Charles University
Prague

Beam Test DST Data Format And Analysis Support Tools

Presented also in DEPFET meeting in Prague, March 2006

Zdeněk Doležal, Peter Kodyš, Pavel Řezníček, Daniel Scheirich

Typical beam test data analysis flow:

Data taking:

- Tune DAQ sequences EXPERTS
- Synchronize triggers EXPERTS

First steps:

- Telescope alignment, track finding/fitting EXPERTS
- DUT intersection prediction EXPERTS
- Data transformation from binary raw data to more readable format (e.g. ROOT) and reprocessing – reduction, possible, but not necessary EXPERTS
- Machine/platform independent access to data for anyone

Basic analysis:

- Common mode, clustering, efficiency, resolution, TB analysis EXPERTS + anyone

Special analyses:

- my hybrid, my energy, my parameters setting, etc. USER (deep programming experience not needed)

Support from experts:

- More user-friendly format (ROOT)
- Good public description of format
- Examples how to access the data and perform analysis

Reports:

- Main report from standard analysis, coordination
- Special sub-reports about specific tasks and problems
- Confirmation reports using independent evaluation algorithms
- Confirmation reports from out of testbeam: from simulation, from older analysis, from labs

Charles University
Prague

Basic preprocessing of data

to prepare them for any physicist users and non-programmers

Special thanks to expert support from Lars & Robert & Jaap

1. precise telescope intersection point using Eta correction algorithm
2. telescope alignment was realized to find shift and rotation for each telescope plane, note that TMinuit class in ROOT was not working precisely if sharp cut for tracks was realized!
3. particle track find for each event where in each axis was exactly one hit in each telescope detected (independent for each axis)
4. Chi2 for track was calculate
5. intersection with detector under test (DUT) was calculate
6. confidence and predicted region for estimation level 95% (2*sigma) was calculate in DUT position

Interstrip position with COG (centre of gravity)

$$x = x_L + P\frac{S_R}{S_L + S_R} = x_L + P\eta$$

Eta for COG – 2 strips

$$\eta = \frac{S_R}{S_L + S_R}$$

Non linear Eta correction

$$x_\eta = x_L + P\frac{\int_0^{\eta_0}\frac{dN}{d\eta}d\eta}{\int_0^1\frac{dN}{d\eta}d\eta} = x_L + Pf(\eta_0)$$

Error of Non linear Eta correction

$$\sigma_{x_\eta} = P\frac{\sum\limits_{Cluster}(ENC_i)}{\sum\limits_{Cluster} S_i}\sqrt{1-2\eta+2\eta^2}\frac{\frac{dN}{d\eta}}{\int_0^1\frac{dN}{d\eta}d\eta}$$

where

$$\eta = f^{-1}(x/P)$$

Alignment algorithm

$$\Delta_x = (x_{meas}+Par[0])\cos(Par[1])+(y_{meas}+Par[2])\sin(Par[3])-x_{ref}$$

$$\Delta_y = -(x_{meas}+Par[0])\sin(Par[1])+(y_{meas}+Par[2])\cos(Par[3])-y_{ref}$$

$$\Delta^2 = \Delta_x^2 + \Delta_y^2$$

$$Minimize = \sum_{Alltracks}(\Delta^2)$$

More details:
http://www-ucjf.troja.mff.cuni.cz/kodys/works/data_analysis/index.html

Charles University
Prague

Generation of DST data files (1)

using version of ROOT: 5.10/00

DST format generated for DEPFET beam test data in January 2006 in DESY, Hamburg, Germany:

For each event:

```
Int_t iEvent;          //order No. of event
Int_t iTBEvent;        //original No. of event in raw data fileInt_t iEvent;
Int_t flag;
// flag: -3: include masking channel or too small energy
// -2: unknown error
// -4: DUT raw data error
// 0: in some plane more events
// 1: in each tel plane one cluster
// 2: in axis 0 each tel plane one cluster
// 1: in axis 1 each tel plane one cluster
Int_t TDC;                  //TDC information
Int_t N_Tel;                //No. of telescopes
Int_t **NClusters;          //No. of clusters (N_Tel telescopes, N_TelPlane axes per telescope)
float ***Channel2;          //channel position from 2 highest strips using COG (max NClusters clusters)
float ***Channel3;          //channel position from 3 highest strips using COG
float ***ChannelEta;        //channel position after Eta algorithm applying
Int_t ***ClusterSize;       //cluster size for neighbored strips > 3*sigmaOfNoise
Int_t ***Signal;            //sum of signal from all strips in cluster
float ***SignalErr;         //error of channel position ChannelEta using theory
                            //            (R. Turchetta, S. Straulino at al.)
float ***Track;             // axis x/y; a0/a1; value/errValue/Chi2
float **DUTIntersection;    // axis x/y; value/TrackEstimation/PredictionRegion
Int_t N_DUT;                //No. of DUTs
Int_t ***DUT;               //full information about DUT response
```

Generation of DST data files (2)

DST format generated for DEPFET beam test data in January 2006 in DESY, Hamburg, Germany:

Next information included to DST:

```
Header of testbeam - identification (text)
DUT Name (text)
DUT Wafer (text)
Who is create DST (text)
DUT type (text)
Preset parameters:
RunNumber
NoOfEvents
Nominal Energy (GeV)
DAQ Date of raw data file creation
DAQ Time of raw data file creation
Bias
No. of telescopes
No. of DUTs
```

Generation of DST data files (3)

DST format generated for DEPFET beam test data in January 2006 in DESY, Hamburg, Germany:

Next information included to DST:

```
About telescopes:

NTelPlanes - No. of planes in telescope
NTelStrips - No. of strips in telescope
NClastMaxFull - No. of clusters in telescope
NTelClusterMax - if No. of clusters in telescope is too big, maximum was set to this value
Pitch[][] - sign include orientation direction compare with plane 0 of first telescope
TelPosition[][] - positions and rotations of all planes include their errors (from alignme
h_TelEtaCor[][] - histograms of ETA correction for each telescope plane
Pedestal[][] - histograms of pedestal for each telescope plane
Threshold[][] - histograms of threshold for each telescope plane
Noise[][] - histograms of noise for each telescope plane

About DUTs:

DUTNRow, DUTNCol - maximal accepted DUT size
h_DUTPosition[] - DUT position - for each DUT
DUTNRow[], DUTNCol[] - DUT maximal matrix size (for 1D sensors (strips) DUTNCol=1) - for e
pitchX[], pitchY[] - DUT cell size matrix or pitch in 1D sensors - for each DUT
```

Charles University
Prague

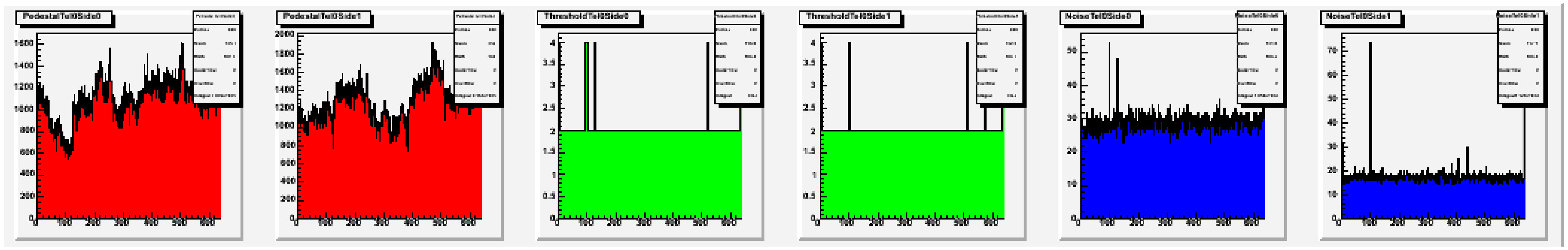
Generation of DST data files (4)

DST format generated for DEPFET beam test data in January 2006 in DESY, Hamburg, Germany:

Histograms included to DST:

```
Telescope Eta Correction [][]
Telescopes: pedestal, threshold, noise [3][][]
For telescope error from eta theory:
    Residuals for all telescopes [][]
    hit map in DUT position [] + DUT Track Estimation [] +
    DUT Prediction Region [] + Chi2 []
For telescope error 12um:
    Residuals for all telescopes [][]
    hit map in DUT position []  + DUT Track Estimation [] +
    DUT Prediction Region [] + Chi2 []
```

pedestal, threshold and noise

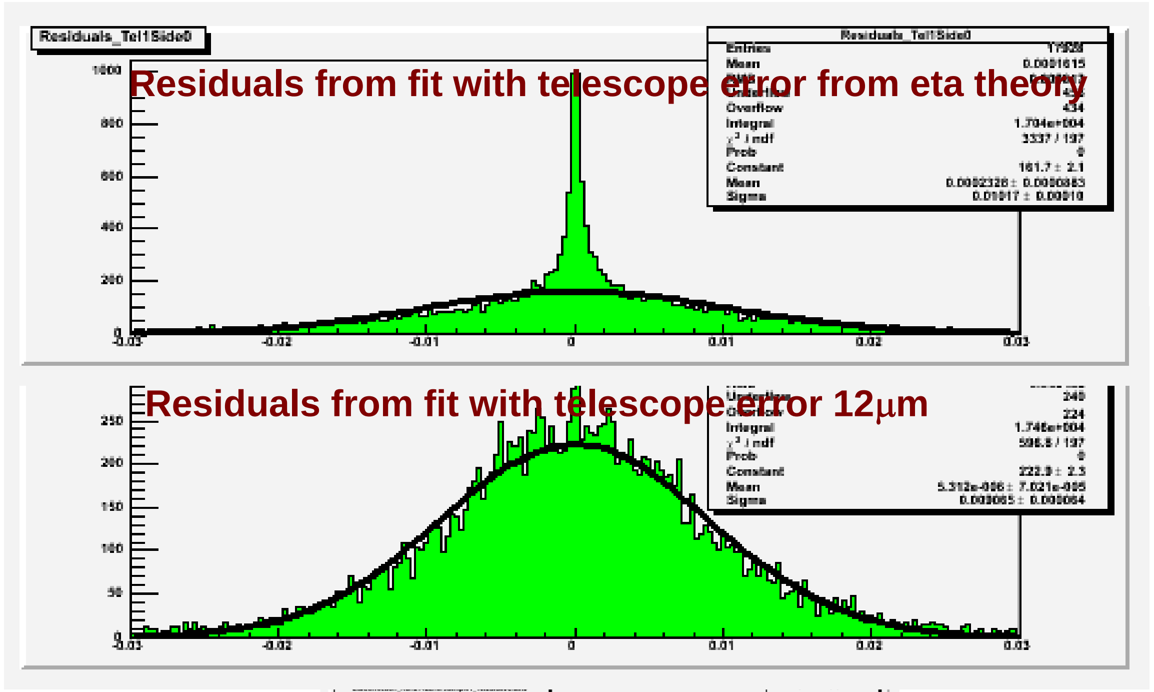


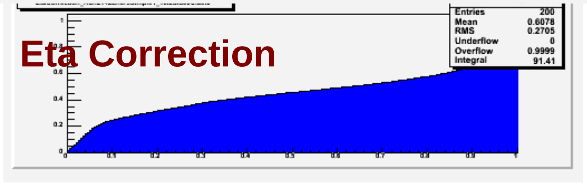
Charles University
Prague

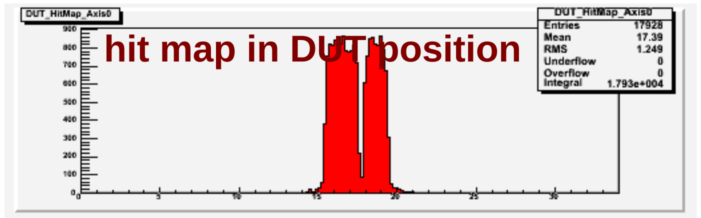
Generation of DST data files (5)

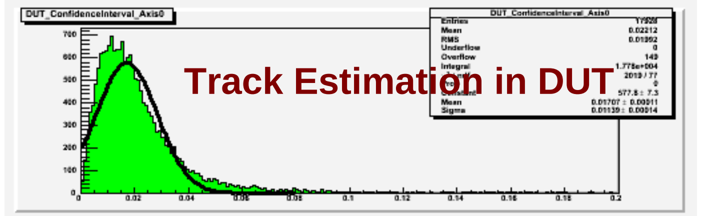
DST format generated for DEPFET beam test data in January 2006 in DESY, Hamburg, Germany:

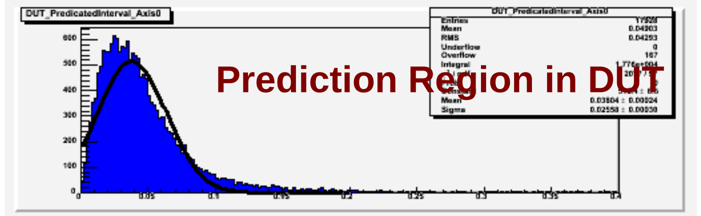
Examples of histograms included to DST:

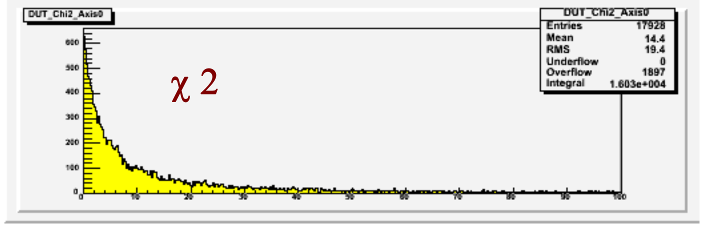












Prepare basic set of macros (1)

Macros:

- **UpgreatDST_depfet.cpp** call .x UpgreatDST_depfet.cpp(RunNo)
functions:
macro shows **content of some event**, **all histograms**, example **how to write loop over some/all events**, how to use **tracking**, how to find **impact point in telescope plane** and make **residuals** include alignment information, how to open DST for **append next histograms** (danger for non-experts)

- **LinearFit.cpp** contains functions to **find track**. This function can be called from the global UpgreatDST_depfet.cpp macro, or separately .x LinearFit.cpp

Charles University
Prague

Prepare basic set of macros (2)

From command line:

Possible is also looking to ROOT DST files straight from command line, there is example:

```
root [0] new TCanvas()      //to create panel by hand for looking plots, not nessesery
(class TCanvas*)0x5a3a168
root [1] TFile *f = new TFile("./dst/DST2116.root")  //open DST file - in ROOT memory
root [2] DST->Draw("Channel2")          //to show histogram of all telescopes - mixture of all (wrong)
root [3] DST->Draw("Channel2[0][1][1]")  //to show histogram of first each hit in tel 1 axis 1
root [4] DST->Draw("Channel2[0][0][0]:Channel2[0][2][0]")  //scatter plot - tel0 and tel2
root [5] DST->Draw("Channel2[0][0][0]:Channel2[0][2][0]","flag==1")  //only if exist track
root [6] DST->Draw("Channel2[0][0][0]:Channel2[0][0][1]","flag==1")  //axis0 vs. axis1 (hitmap)
root [7] DST->Draw("Channel2[0][1][1]","Channel2[0][1][1]<400&Channel2[0][1][1]>200&flag==1")
root [8] DST->Show()  //content of event 0
root [9] DST->Show(10)  //content of event 10
```

Some comments how to work with DST format (1)

Dynamic size of arrays in C++ code:

```
float **ikuk2;  //create array with size iYmax x iXmax
ikuk2 = malloc(sizeof(float *) * iYmax);
for (int i = 0; i < iYmax; i++) {
  ikuk2[i] = malloc(sizeof(float) * iXmax);
}
ikuk2[2][3] = 26;
printf("%f\n",ikuk2[2][3]);
```

Dynamic size of arrays 1D in ROOT code:

```
float *ikuk;>  //create array with size iXmax
ikuk = (float*) new float[iXmax];
ikuk1[3] = 26;
printf("%f\n",ikuk1[3]);
```

Dynamic size of arrays 2D in ROOT code:

```
int **ikuk3;  //create array with size iYmax x iXmax
ikuk3 = (int**) new int*[iYmax];
for (int i = 0; i < iYmax; i++) {
  ikuk3[i] = (int*) new int[iXmax];
}
ikuk3[2][3] = 21;
printf("%i\n",ikuk3[2][3]);
```

Some comments how to work with DST format (2)

Dynamic size of arrays 2D mapping to objects in root data files in ROOT code:

```
sprintf(text,"PitchOfTelescopes");
if(h_DeleteTH1F = (TH1F*) gDirectory->GetList()->FindObject(text)) delete h_DeleteTH1F;
h_Pitch = (TH1F*)fDST->Get(text);
for (int i=0;i < NTel;i++) {
  for (int j=0;j < NTelPlanes;j++) {
    printf("Tel%i Axis%i Pitch %6.3f\n",i,j,h_Pitch->GetBinContent(NTelPlanes*i+j+1));
    Pitch[i][j] = h_Pitch->GetBinContent(NTelPlanes*i+j+1);
  }
}
```

Dynamic size of arrays 3D in ROOT code:

```
double ***fkuk5;  //create array with size iZmax x iYmax x iXmax
fkuk5 = (double***) new double**[iZmax];
for (int k = 0; k < iZmax; k++) {
  fkuk5[k] = (double**) new double*[iYmax];
  for (int i = 0; i < iYmax; i++) {
    fkuk5[k][i] = (double*) new double[iXmax];
    for (int j=0;j < iXmax;j++) fkuk5[k][i][j] = 0;  // preset to 0
  }
}
fkuk5[3][2][3] = 21.01;
printf("3D: %f\n",fkuk5[3][2][3]);
```

Some comments how to work with DST format (3)

In all previouse case raws of matrix are mapped in memory in no compact space and is impossible to share them out of macro (e.g. in submacros), for solving this problem we need to do:

```
// first we create linear 1D matrix to alocate memory space:
float *Channel2Lin = (float*) new float[NTel*NTelPlanes*NTelClusterMax];
// than create 3D matrix:
eventn->Channel2 = (float***) new float**[NTel];
for (int i = 0; i < NTel; i++) {
  eventn->Channel2[i] = (float**) new float*[NTelPlanes];
  for (int j = 0; j < NTelPlanes; j++) {
// finaly we map elements from 3D matrix tu linar space of 1D matrix:
    eventn->Channel2[i][j] = (float*) (Channel2Lin + NTelPlanes*NTelClusterMax*i + NTelClusterMax*j);
  }
}
```

Mapping to Branch in TREE od ROOT file:

```
DST->SetBranchAddress("Channel2",Channel2Lin);
printf("%f %f\n",Channel2Lin[3],Channel2[0][0][3]);
```

Some comments how to work with DST format (4)

How to share dynamic arrays between macros:

```
void KukMain(void) {

  int NDUT = 4;
  int DUTNRow = 5;
  int DUTNCol = 6;

  int *MaskChannelsLin;
  MaskChannelsLin = (int*) new int [NDUT*DUTNRow*DUTNCol];
  int ***MaskChannels;
  MaskChannels = (int ***) new int **[NDUT];
  for (int k=0;k < NDUT;k++) {
    MaskChannels[k] = (int **) new int *[DUTNRow];
    for (int i=0;i < DUTNRow;i++) {
      MaskChannels[k][i] = (int*) (MaskChannelsLin + DUTNRow*DUTNCol*k + DUTNCol*i);
      for (int j=0;j < DUTNCol;j++) MaskChannelsLin[DUTNRow*DUTNCol*k + DUTNCol*i+j] = 4;
    }
  }
  printf("****> %i %i\n",MaskChannels[0][3][4],MaskChannelsLin[24]);
  DUTAnalysis1(MaskChannelsLin);
  DUTAnalysis2(MaskChannels);
}

void DUTAnalysis1(int *MaskChannels) {
  printf("----> %i\n",MaskChannels[0]);
}

void DUTAnalysis2(int ***MaskChannelsOK) {
int ***MaskChannels=MaskChannelsOK;
  printf("----> %i\n",MaskChannels[0][0][0]);
}
```

Conclusions

- DST format was defined
- Complete data from January TB in DESY were transformed to DSTs
- DST are available also on web
- Example macros how to work with DST were prepared
- Account on a Prague linux machine where all DSTs are stored and all macros are working under ROOT can be provided – contact: peter.kodys@mff.cuni.cz

More details:
http://www-ucjf.troja.mff.cuni.cz/kodys/works/data_formats/index.html
http://www-ucjf.troja.mff.cuni.cz/kodys/works/data_analysis/index.html