Pds Camera

Pds::Camera Namespace Reference

Class FrameCoord

Public Member Functions

```
FrameCoord() {}
FrameCoord(unsigned short x, unsigned short y) : column(x), row(y) {}
```

Public Attributes

```
uint16_t column;
uint16_t row;
```

Class FrameFccdConfigV1

class for configuring FCCD frame

Public Types

```
enum { Version=1 };
```

Public Member Functions

```
FrameFccdConfigV1();
FrameFccdConfigV1(const FrameFccdConfigV1&);
//  size of this structure
unsigned  size()        const         { return sizeof(*this); }
```

Class FrameFexConfigV1

class for configuring online frame feature extraction process

Public Types

```
enum { Version=1 };
enum Forwarding { NoFrame, FullFrame, RegionOfInterest };
enum Processing { NoProcessing, GssFullFrame, GssRegionOfInterest, GssThreshold };
```

Public Member Functions

```
    FrameFexConfigV1();
     FrameFexConfigV1( Forwarding        forwarding,
                       unsigned          fwd_prescale,
                       Processing        processing,
                       const FrameCoord& roiBegin,
                       const FrameCoord& roiEnd,
                       unsigned          threshold,
                       unsigned          masked_pixels,
                       const FrameCoord* masked_coords );

     FrameFexConfigV1(const FrameFexConfigV1&);

     //  forwarding policy for frame data
     Forwarding        forwarding() const;

     //  prescale of events with forwarded frames
     unsigned          forward_prescale() const;

     //  algorithm to apply to frames to produce processed output
     Processing        processing() const;

     //  coordinate of start of rectangular region of interest (inclusive)
     const FrameCoord& roiBegin  () const;

     //  coordinate of finish of rectangular region of interest (exclusive)
     const FrameCoord& roiEnd    () const;

     //  pixel data threshold value to apply in processing
     unsigned          threshold () const;

     //  count of masked pixels to exclude from processing
     unsigned          number_of_masked_pixels () const;

     //  location of masked pixel coordinates
     //  appended to the end of this structure
     const FrameCoord& masked_pixel_coordinates() const;

     //  size of this structure
     //  (including appended masked pixel coordinates)
     unsigned size() const;
```

Class FrameV1

Class for rectangular frame data

Public Types

```
     enum {Version=1};
```

Public Member Functions

```
  FrameV1() {}
  //  FrameV1 with unassigned contents
  FrameV1(unsigned width, unsigned height, unsigned depth, unsigned offset);
  //  Copy constructor
  FrameV1(const FrameV1&);
public:
  //  number of pixels in a row
  unsigned width () const;
  //  number of pixels in a column
  unsigned height() const;
  //  number of bits per pixel
  unsigned depth () const;
  //  fixed offset/pedestal value of pixel data
  unsigned offset() const;

  // number of bytes per pixel
  unsigned depth_bytes() const;
  // size of pixel data appended to the end of this structure
  unsigned data_size  () const;

  //  beginning of pixel data
  const unsigned char* data() const;
  //  location of individual pixel datum
  const unsigned char* pixel(unsigned x,unsigned y) const;
```

Class TwoDGaussianV1

Public Types

```
  enum { Version=1 };
```

Public Member Functions

```
  TwoDGaussianV1();
  TwoDGaussianV1(uint64_t n,
                 double xmean,
                 double ymean,
                 double major_ax_width,
                 double minor_ax_width,
                 double tilt);
  ~TwoDGaussianV1();

  uint64_t integral() const;
  double xmean() const;
  double ymean() const;
  double major_axis_width() const;
  double minor_axis_width() const;
  double major_axis_tilt () const;
```