SVX reconstruction package

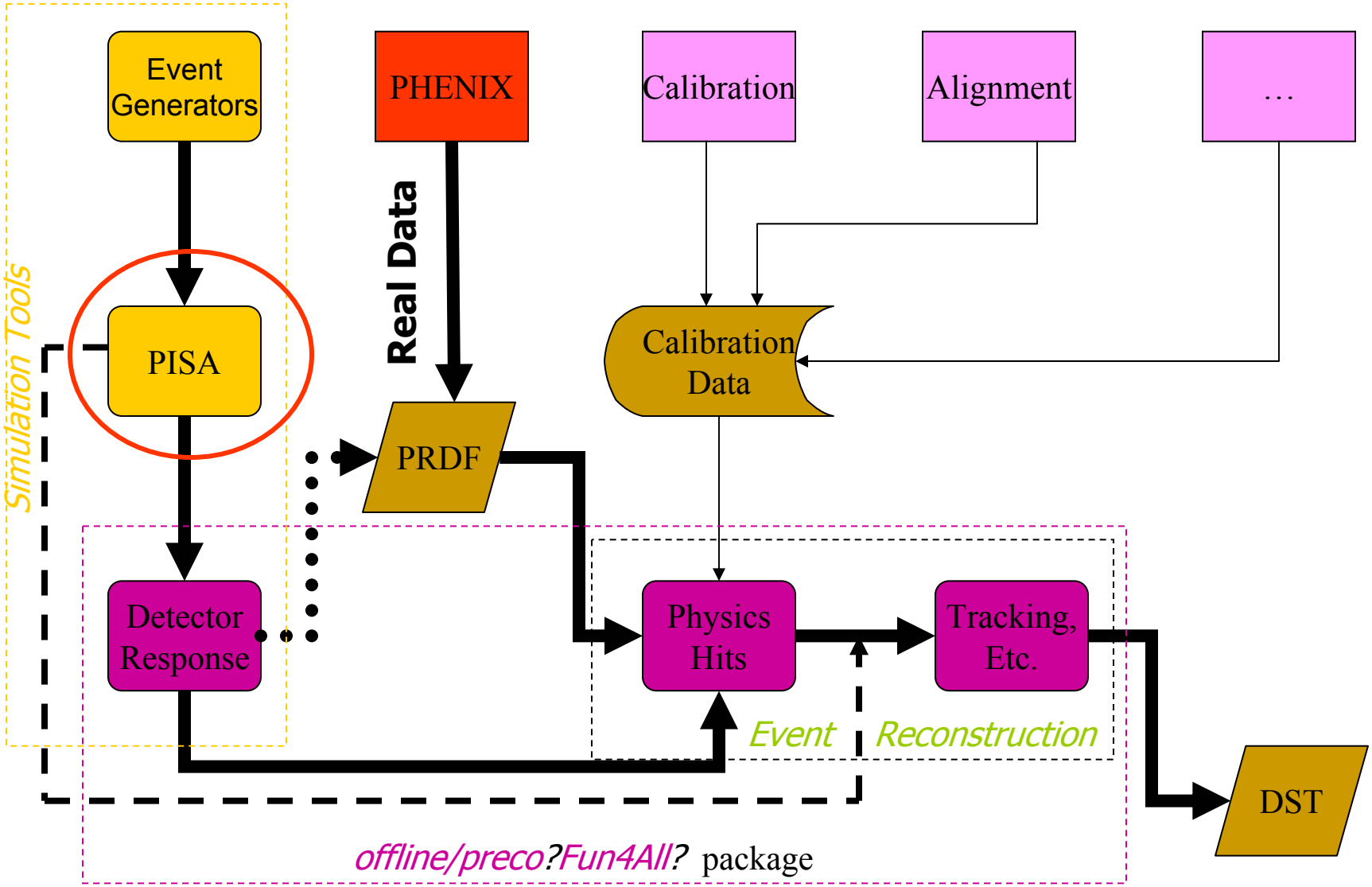
Initial submission and current status

J. T. Mitchell and V. L. Rykov

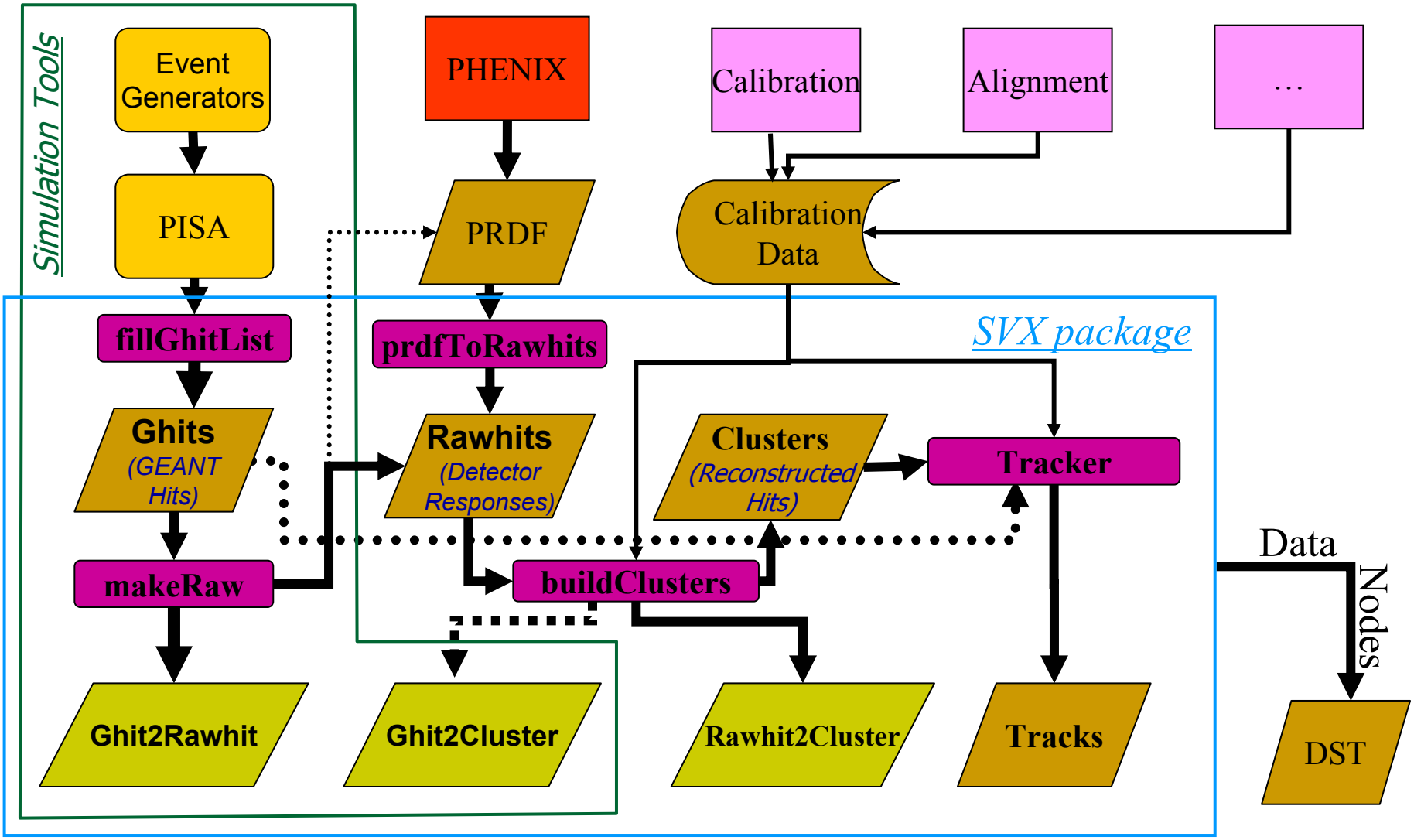
Outlook:

- Overview
- Package structure
- *SvxHit* class and its derivatives
- Hit containers (data nodes)
- Hit relaters
- Containers for hit relaters (data nodes)
- *SvxSensor* class hierarchy
- *SvxReco* class: analysis manager and interface to Fun4All
- Brief user guide
- Illustrations/examples
- Things to do

PHENIX Software Data Flow Chart



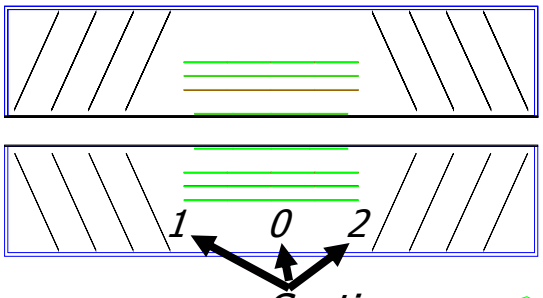
SVX package data flow chart

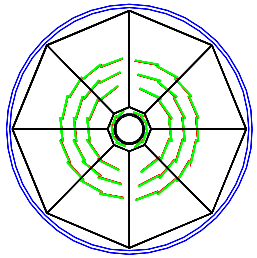


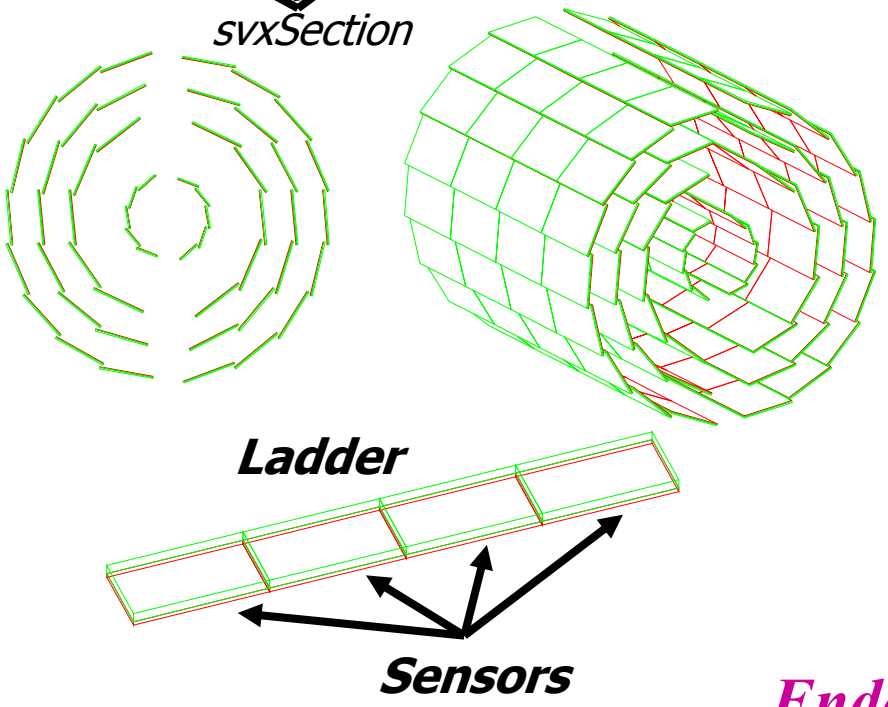
Main classes/hierarchies

- Detector hits - SvxHit : public PHobject (SvxGhit , SvxRawhit, SvxCluster)
- Data nodes: containers for SvxHit objects (SvxGhit, SvxRawhit, SvxCuster)
  SvxGhitList : public PHObject, SvxRawhitList : public PHObject,
  SvxClusterList : public PHObject
- Hit relaters (who originated from whom?)
  SvxGhitRawhit : public PHObject, SvxRawhitCluster : public PHObject,
  SvxGhitCluster : public PHObject
- Data nodes: containers for hit relaters SvxGhitRawhit, SvxRawhitCluster, SvxGhitCluster
  SvxGhitRawhitList : public PHObject, SvxRawhitClusterList : public PHObject,
  SvxGhitClusterList : public PHObject
- "The worker": hit and hit relater producer - SvxSensor : public PHObject
- Reconstruction manager, interface to Fun4All - SvxReco : public SubsysReco
- Track segments (not used in the current version) - SvxSegmentList and SvxSegment

VTX in PISA (SVX detector set)







Cage

- *Rohacell, 5 mm, X_0=543 cm*
- *Filled with Air at T=0°C*

Barrel

- *1 pixel + 3 strip layer*
- *Default parameters:*

Sen. size: 13.9×56.7 mm² (pixels)
34.3×64.6 mm² (strips)

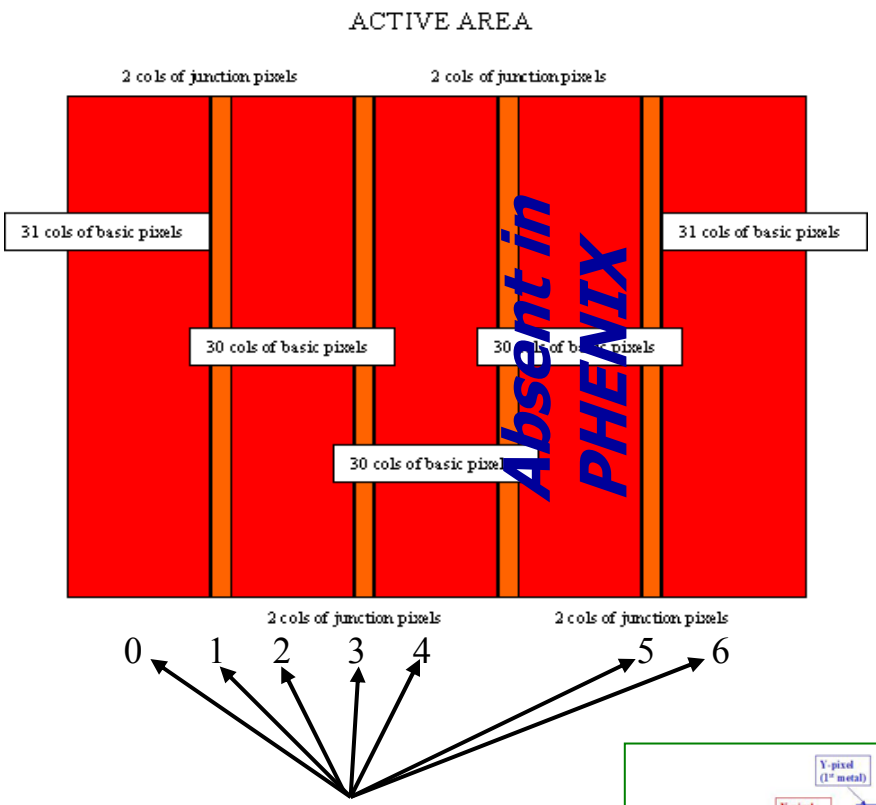
Sect./Layer:	2	2	2	2
Ladd./Sect:	5	5	7	9
Sen./Ladd.:	4	4	4	4
Length., cm:	22.7	25.8	25.8	25.8
R, cm:	2.5	6.0	8.0	10.0
Si. thick, μ:	200	400	400	400
Pasv. X_0, %:	1.0	1.0	1.0	1.0

Endcap: *8 non-segmented polygons of pure Silicon*

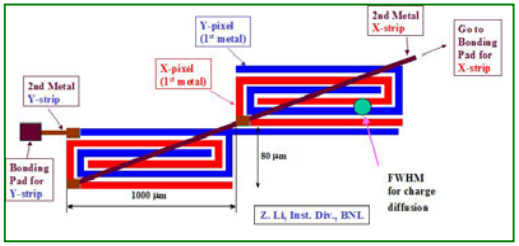
Fake barrel support rings (GFRP) not shown

VTX sensors

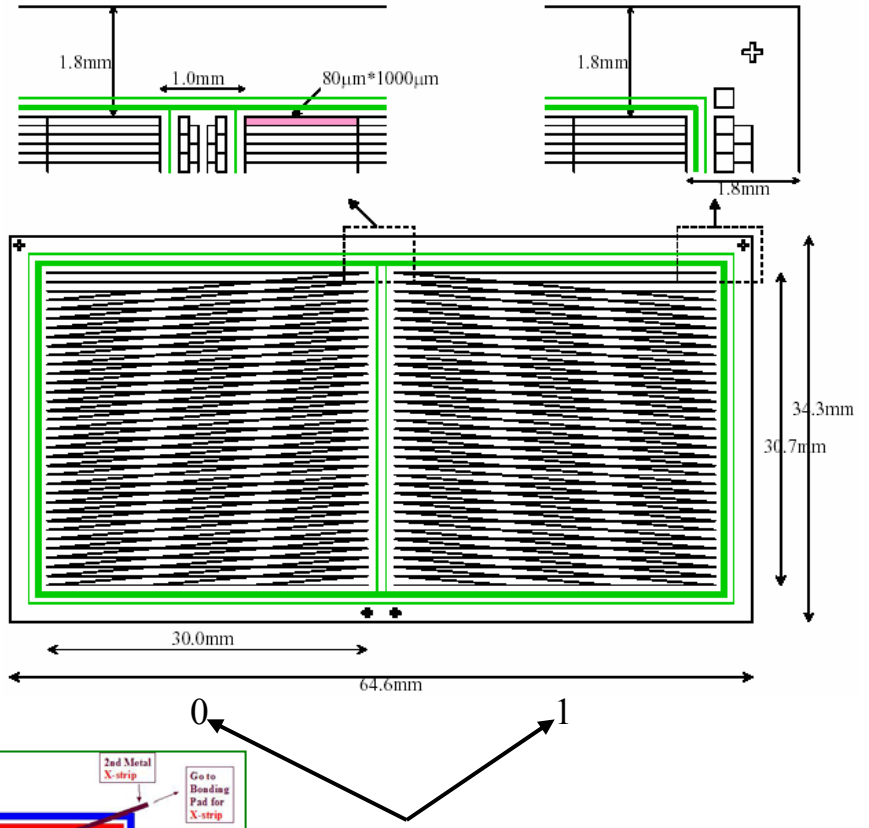
Pixel sensor (ALICE ladder)



sensorSection
sensorReadout = 0

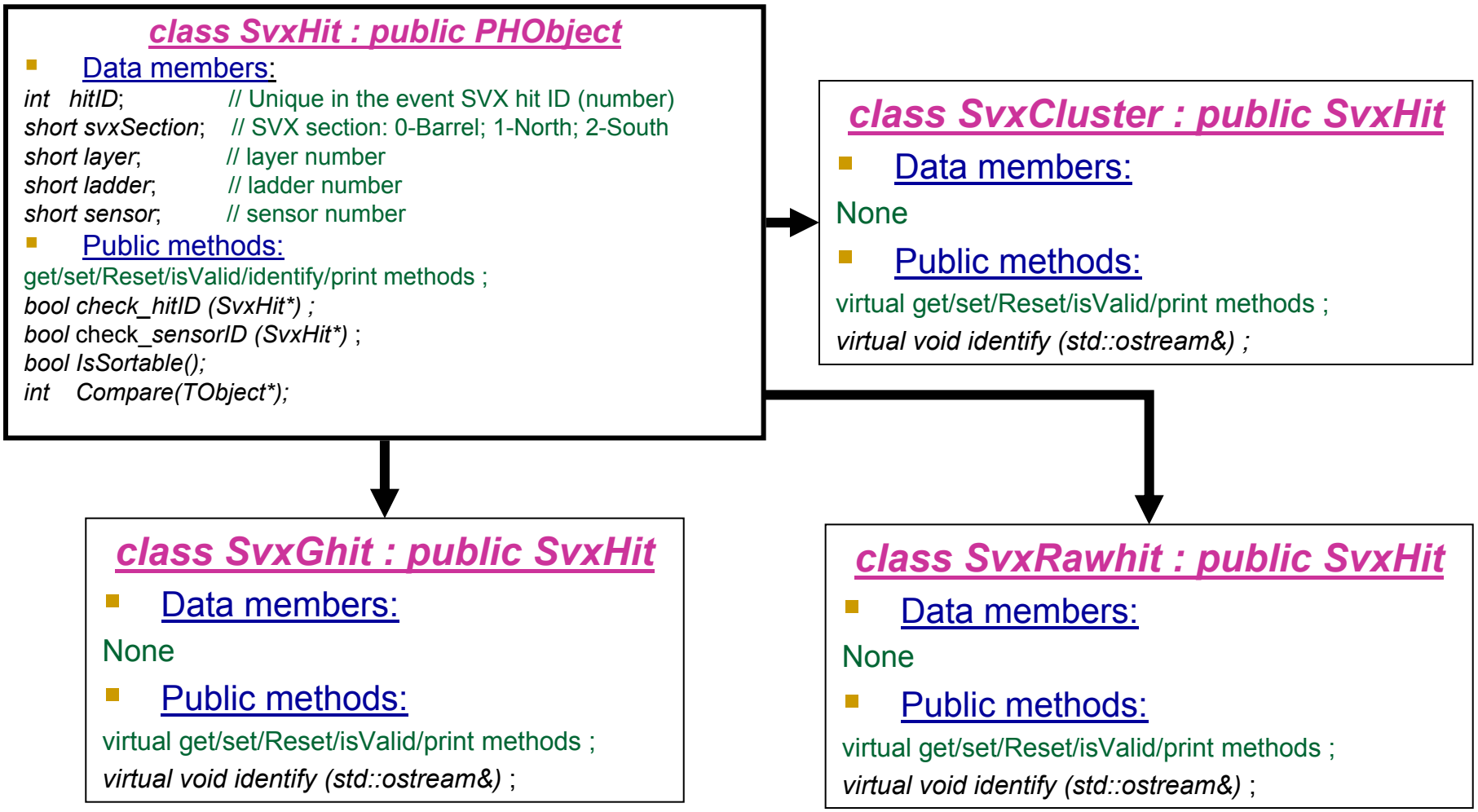


Strip (Stripixel) sensor



sensorSection
sensorReadout = 0 (x) & 1 (u)

SvxHit hierarchy



SvxHit hierarchy (Cont.)

class SvxGhitv1 : public SvxGhit

- Data members:

int isubevent; // PISA subevent number
int nfile; // PISA file number
int mctrack; // Monte Carlo track ID
int idPart; // GEANT Particle ID
int track; // Track number
float dele; // Energy loss, GeV
float tof; // Time-of-flight, s
float xyzglobal[3]; // Average global position, cm
float xyzlocalin[3]; // Entry local coordinates, cm
float xyzlocalout[3]; // Exit local coordinates, cm
float pmomxyz[3]; // Momentum vector, GeV/c

- Public methods:

get/set/Reset/isValid/identify/print methods ;

class SvxRawhitv1 : public SvxRawhit

- Data members:

short sensorSection ; // sensor section
short sensorReadout; // sensor readout
short sensorType; // sensor type
int channel; // channel
int adc; // adc

- Public methods:

get/set/Reset/isValid/identify/print methods ;

class SvxClusterv1 : public SvxCluster

- Data members:

short sensorType; // Sensor type
short edgeflag; // !=0 if at the sensitive area edge
int adc[2]; // adc sum for each readout (x & u)
float xyz_local[3]; // local position
float xyz_global[3]; // global position
float size_xyz_global[3][3]; // global cluster size

- Public methods:

get/set/Reset/isValid/identify/print methods ;

Data nodes: SvxHit containers

class SvxGhitList : public PHObject

- Data members: None
- Public methods: All virtual

class SvxGhitListv1 : public SvxGhitList

- Data members:

unsigned int nGhit; // counter
TClonesArray Ghit; // container

- Public methods:

get/set/Reset/isValid/identify/Sort/print;
SvxGhit addGhit(unsigned int);*
void removeGhit(unsigned int);
int indexOfGhit(…);
int Compress();
int set_TClonesArraySize(unsig. int);

class SvxRawhitList : public PHObject

- Data members: None
- Public methods: All virtual

class SvxRawhitListv1 : public SvxRawhitList

- Data members:

unsigned int nRawhit; // counter
TClonesArray Rawhit; // container

- Public methods:

get/set/Reset/isValid/identify/Sort/print;
SvxGhit addRawhit(unsigned int);*
void removeRawhit(unsigned int);
int indexOfRawhit(…);
int Compress();
int set_TClonesArraySize(unsig. int);

class SvxClusterList : public PHObject

- Data members: None
- Public methods: All virtual

class SvxClusterListv1 : public SvxClusterList

- Data members:

unsigned int nCluster; // counter
TClonesArray Cluster; // container

- Public methods:

get/set/Reset/isValid/identify/Sort/print;
SvxGhit addCluster(unsigned int);*
void removeCluster(unsigned int);
int indexOfCluster(…);
int Compress();
int set_TClonesArraySize(unsig. int);

Hit relaters (who originated from whom?)

class SvxGhitRawhit : public PHObject

- Data members: None
- Public methods: All virtual

class SvxRawhitCluster : public PHObject

- Data members: None
- Public methods: All virtual

class SvxGhitCluster : public PHObject

- Data members: None
- Public methods: All virtual

class SvxGhitRawhitv1 : public SvxGhitRawhit

- Data members:

int ghitID; // SvxGhit ID
int rawhitID; // SvxRawhit ID

- Public methods:

get/set/Reset/isValid/identify/print;
bool IsSortable();
int Compare(TObject);*

class SvxRawhitClusterv1 : public SvxRawhitCluster

- Data members:

int rawhitID; // SvxRawhit ID
int clusterID; // SvxCluster ID

- Public methods:

get/set/Reset/isValid/identify/print;
bool IsSortable();
int Compare(TObject);*

class SvxGhitClusterv1 : public SvxGhitCluster

- Data members:

int ghitID; // SvxGhit ID
int clusterID; // SvxCluster ID

- Public methods:

get/set/Reset/isValid/identify/print;
bool IsSortable();
int Compare(TObject);*

Data nodes: containers for hit relaters

class SvxGhitRawhitList : public PHObject

- Data members: None
- Public methods: All virtual

class SvxRawhitClusterList : public PHObject

- Data members: None
- Public methods: All virtual

class SvxGhitClusterList : public PHObject

- Data members: None
- Public methods: All virtual

class SvxGhitRawhitListv1 : public SvxGhitRawhitList

- Data members:

unsigned int nGhitRawhit; // counter
TClonesArray GhitRawhit; // contnr.

- Public methods:

get/set/Reset/isValid/identify/Sort/print;
SvxGhitRawhit addGhitRawhit(unsigned int);*
void removeGhitRawhit(unsigned int);
int Compress();
int set_TClonesArraySize(unsig. int);
int indexOfGhit(*); indexOfRawhit(*)

class SvxRawhitClusterListv1 : public SvxRawhitClusterList

- Data members:

unsigned int nRawhitCluster; // cntr.
TClonesArray RawhitCluster; // contnr.

- Public methods:

get/set/Reset/isValid/identify/Sort/print;
SvxRawhitCluster addRawhitCluster(unsigned int);*
void removeRawhitCluster(unsig. int);
int Compress();
int set_TClonesArraySize(unsig. int);
int indexOfRawhit(*); indexOfCluster(*)

class SvxGhitClusterListv1 : public SvxGhitClusterList

- Data members:

unsigned int nGhitCluster; // counter
TClonesArray GhitCluster; // contnr.

- Public methods:

get/set/Reset/isValid/identify/Sort/print;
SvxGhitCluster addGhitCluster(unsigned int);*
void removeGhitCluster(unsig. int);
int Compress();
int set_TClonesArraySize(unsig. int);
int indexOfGhit(*); indexOfCluster(*)

SvxPixStructure hierarchy ("bricks" for SvxSensor)

class SvxPixStructure : public PHObject

- Data members: None
- Public methods: All virtual

template class SvxPixStruct : public SvxPixStructure

- Data members:

Static pixel structure parameters;

Container and counter for the "fired" pixels;

Container and counter for the SvxRawhit pointers;

- Public methods:

get/set/add/remove/Reset/isValid/identify/print;

// Finding fired pixels

int firePixels (float, float, float) ; // Finding fired pixels (NO charge sharing)

int firePixels (float, float, float, float, float) ; // Finding fired pixels (WITH charge sharing*)

---

* In the current version of *firePixels(float,float, float, float, float),* charge diffusion is not accounted.

The Workers: SvxSensor hierarchy

class SvxSensor: public PHObject

- Data members:

// Sensor ID in the SVX
short svxSection; // SVX section
short layer; // layer in svxSection
short ladder; // ladder in the layer
short sensor; // sensor in the ladder
// Sensor global position
float rotMatrix[3][3] ; // Rot. matrix
float transVector[3]; // Trans. vector

- Public methods:

get/set/Reset/isValid/identify/print;
local<->global transformations;
A bunch of other virtual methods;
bool checkID (SvxHit) ;*

template class SvxStripixel : public SvxSensor

- Data members:

Static sensor parameters;
// Pixel structures the sensor is build of
SvxPixStructure senSec[nSec][nRd] ;

- Public methods:

get/set/Reset/isValid/identify/print;
// SvxRawhit maker
int makeRawhits (SvxGhitList ,*
SvxRawhitList ,*
SvxGhitRawhitList) ;**

---

* In the current *makeRawhit(…)* version, the charge between x- and u-strips is always evenly divided: 50% to 50%. Charge diffusion is not accounted. No noise is added.

class SvxPixel1v1 : public SvxStripixel<1>
- Public methods:
void identify (std::ostream&) ;
*int findClusters (…);***

class SvxStrip11v1 : public SvxStripixel<11>
- Public methods:
void identify (std::ostream&) ;
*int findClusters (…);***

class SvxStrip21v1 : public SvxStripixel<21>
- Public methods:
void identify (std::ostream&) ;
*int findClusters (…);***

class SvxStrip22v1 : public SvxStripixel<22>
- Public methods:
void identify (std::ostream&) ;
*int findClusters (…);***

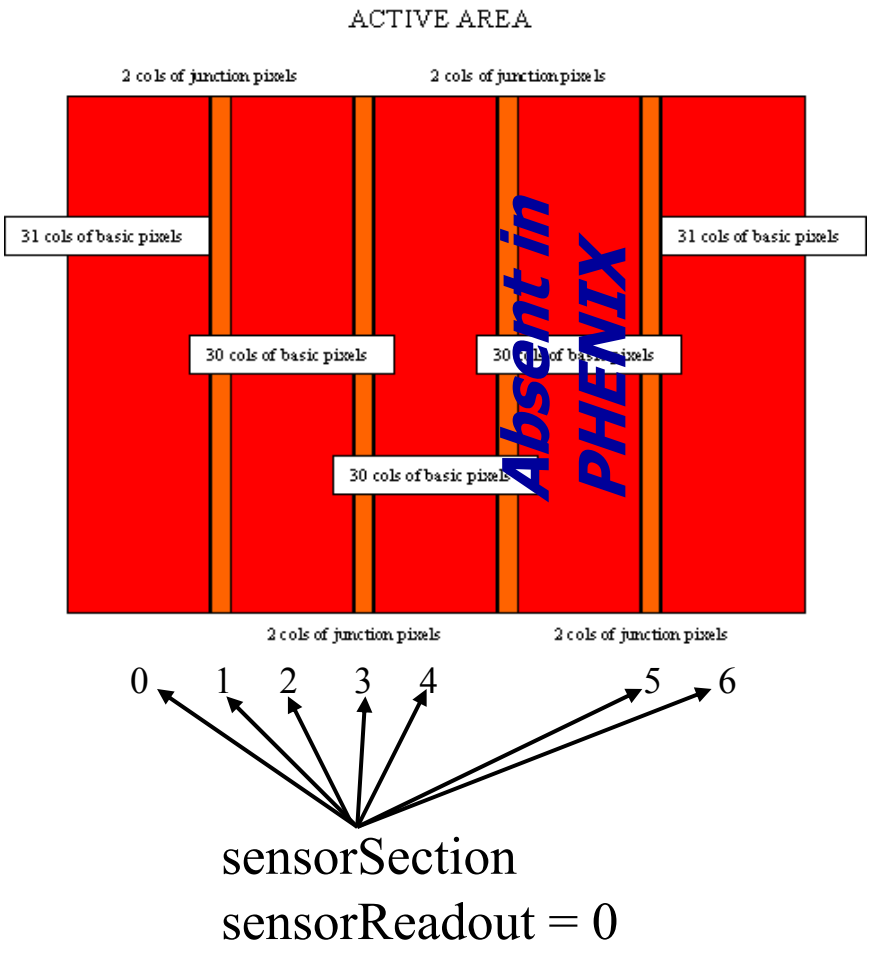
Here could be placed dozens more*** …

** In the current *findCluster(…)* version, *clusters are single pixels and strip crossings; still no combining of adjacent "pixels" together.*

*** *See https://www.phenix.bnl.gov/WWW/p/draft/vrykov/OverlappingWiredPads_Rykov20020605.ppt & .pdf*

Sensor types:

SvxPixel1v1 : public SvxStripixel<1> , sensorType = 1



- **Parameters for SvxStripixel<1>**

nSection = 7 ;
nReadout = 1 ;
secTYPE[7] = {2, 3, 1, 3, 1, 3, 2} ;
secXpos[7] = {0., 0., 0., 0., 0., 0., 0.} ;
secZpos[7] = {-2.12125, -1.4, -0.7, 0., 0.7, 1.4, 2.12125} ;

- **Parameters for SvxPixStruct<1>**

xPitch = 50 mkm, nXpitch = 256 ;
zPitch = 425 mkm, nZpitch = 30 ;

- **Parameters for SvxPixStruct<2>**

xPitch = 50 mkm, nXpitch = 256 ;
zPitch = 425 mkm, nZpitch = 30 ;

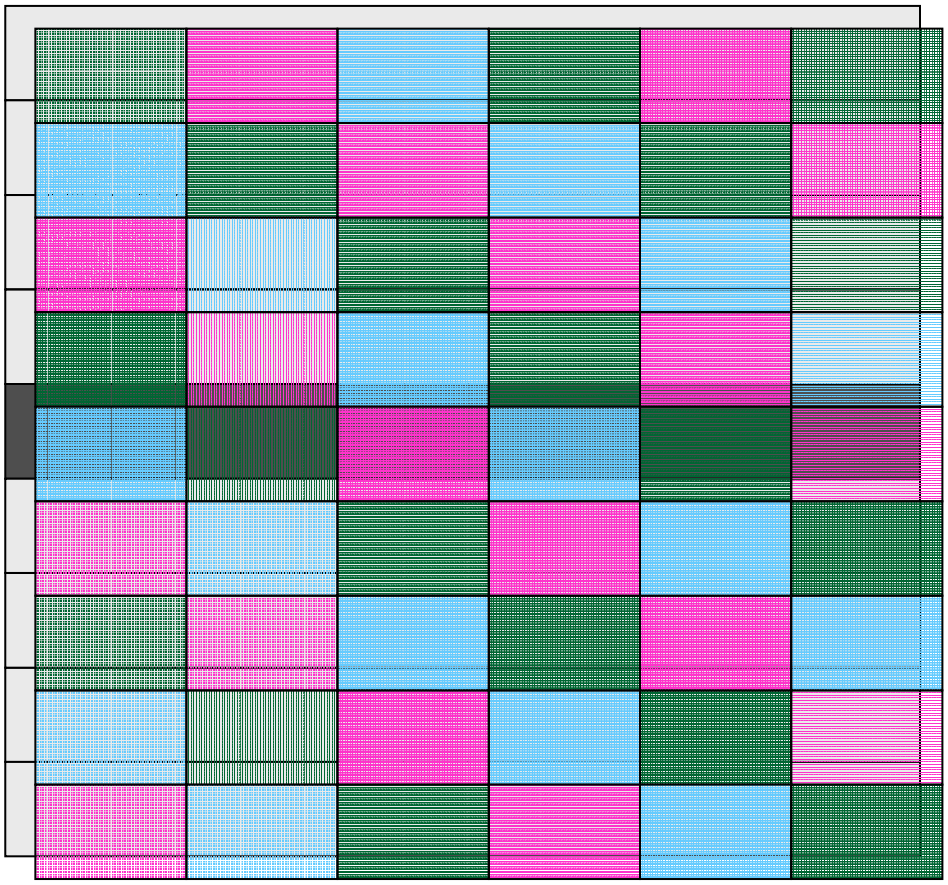
- ***Parameters for SvxPixStruct<3>***

xPitch = 50 mkm, nXpitch = 256 ;
zPitch = 625 mkm, nZpitch = 2 ;

- **Total number of channels**: ***32,768***

Sensor types:

SvxStrip11v1 : public SvxStripixel<11> , sensorType = 11



- **Parameters for SvxStripixel<11>**

nSection = 2 ;
nReadout = 2 ;
secTYPE[2][2] = {{10, 10}, {10, 10}} ;
secXpos[2][2] = {{0.0004, -0.0004}, {0.0004, -0.0004}} ;
secZpos[2][2] = {{-1.54945, -1.54865}, {1.54945, 1.54865}} ;

- **Pixels to channel relations**

x-channel = x-row ;
u-channel = x-row + z-column ;

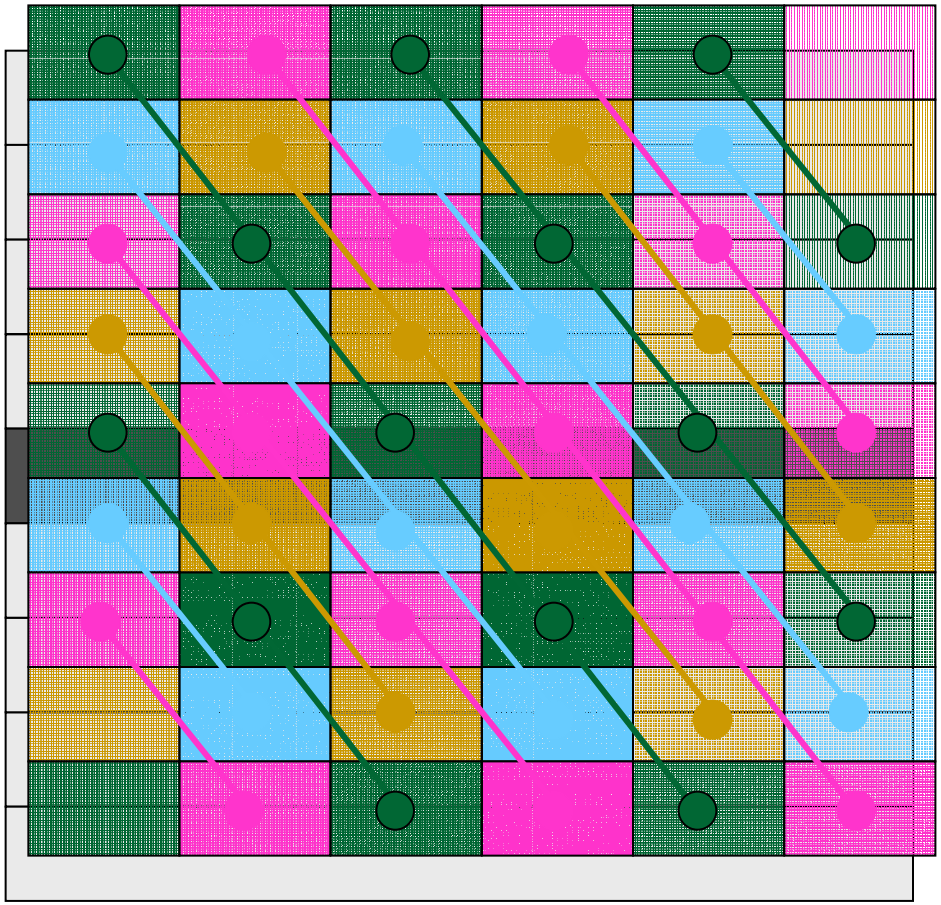
- Number of x-channels*: 768*
- Number of u-channels*: 768 (826)*
- **Total number of channels***: **1536 (1594)***

- **Parameters for SvxPixStruct<10>**

xPitch = 80 mkm, nXpitch = 384 ;
zPitch = 1000 mkm, nZpitch = 30 ;

Sensor types:

*SvxStrip2*v1 : public SvxStripixel<2*>* , sensorType = 2*



- **Parameters for SvxStripixel<2*>**

nSection = 2 ;

nReadout = 2 ;

secZpos[2][2] = {{-1.54945, -1.54865}, {1.54945, 1.54865}} ;

| secXpos[i][0] – secXpoz[i][1] | = xPitch/2 ;

- **Pixels to channel relations**

x-channel = x-row ; u-channel = x-row + 2×z-column ;

- **Parameters for SvxStripixel<21>**

secTYPE[2][2] = {{10, 10}, {10, 10}} ;

secXpos[2][2] = {{-0.0020, 0.0020}, {-0.0020, 0.0020}} ;

Number of x-channels: *768*

Number of u-channels: *768 (884)*

Total number of channels: ***1536 (1652)***

- **Parameters for SvxStripixel<22>**

secTYPE[2][2] = {{20, 20}, {20, 20}} ;

secXpos[2][2] = {{-0.0009, 0.0071}, {-0.0009, 0.0071}} ;

Number of x-channels: *384*

Number of u-channels: *384 (500)*

Total number of channels: ***768 (884)***

- **Parameters for SvxPixStruct<20>**

xPitch = 160 mkm, nXpitch = 192 ;

zPitch = 1000 mkm, nZpitch = 30 ;

The Boss: SvxReco class
The manager and interface to Fun4All

class SvxReco : public SubsysReco

- **Some data members:**

```
// Data nodes
SvxGhitList          *d_ghit;          // GEANT hits
SvxRawhitList        *d_rawhit;        // SVX raw hits
SvxClusterList       *d_cluster;       // SVX clusters
SvxGhitRawhitList    *d_ghit2rawhit;   // SVX ghit<->rawhit relater
SvxRawhitClusterList *d_rawhit2cluster; // SVX rawhit<->cluster relater
SvxGhitClusterList   *d_ghit2cluster;  // SVX ghit<->cluster relater
// SvxSensor objects: barrel only
SvxSensor barSensor[nBarLayers][SVXMAXBARLADDER][SVXMAXBARSENSOR];
```

Input:

PISAEvent root file (GEANT hits)

Output:

svxRecoOut.root file (diagnostic TTree *svx)

svxHistogramOut.root file (diagnostic histograms)

- **Some public methods:**

```
// Interface to Fun4All
int Init(PHCompositeNode *topNode);
int InitRun(PHCompositeNode *topNode);
int process_event(PHCompositeNode *topNode);
int ResetEvent(PHCompositeNode *topNode);
int Reset(PHCompositeNode *topNode);
int End(PHCompositeNode *topNode);
```

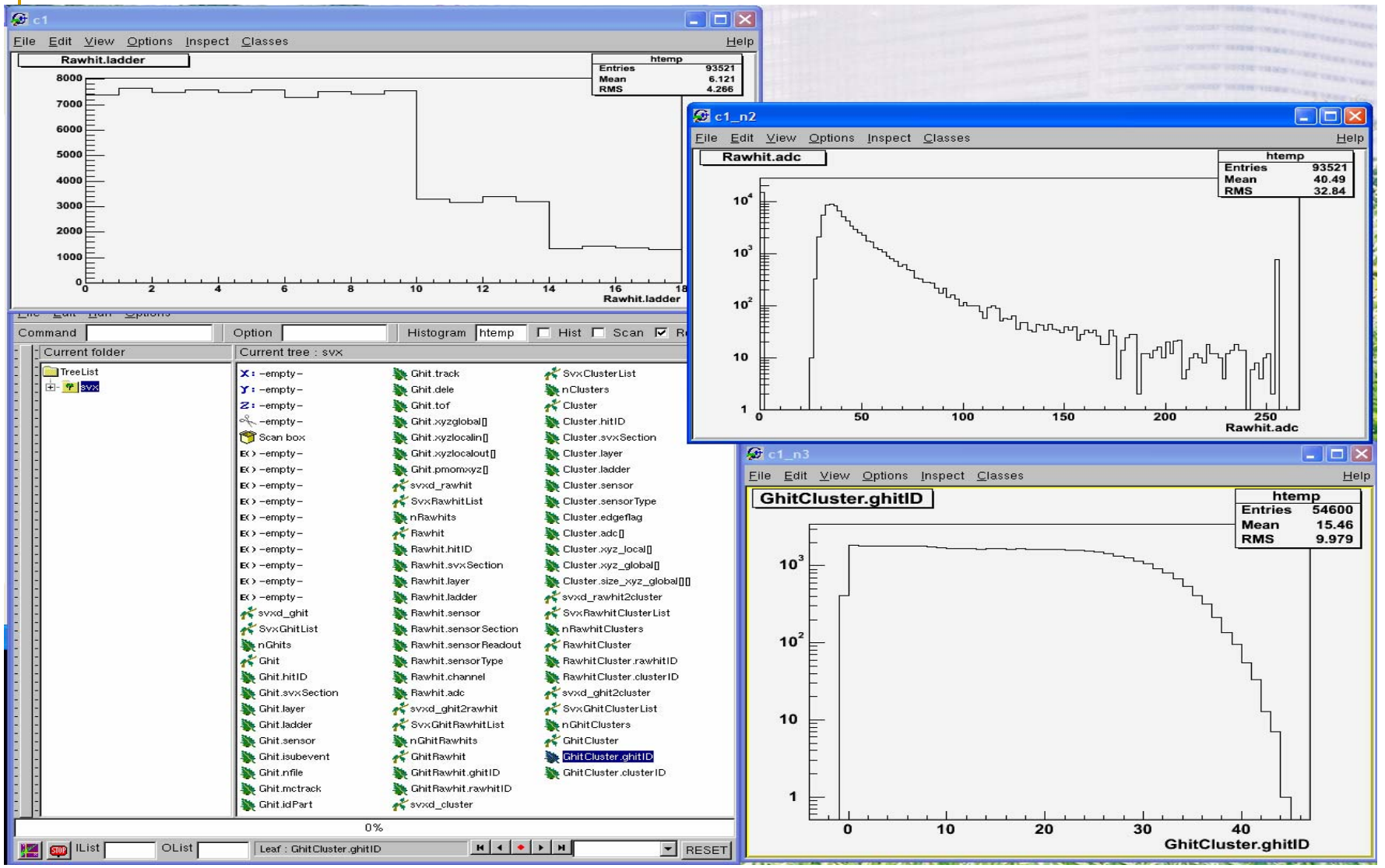
```
// Analysis methods
int fillGhitList();      // Fill GEANT hits, d_ghit, from the PISA tree
int fillRawhitList();    // Fill d_rawhit & d_ghit2rawhit from d_ghit
int fillClusterList();   // Fill SVX clusters, using raw hits
int fillHistograms();    // Fill diagnostic histograms
```

Running the SVX package: excerpts from README file

- ***Location in the PHENIX CVS repository:***
  1. *offline/packages/svx - Contains Silicon vertex reconstruction code*
  2. *offline/packages/svx/wrk - Contains sample macros for running analysis*

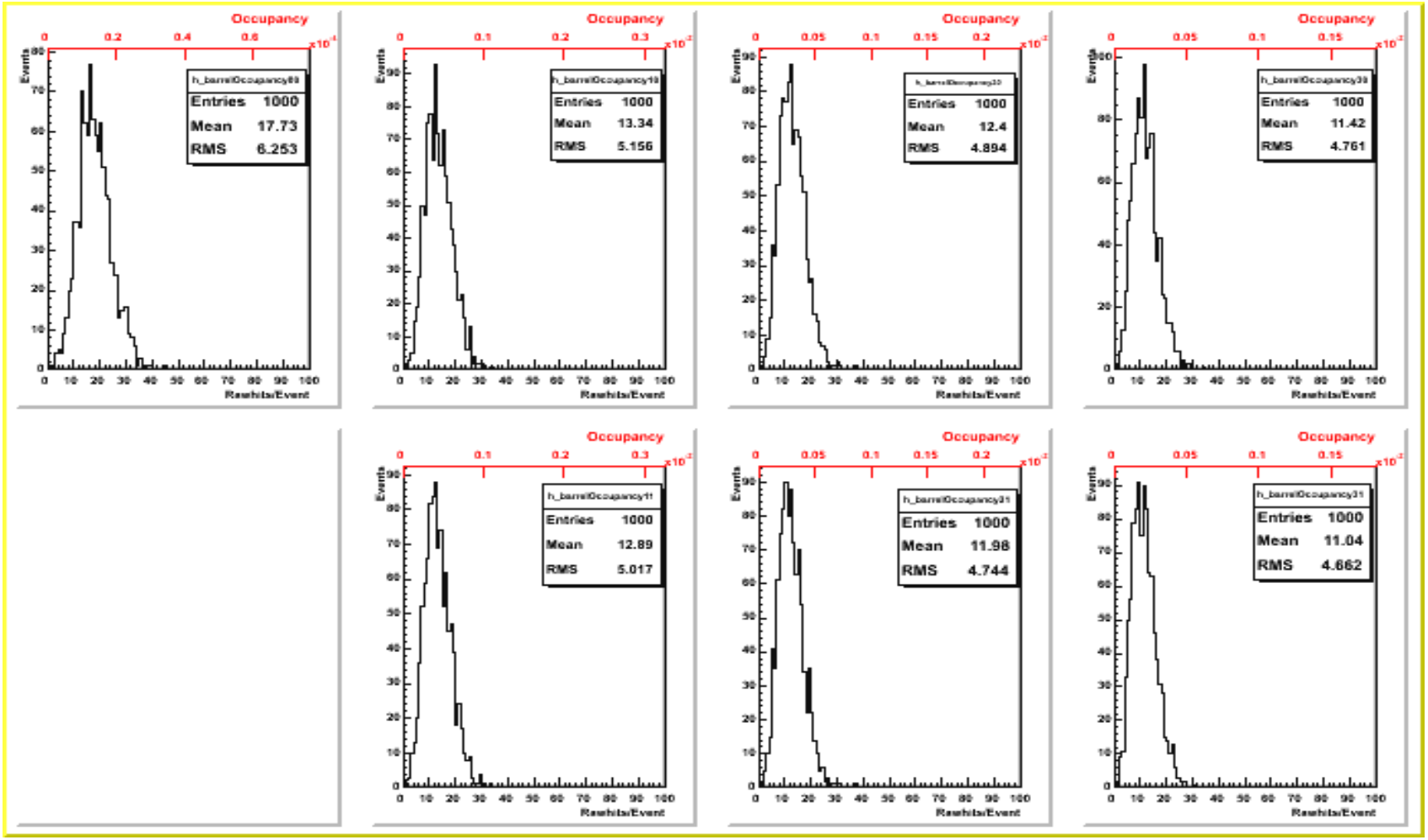
- ***How to run:***
  1. *Check out and build SVX package using the usual PHENIX procedure.*
  2. *Point your LD_LIBRARY_PATH environment variable to the libraries you have built.*
  3. *Go to a PISA work directory. Run PISA with SVX detector "ON" and create PISAEvent.root & svxPISA.par files.*
  4. *Go to the analysis work directory (e.g. offline/packages/svx/wrk). Link (or copy) PISAEvent.root & svxPISA.par files into this directory. Start root and type ".x reco_svx.C".This macro will process 5 events and fill the diagnostic TTree and few histograms. TTree is stored in the svxRecoOut.root file, and histograms are stored in the svxHistogramOut.root file.*
  5. *Start root, connect svxHistogramOut.root file and draw some histograms, using "draw_svx_*.C" macros.*
  6. *Modify (locally!!!) "analyze_svx.C" (and/or everything else) and use it for running your analysis with the input from the svxRecoOut.root file.*

Examples: *TTree *svx* in the output file *svxRecoOut.root*



Examples: **Occupancy**: *PYTHIA: pp@200 GeV, P_T^{jet}>10 GeV/c*

No charge sharing between pixels and strips



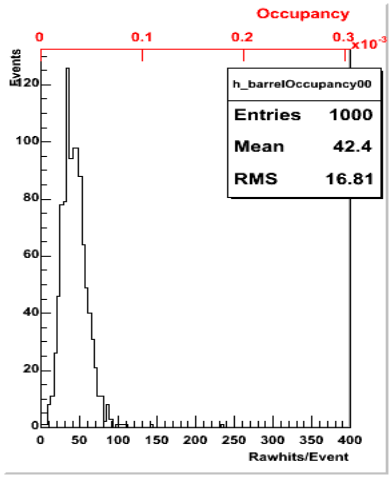
x

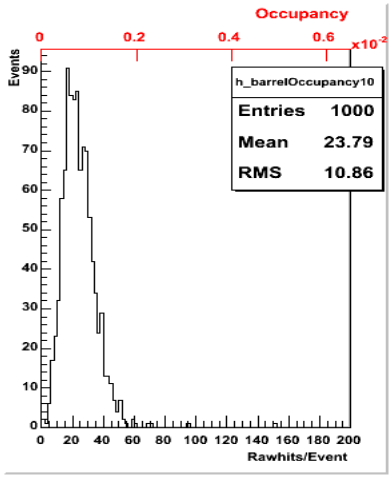
u

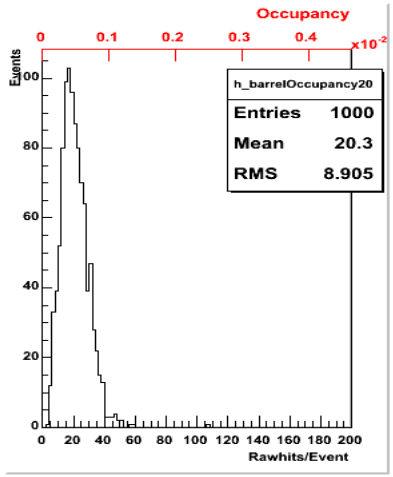
Layer 0: *SvxPixel1* Layer 1: *SvxStrip11* Layer 2: *SvxStrip11* Layer 3: *SvxStrip11*

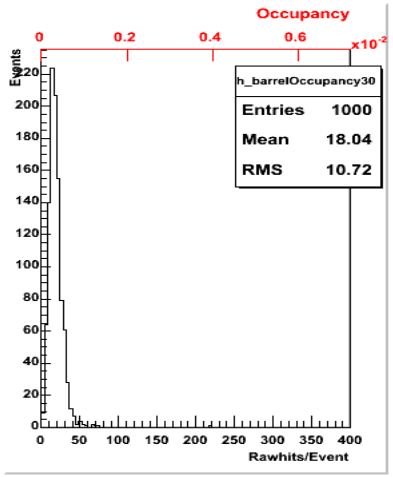
Examples: **Occupancy:** *PYTHIA: pp@200 GeV, P_T^{jet}>10 GeV/c*

With charge sharing between pixels and strips



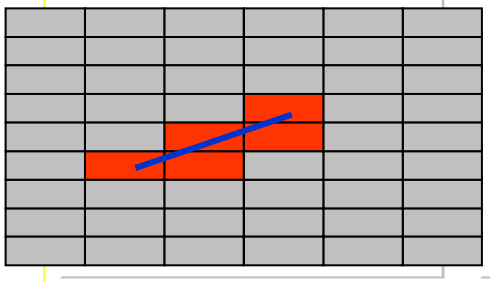


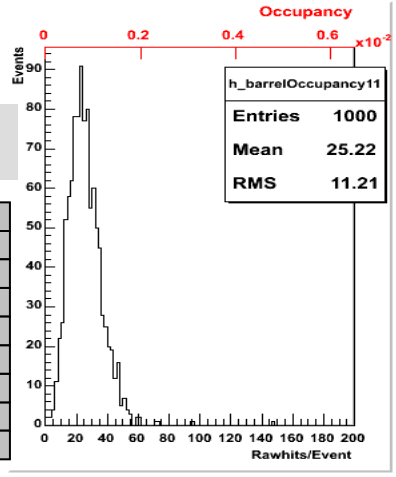


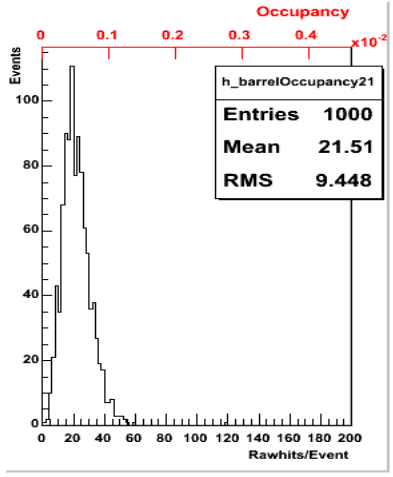


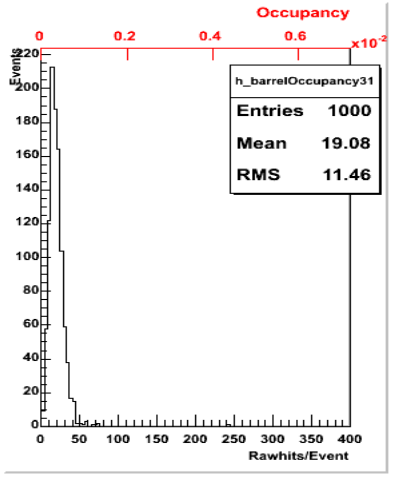
x

Charge sharing scheme:







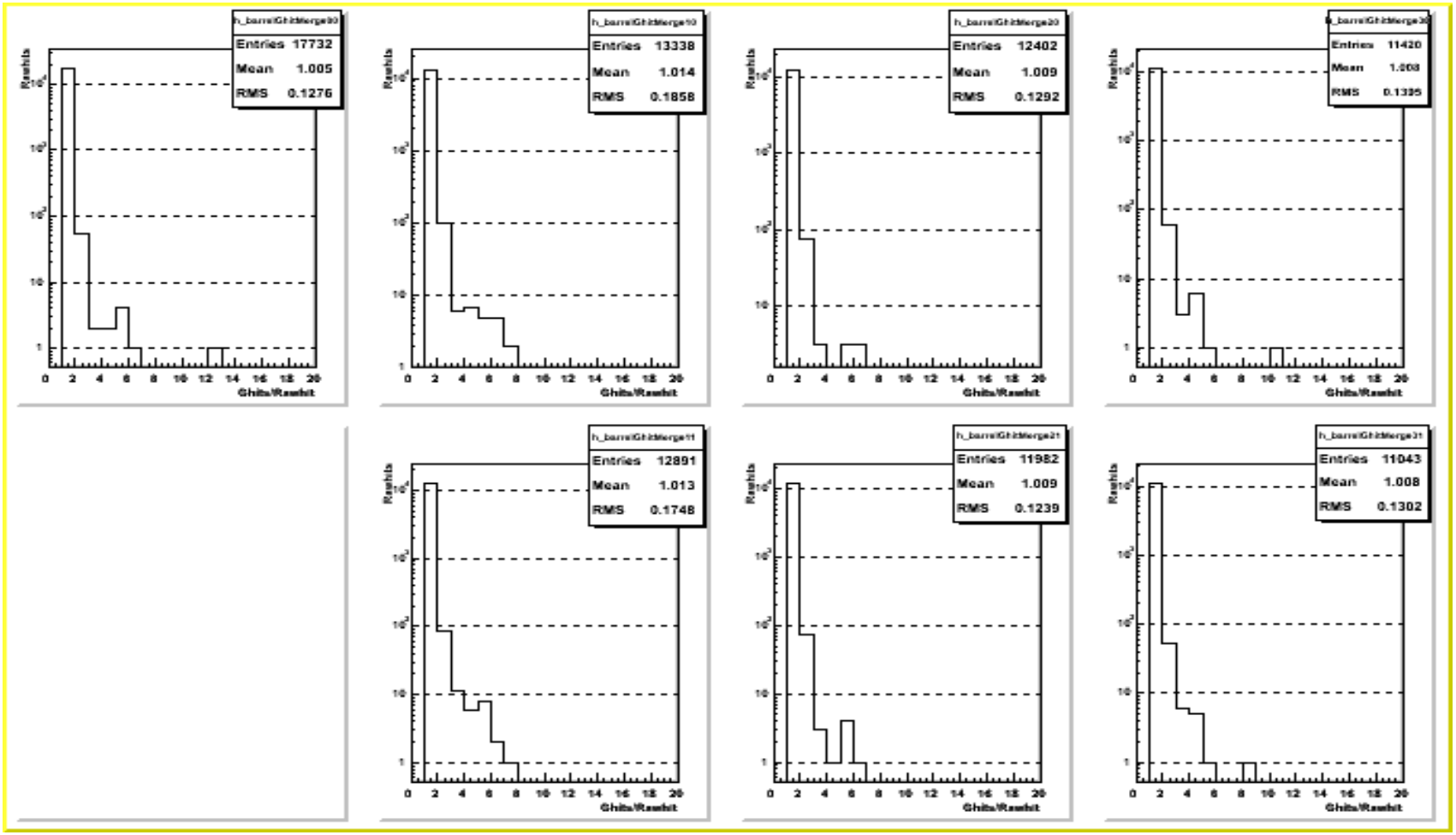


u

Layer 0: *SvxPixel1* Layer 1: *SvxStrip11* Layer 2: *SvxStrip11* Layer 3: *SvxStrip11*

Examples: **Ghit merge:** *PYTHIA: pp@200 GeV, P_T^{jet}>10 GeV/c*

No charge sharing between pixels and strips



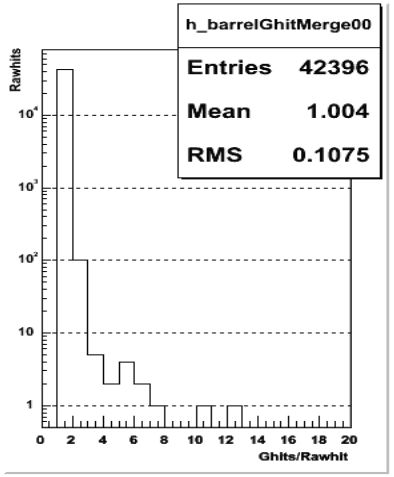
x

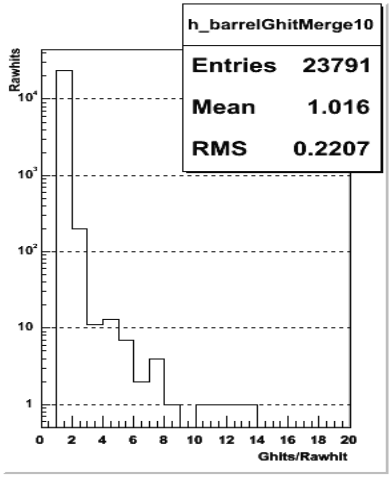
u

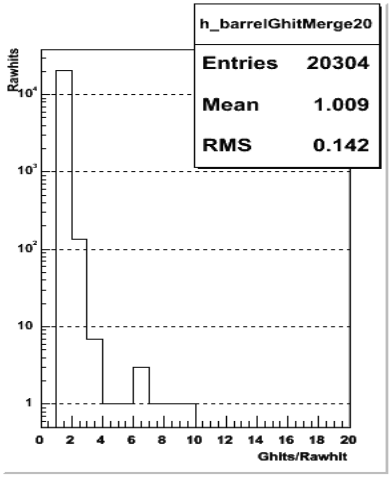
Layer 0: *SvxPixel1* Layer 1: *SvxStrip11* Layer 2: *SvxStrip11* Layer 3: *SvxStrip11*

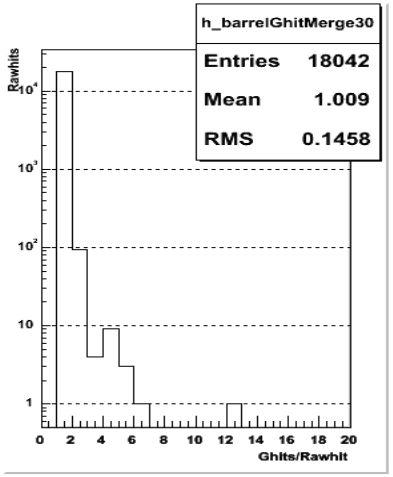
Examples: **Ghit merge:** *PYTHIA: pp@200 GeV, P_T^{jet}>10 GeV/c*

With charge sharing between pixels and strips



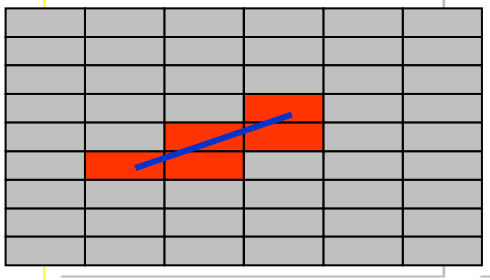


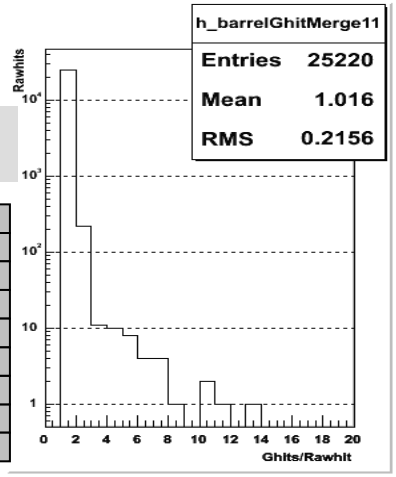


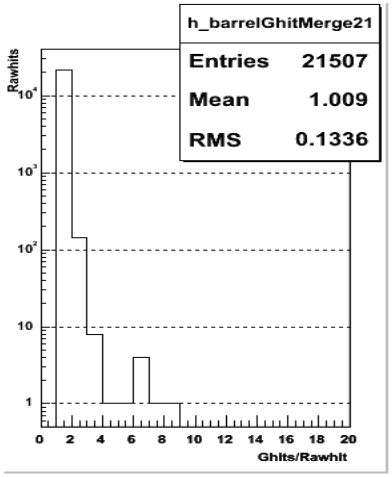


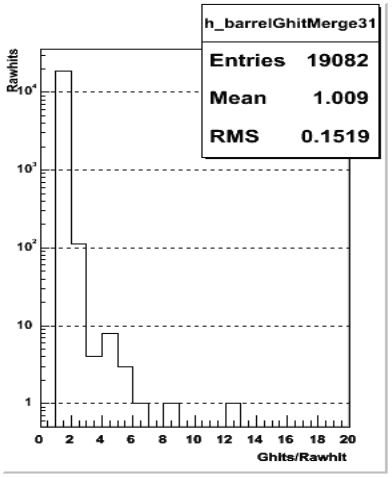
x

Charge sharing scheme:





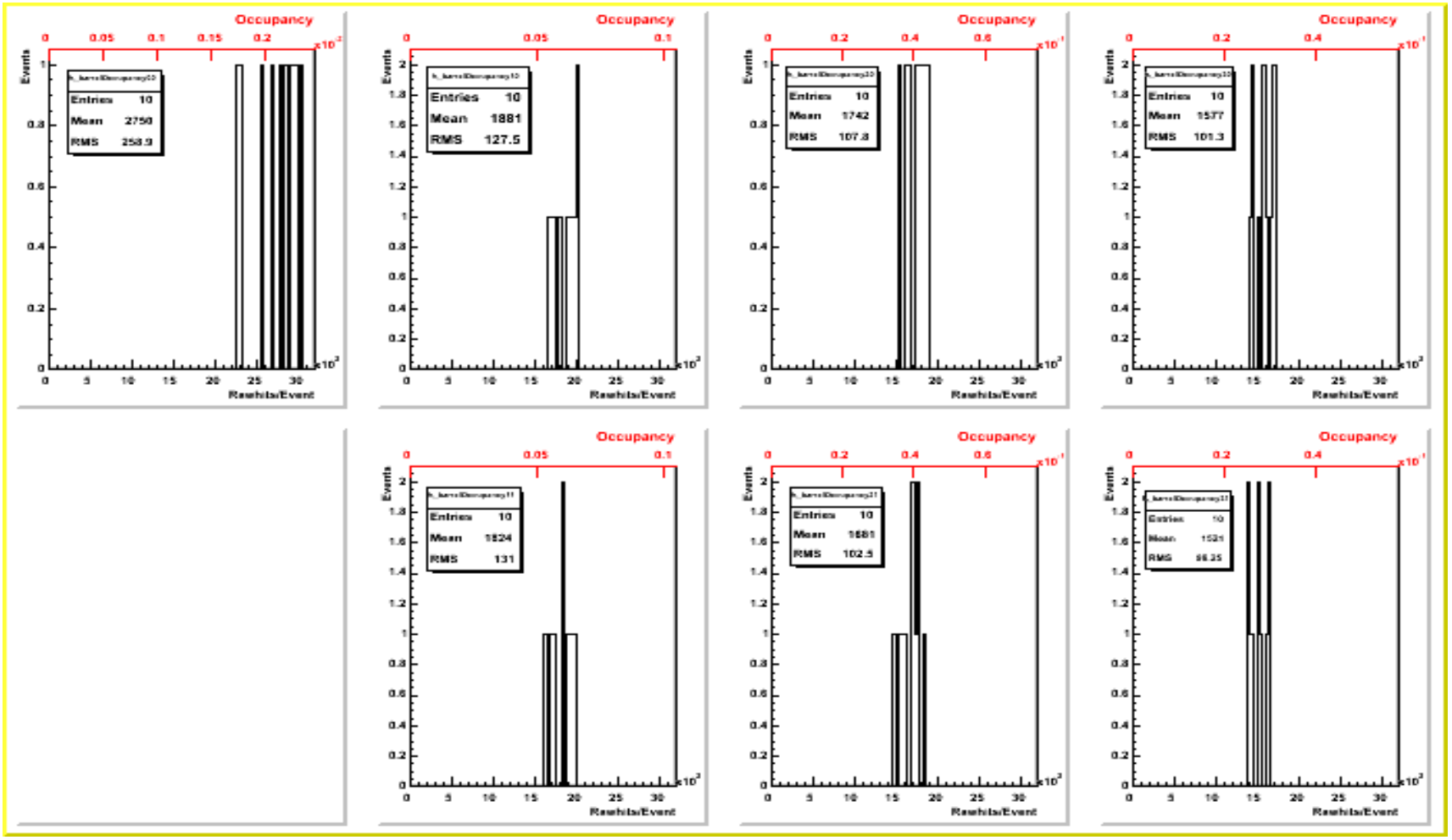




u

Layer 0: *SvxPixel1* Layer 1: *SvxStrip11* Layer 2: *SvxStrip11* Layer 3: *SvxStrip11*

Examples: **Occupancy:** *HIJING: Central AuAu@200 GeV$_{NN}$:* No charge sharing between pixels and strips

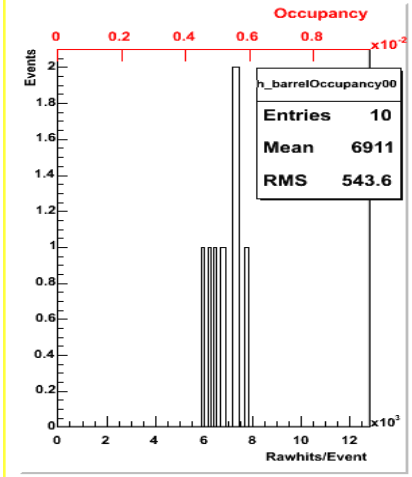


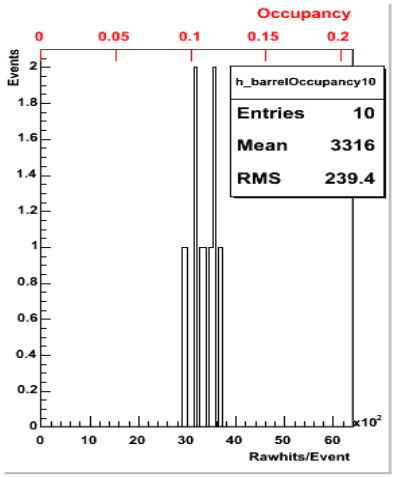
x

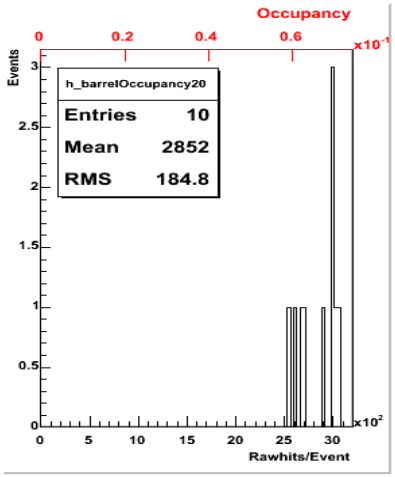
u

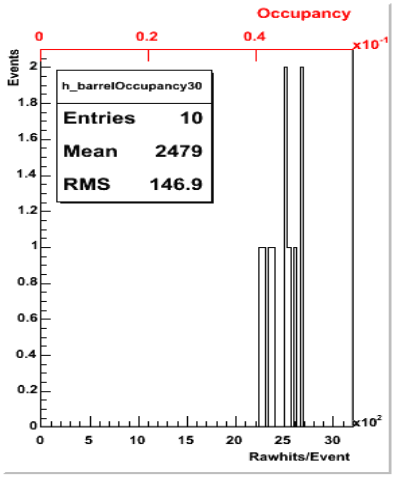
Layer 0: *SvxPixel1* Layer 1: *SvxStrip11* Layer 2: *SvxStrip11* Layer 3: *SvxStrip11*

Examples: **Occupancy**: *HIJING: Central AuAu@200 GeV$_{NN}$*: With charge sharing between pixels and strips



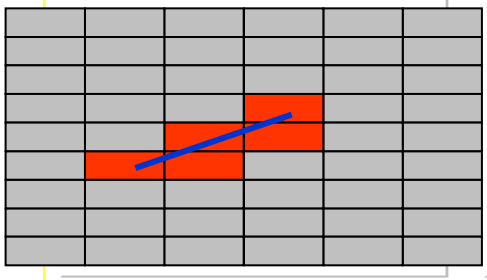


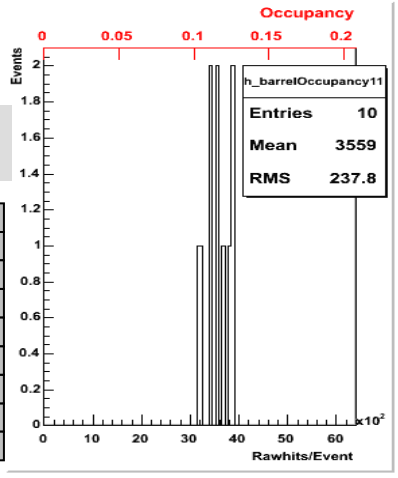


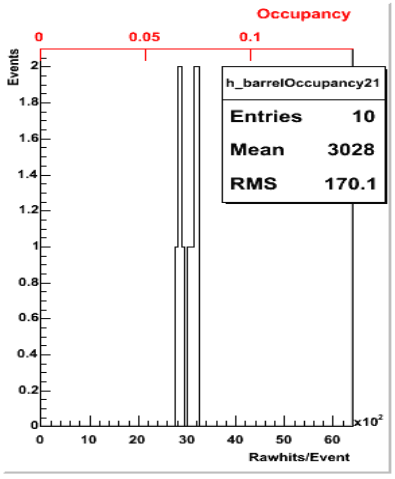


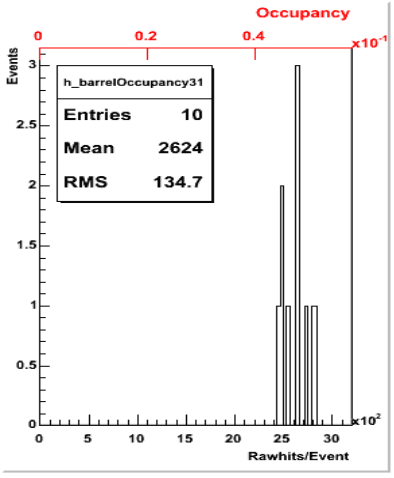
x

Charge sharing scheme:





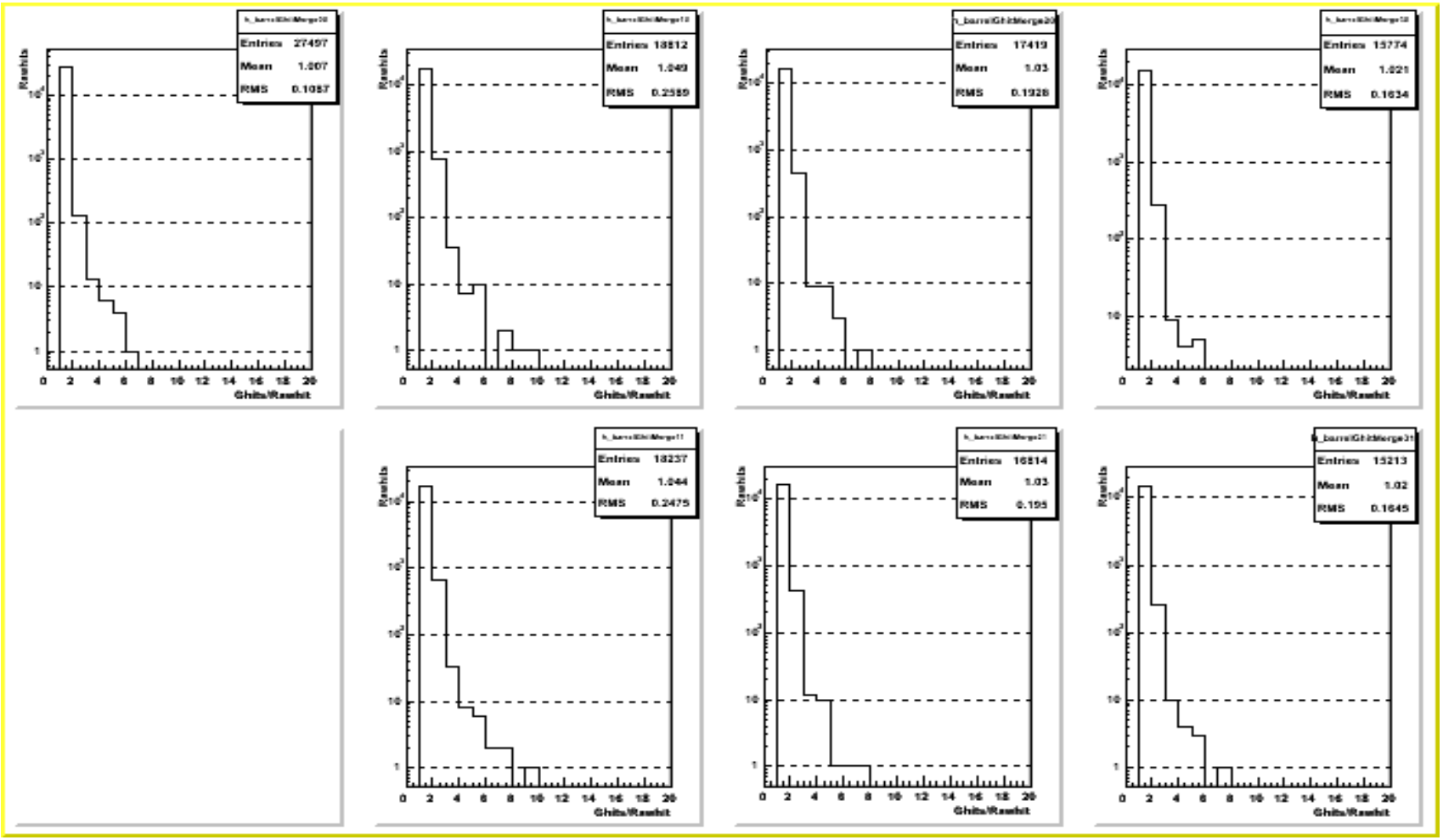




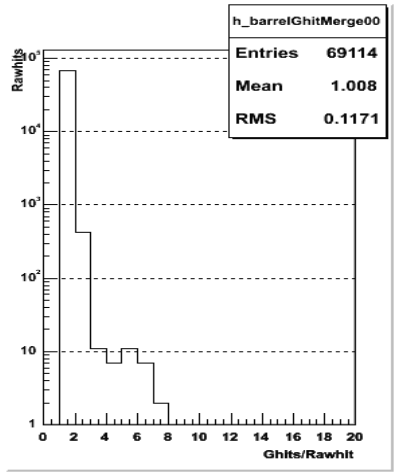
u

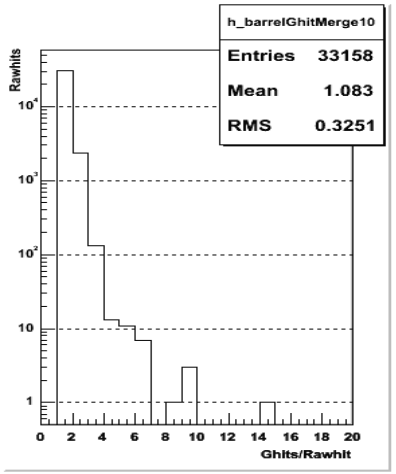
Layer 0: *SvxPixel1* Layer 1: *SvxStrip11* Layer 2: *SvxStrip11* Layer 3: *SvxStrip11*

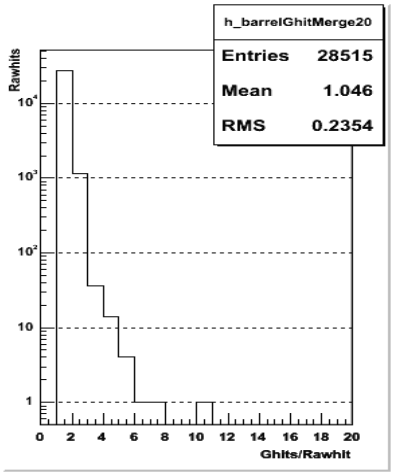
Examples: **Ghit merge**: *HIJING: Central AuAu@200 GeV$_{NN}$*: No charge sharing between pixels and strips

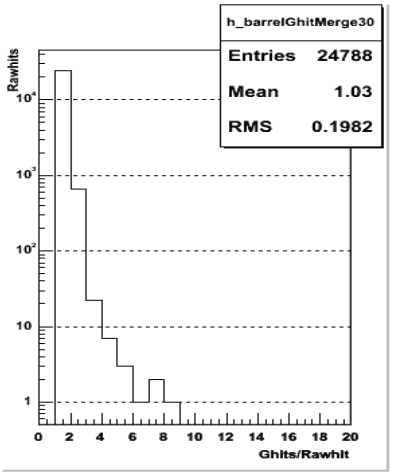


Examples: **Ghit merge:** *HIJING: Central AuAu@200 GeV$_{NN}$:* With charge sharing between pixels and strips



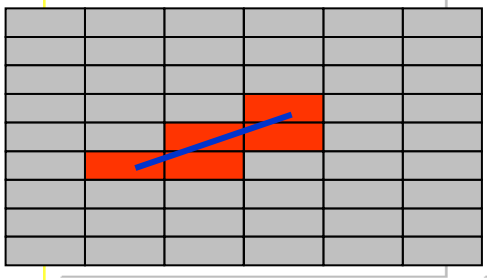


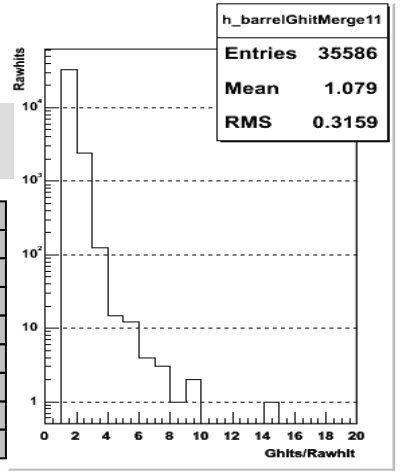


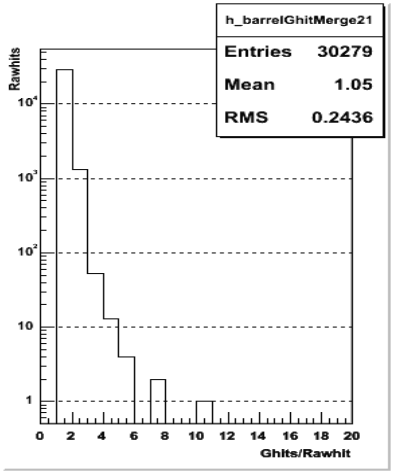


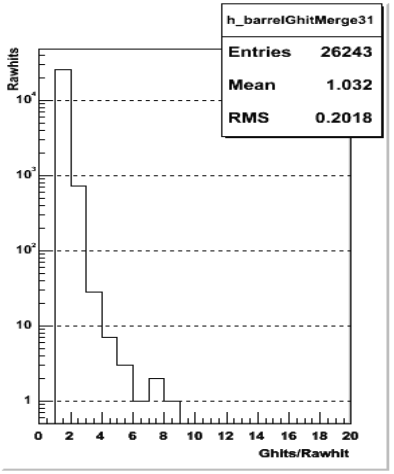
x

Charge sharing scheme:







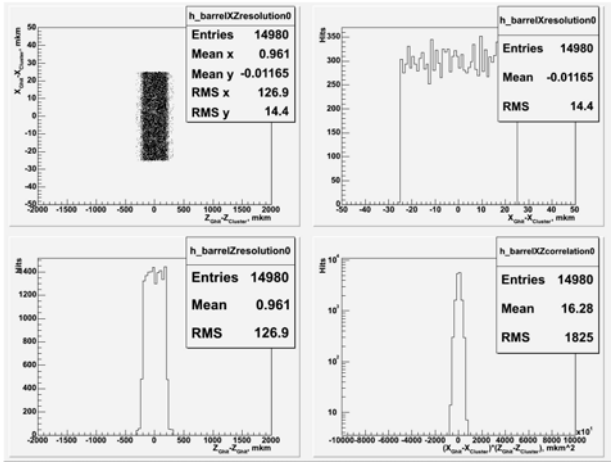


u

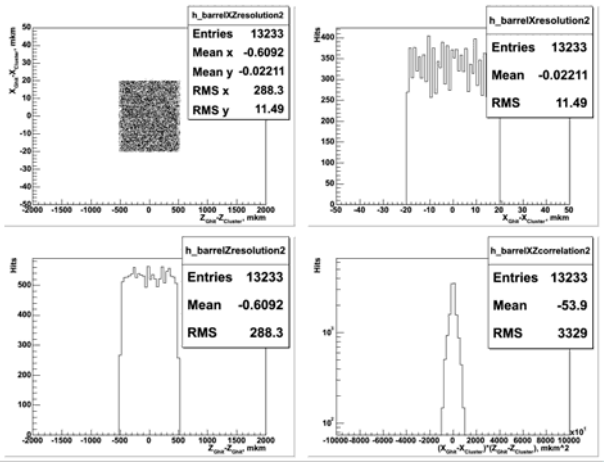
Layer 0: *SvxPixel1* Layer 1: *SvxStrip11* Layer 2: *SvxStrip11* Layer 3: *SvxStrip11*

Examples: *Position resolution*

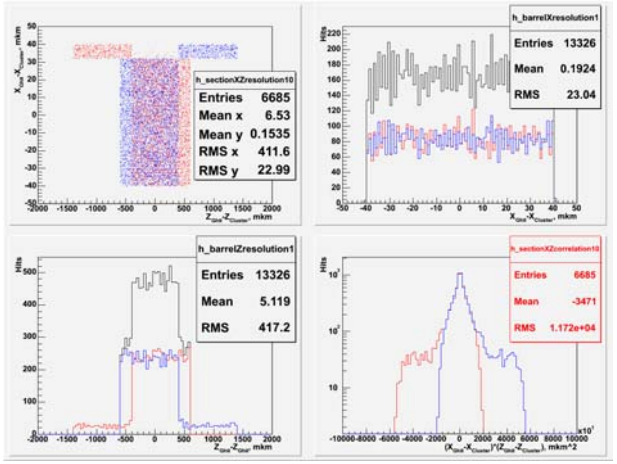
No charge sharing between pixels and strips



SvxPixel1 (32768 ch/sen)



Svxstrip21 (1652 ch/sen)



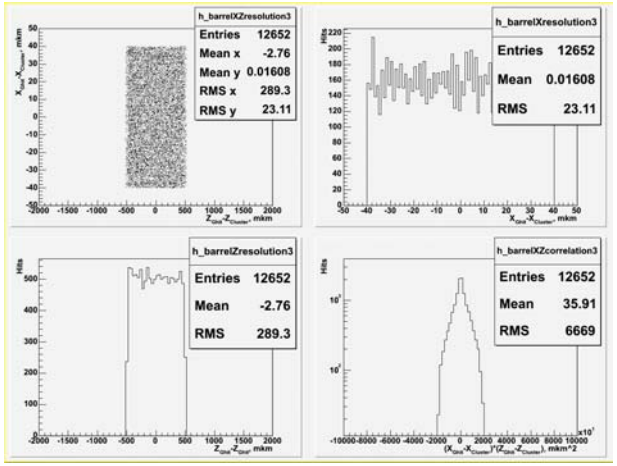
SvxStrip11 (1594 ch/sen)

Red: z-local < 0

Blue: z-local > 0

Black: full sensor

With charge sharing, Z-resolution may partially recover back to ~300 mkm, and X-Z-correlations reduce, particularly for non-perpendicular crossings. Needs to be confirmed with simulations.

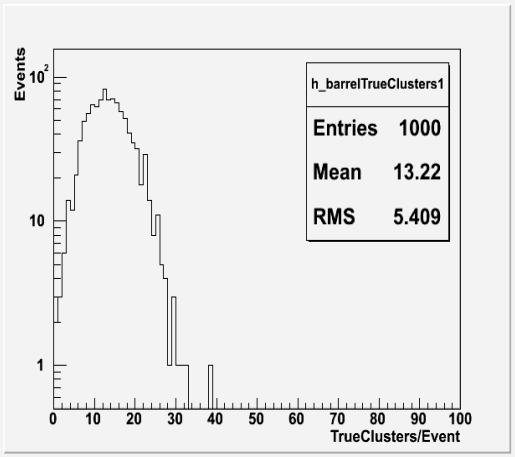


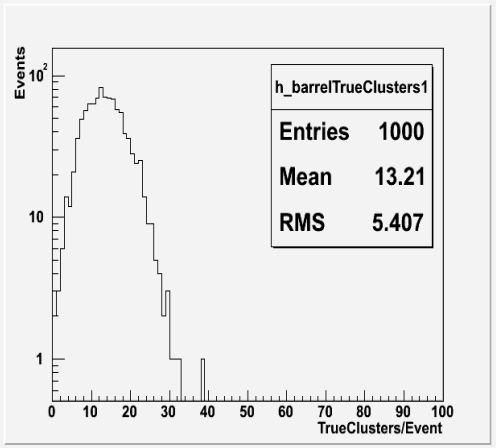
SvxStrip22 (884 ch/sen)

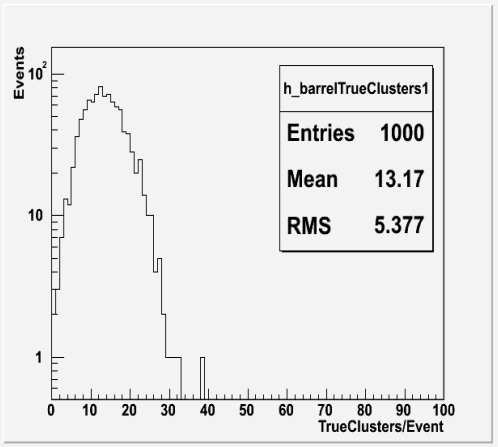
$$< \Delta x \cdot \Delta z >= 3471\,\mu^2 \approx 0.36 \cdot \sqrt{< (\Delta x)^2 > \cdot < (\Delta z)^2 >}$$

Examples: **Clusters in layer 1** (R=6 cm): *PYTHIA: pp@200 GeV, P_T^{jet}>10 GeV/c*

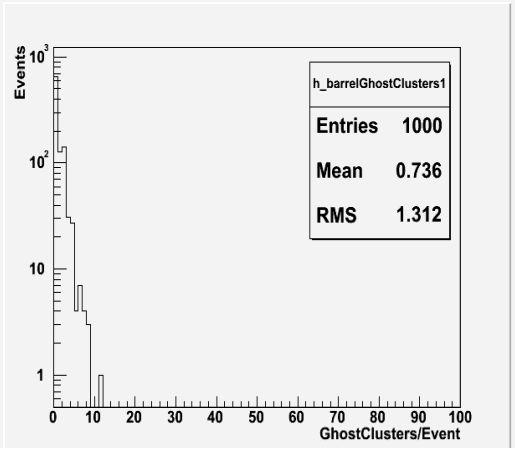
No charge sharing between strips

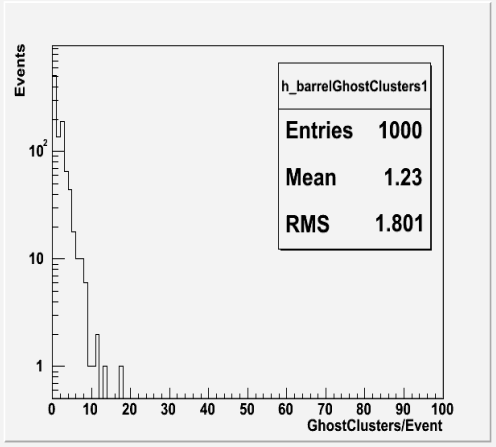


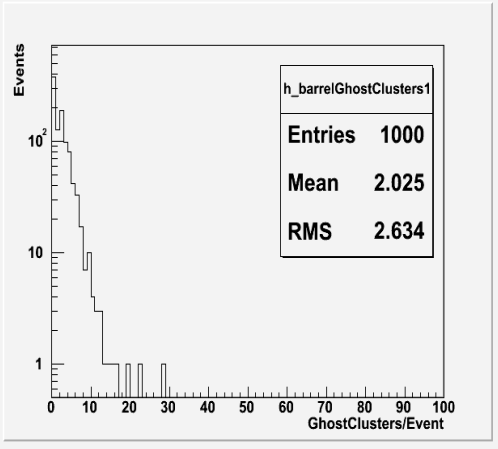




True







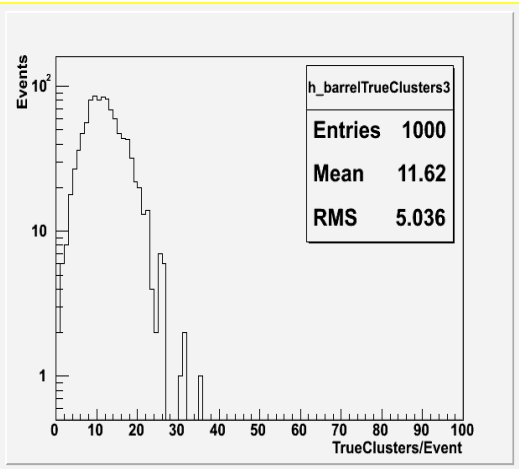
Ghost

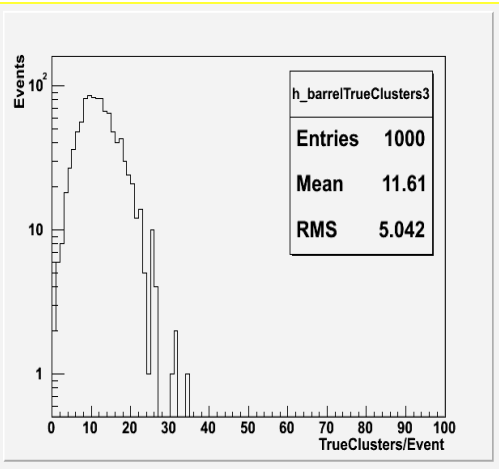
SvxStrip11 (1594 ch/sensor) *SvxStrip21* (1652 ch/sensor) *SvxStrip22* (884 ch/sensor)

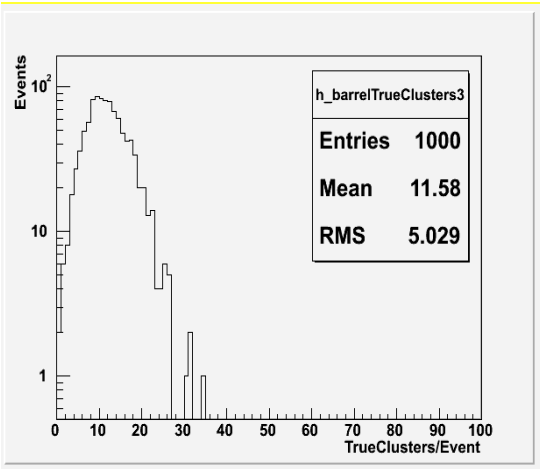
0.02 clust./cm², <d > < 72 mm 0.02 clust./cm², <d > < 71 mm 0.02 clust./cm², <d > < 69 mm

Examples: **Clusters in layer 3** (R=10 cm): *PYTHIA: pp@200 GeV, P_T^{jet}>10 GeV/c*

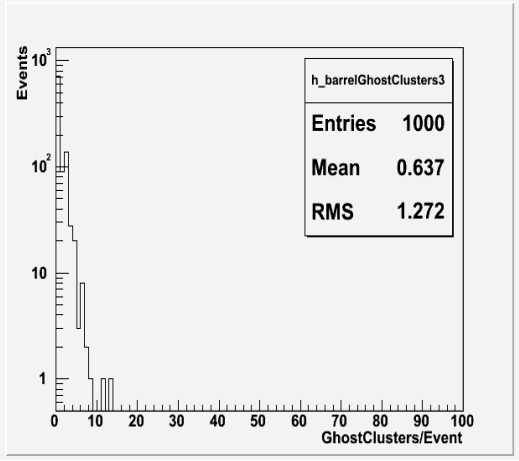
No charge sharing between strips

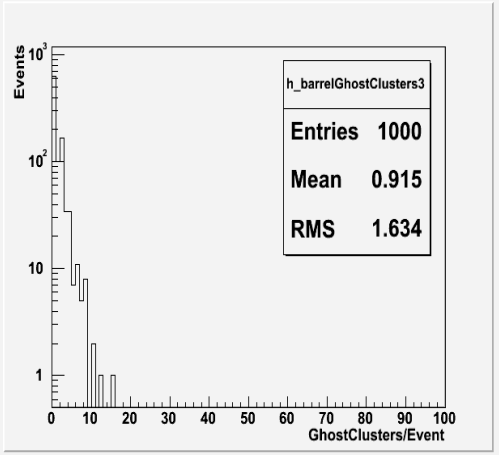


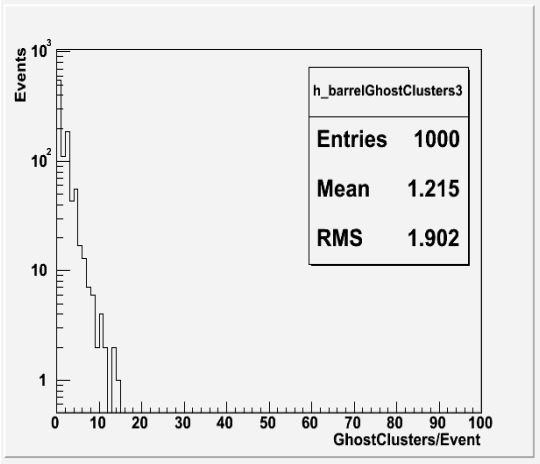




True







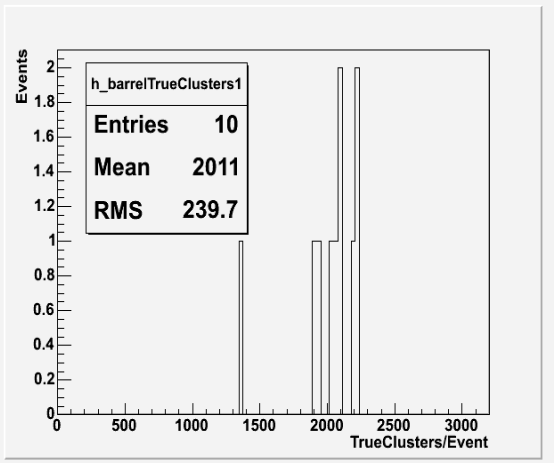
Ghost

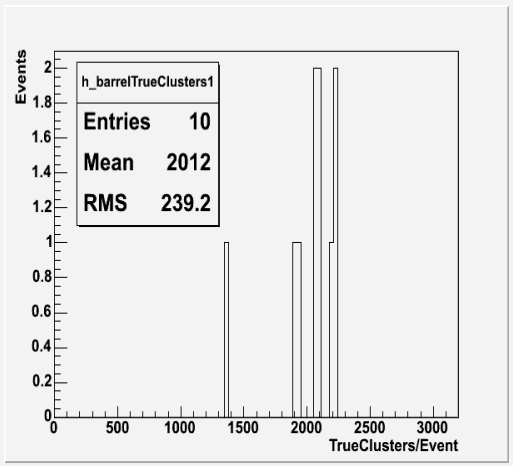
SvxStrip11 (1594 ch/sensor) — *0.01 clust./cm², <d > < 103 mm*

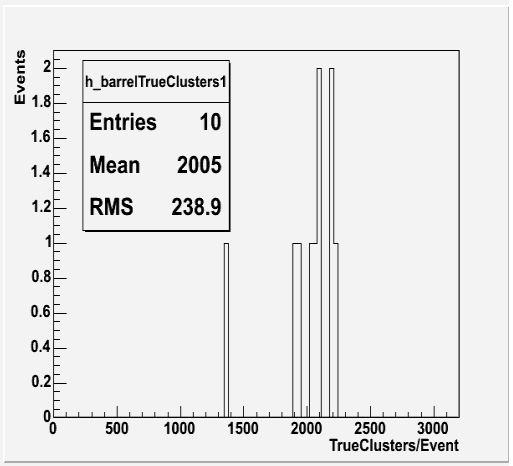
SvxStrip21 (1652 ch/sensor) — *0.01 clust./cm², <d > < 102 mm*

SvxStrip22 (884 ch/sensor) — *0.01 clust./cm², <d > < 101 mm*

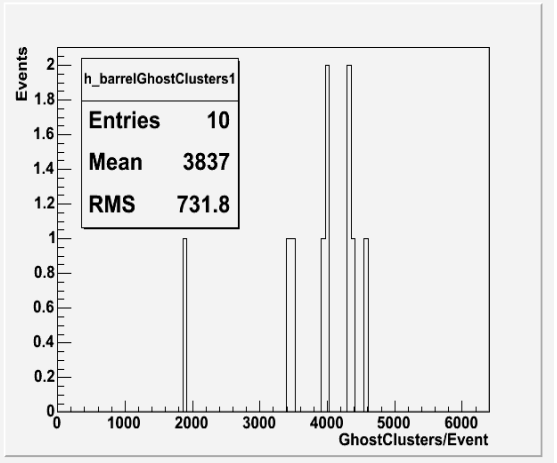
Examples: **Clusters in layer 1** (R=6 cm): *HIJING: Central AuAu@200 GeV$_{NN}$:* No charge sharing between strips

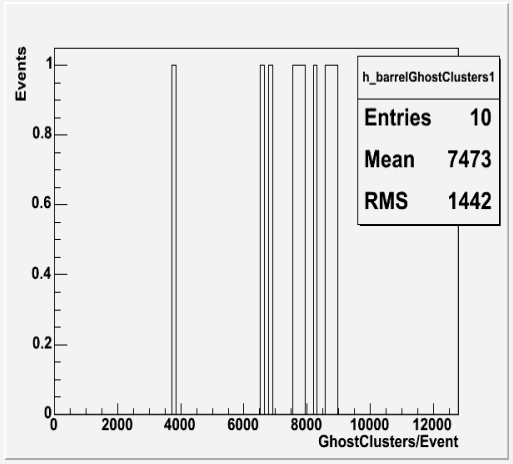


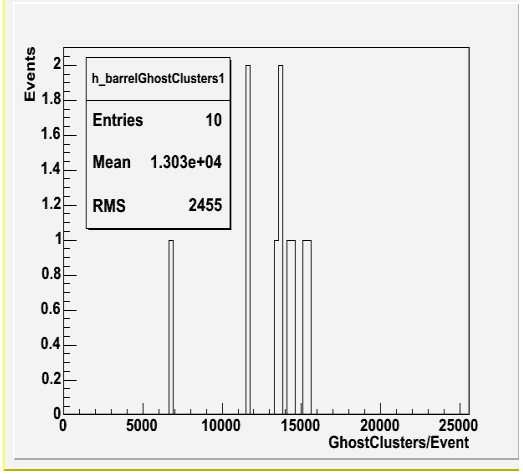




True





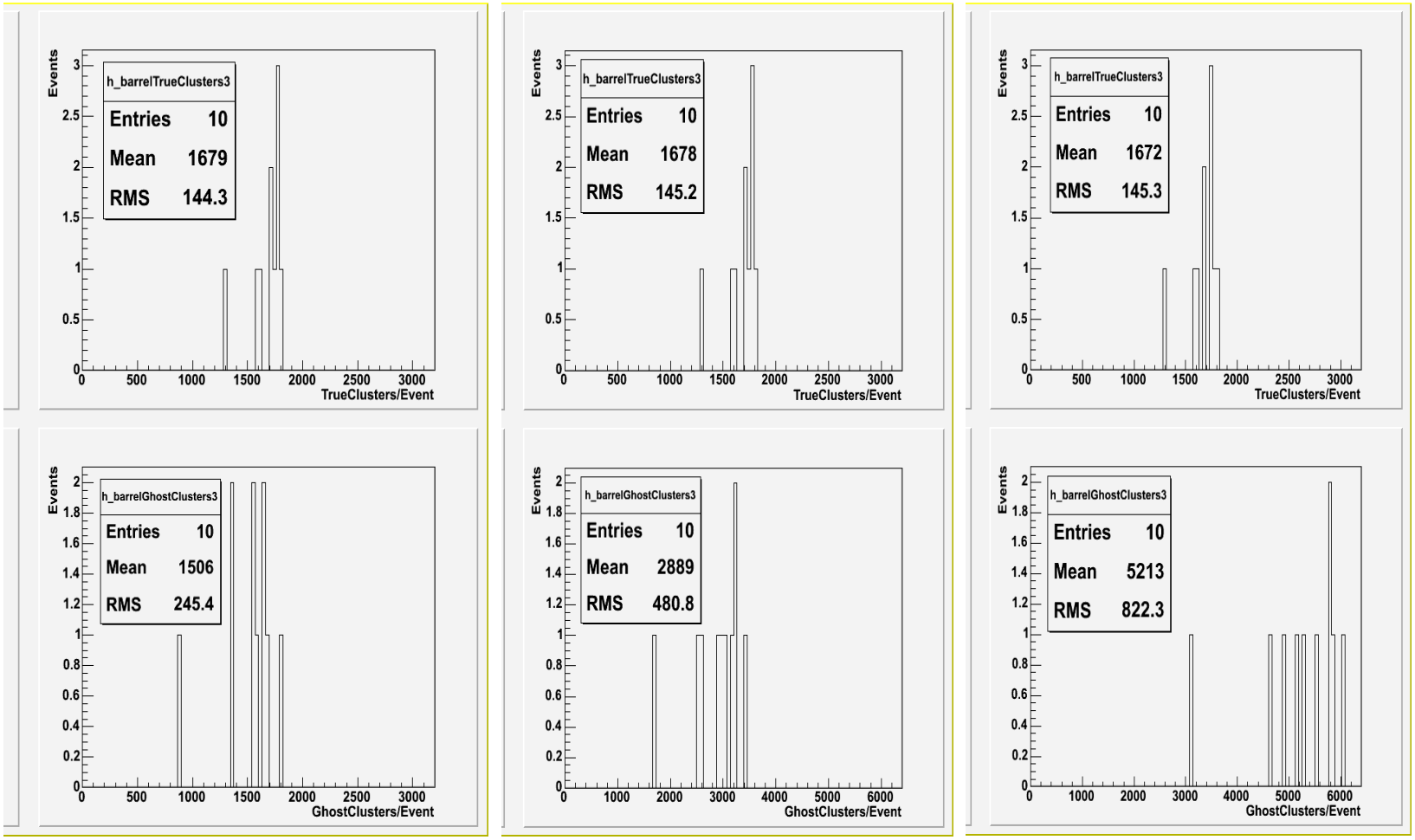


Ghost

SvxStrip11 (1594 ch/sensor) *SvxStrip21 (1652 ch/sensor)* *SvxStrip22 (884 ch/sensor)*

8.1 clust./cm^2, <d >≈3.5 mm *13.2 clust./cm^2, <d >≈2.7 mm* *20.9 clust./cm^2, <d >≈2.2 mm*

Examples: **Clusters in layer 3** (R=10 cm): *HIJING: Central AuAu@200 GeV$_{NN}$*: No charge sharing between strips



True

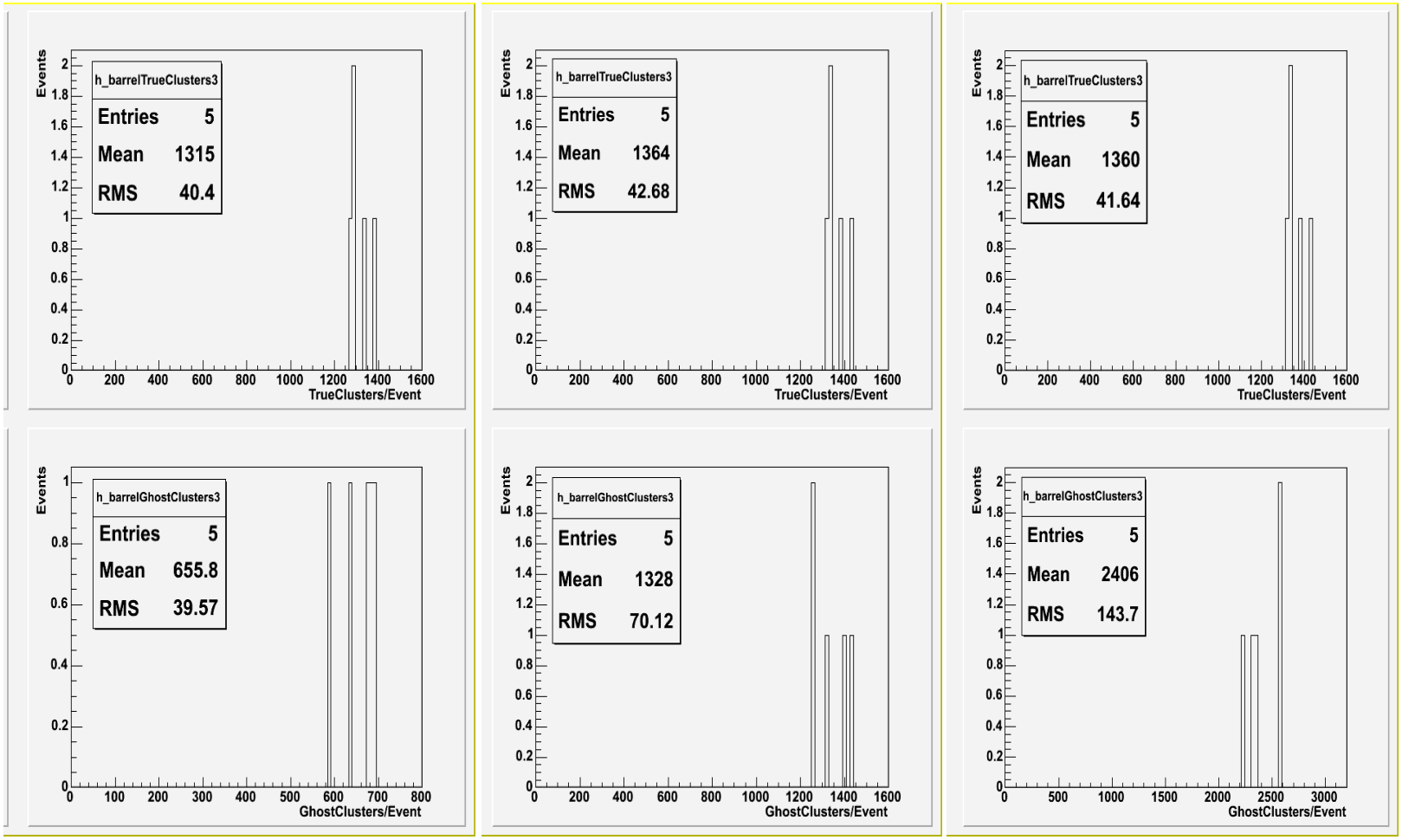
Ghost

SvxStrip11 (1594 ch/sensor) — *2.5 clust./cm², <d >≈6.4 mm*

SvxStrip21 (1652 ch/sensor) — *3.5 clust./cm², <d >≈5.3 mm*

SvxStrip22 (884 ch/sensor) — *5.3 clust./cm², <d >≈4.3 mm*

Examples: **Clusters in layer 3** (R=14 cm): *HIJING: Central AuAu@200 GeV$_{NN}$*: No charge sharing between strips



True

Ghost

SvxStrip11 (1594 ch/sensor) *SvxStrip21 (1652 ch/sensor)* *SvxStrip22 (884 ch/sensor)*

1.1 clust./cm², <d >≈9.6 mm *1.5 clust./cm², <d >≈8.2 mm* *2.1 clust./cm², <d >≈6.9 mm*

Things to do (a lot !!!): PISA

- Short terms
  - Understand the origin of large (long) GEANT hits and make an adjustment in the SVX PISA.
- Long term
  - Arrange writing the SVX parameters, rotation matrices and translation vectors from PISA in the way compliant with the current PISA/Fun4All standards.
  - Build all the chips, wirings, cooling, etc. structures on the sensors as the design evolves.
  - Build the support(s), cage(s), cablings, cooling, etc. as all these evolve in the design.
  - Examine and tune up various GEANT physics cuts.
  - More …

Things to do (even more !!!): SvxReco

- Short terms
  - Accounting charge diffusion in the Si sensors for the SvxRawhit building. Tune up model parameters with the sensor test data.
  - Adding noise.
  - Writing cluster-finder codes, using few versions for merging “pixels”. Evaluating and comparing the versions (position resolution, merging/splitting GEANT hits, etc.), tuning the parameters with the sensor test data.
  - Tracking (out of scope of this report).
  - Performance optimization/clean up (memory, CPU usage).
- Longer terms
  - Mapping DAQ raw data format onto the pixels and strips as the sensor engineering and design evolve. Eventually, reading and handling data from PRDFs.
  - Think about the calibration and alignment procedures and data base. Incorporate calibration and alignment procedures into the SvxReco package.
  - Full integration of the package into the Fun4All framework.
  - Make the package easily switchable between the simulation and real data analysis modes.
  - More and more …

Fresh smart brains and good hands are badly needed !