1;
```





nsegment = 1,..,4

ncounter = 1,2

nwire = 1,..,16

nstrip = 1,..,16

Time-of-flight information

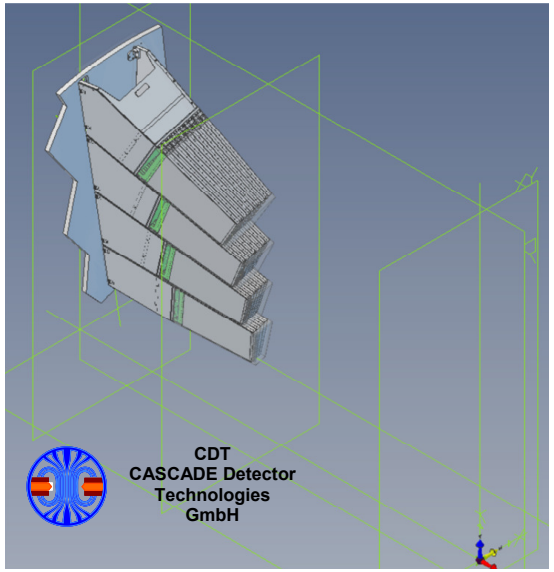
- The neutron time of flight is calculated from the difference of the neutron and chopper timestamps. The time-of-flight obtained from this difference must be corrected in different ways for the runs collected in normal operation mode (full pulse) and Wavelength Frame Multiplication (WFM) mode (the full pulse is split into 6 sub-pulses with the help of a pair of WFM choppers). The time corrections are specific to each instrument (i.e., the time corrections for the V20 data will not be valid for the DREAM instrument.)

x,y,z-coordinates of the detector voxels

- Specifying the location of the neutron event in terms of (module, SUMO, segment, counter, wire and strip number) is not enough to perform the transformation from the time-of-flight space to the wavelength- and d-spacing-spaces (according to the Bragg law). In order to calculate the wavelength of the detected neutron one needs the information on the coordinates (x, y, z) of the interaction point with respect to the sample position. Currently there are two ways to do this for the Jalousie data.

Obtaining the coordinates of the centres of the detector voxels, method 1

- This method is being used by CDT + FZJ



Obtain the coordinates of the SUMO edges from the CAD engineering drawings

Python script to calculate/interpolate the positions of the voxel centres

Table with the coordinates of the voxels

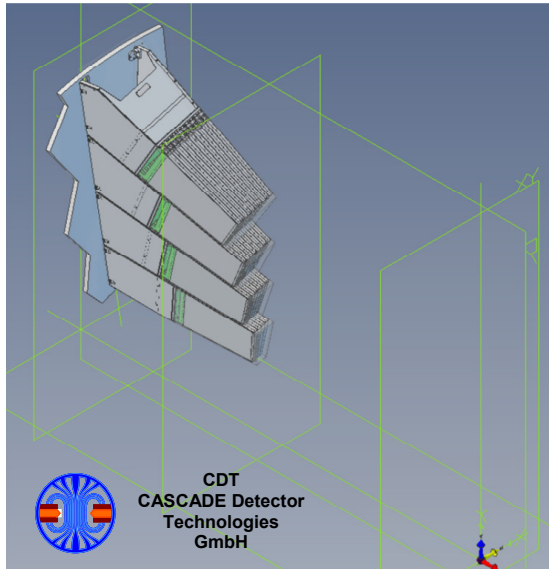
Read into the data-analysis code

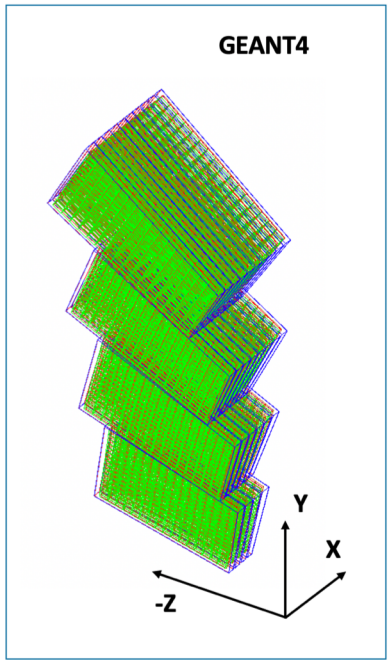
Jülich data analysis

Done by CDT

Done by FZJ

Obtaining the coordinates of the centres of the detector voxels, method 2





Obtain the coordinates of the SUMO edges from the CAD engineering drawings and match them with the positions of the GEANT4 computer models for the SUMOs

→

Dump the positions of the voxel centres in a ROOT tree

Read into the data-analysis code →

Lund data analysis

Done in Lund