Myana Reference Manual

Here is a list of all files with brief descriptions:

main.hh - declares enums and functions
main.cc - implements getters and other utility functions and classes.
myana.hh - declaration of user functions (`beginrun`, `event` etc)
myana.cc - implementation of user functions (`beginrun`, `event` etc)
myana_morefeatures.cc - same as myana.cc, but has more examples.

main

Namespaces

These are the namespaces used.

```
namespace          Pds
namespace          Pds::CsPad
```

Defines

Define some aliases.

```
#define         getOpal1kConfig(x)   getFrameConfig(x)
#define         getTm6740Config(x)   getFrameConfig(x)
#define         getOpal1kValue(w, x, y, z)   getFrameValue(w, x, y, z)
#define         getTm6740Value(w, x, y, z)   getFrameValue(w, x, y, z)

#define         FNAME_LEN   256
#define         Opal1kDetectorIndex(x, y)   FrameDetectorIndex(x, DetInfo::Opal1000, y)
#define         Tm6740DetectorIndex(x, y)   FrameDetectorIndex(x, DetInfo::TM6740 , y)
#define         FccdDetectorIndex(x, y)   FrameDetectorIndex(x, DetInfo::Fccd, y)
```

Enumerations

These are "named integers" that are used to control e.g. which detector to fetch information from.
Most of these are copied from pdsdata/xtc/DetInfo.hh.

```
enum          AcqDetector {
  AmoIms = 0, AmoGasdet = 1, AmoETof = 2, AmoITof = 3,
  AmoMbes = 4, AmoVmiAcq = 5, AmoBpsAcq = 6, Camp = 7,
  SxrBeamlineAcq1 = 0, SxrBeamlineAcq2 = 1, SxrEndstationAcq1 = 2, SxrEndstationAcq2 = 3,
  NumAcqDetector = 8
}
enum          FrameDetector {
  AmoVmi = 0, AmoBps1 = 1, AmoBps2 = 2, SxrBeamlineOpal1 = 0,
  SxrBeamlineOpal2 = 1, SxrEndstationOpal1 = 2, SxrEndstationOpal2 = 3, SxrFccd = 4,
  XppSb1PimCvd = 0, XppMonPimCvd = 1, XppSb3PimCvd = 2, XppSb4PimCvd = 3,
  NumFrameDetector = 5
}
enum          PrincetonDetector {
  SxrBeamlinePrinceton1 = 0, SxrBeamlinePrinceton2 = 1,
  SxrEndstationPrinceton1 = 2, SxrEndstationPrinceton2 = 3,
  AmoPrinceton1 = 0, AmoPrinceton2 = 1,
  AmoPrinceton3 = 2, AmoPrinceton4 = 3,
  NumPrincetonDetector = 4
}
enum          IpimbDetector {
  SxrBeamlineIpimb1 = 0, SxrBeamlineIpimb2 = 1, SxrBeamlineIpimb3 = 2,
  SxrBeamlineIpimb4 = 3, SxrBeamlineIpimb5 = 4, SxrBeamlineIpimb6 = 5,
  SxrBeamlineIpimb7 = 6, SxrBeamlineIpimb8 = 7, SxrBeamlineIpimb9 = 8,
  SxrBeamlineIpimb10 = 9, SxrBeamlineIpimb11 = 10, SxrBeamlineIpimb12 = 11,
  SxrBeamlineIpimb13 = 12, SxrBeamlineIpimb14 = 13, SxrBeamlineIpimb15 = 14,
  SxrBeamlineIpimb16 = 15,
  SxrEndstationIpimb1 = 16, SxrEndstationIpimb2 = 17, SxrEndstationIpimb3 = 18,
  SxrEndstationIpimb4 = 19, SxrEndstationIpimb5 = 20, SxrEndstationIpimb6 = 21,
  SxrEndstationIpimb7 = 22, SxrEndstationIpimb8 = 23, SxrEndstationIpimb9 = 24,
  SxrEndstationIpimb10 = 25, SxrEndstationIpimb11 = 26, SxrEndstationIpimb12 = 27,
  SxrEndstationIpimb13 = 28, SxrEndstationIpimb14 = 29, SxrEndstationIpimb15 = 30,
  SxrEndstationIpimb16 = 31,
  XppSb1Ipm = 0, XppSb1Pim = 1, XppMonPim = 2, XppSb2Ipm = 3,
  XppSb3Ipm = 4, XppSb3Pim = 5, XppSb4Pim = 6, NumIpimbDetector = 32
}
enum          PnCcdDetector { PnCcd0 = 0, PnCcd1 = 1, NumPnCcdDetector = 2 }
enum          EncoderDetector { SxrBeamlineEncoder1 = 0, NumEncoderDetector = 1 }
enum          Result { OK, End, Error }
```

Typedefs

These are defined in `main.cc`.

```
  typedef vector< AcqWaveForm >  AcqChannelList
  typedef int(*)  TEpicValueFuncPtr (const EpicsPvHeader &epicsPv1, const void **ppValue,
                                     int *piDbrype, struct tm *pTmTimeStamp, int *piNanoSec)
```

Classes

These are all defined in `main.cc`.

- **struct AcqWaveForm**
  Public Attributes

  ```
  double                 vfTrig
  vector< double >         vfTime
  vector< double >         vfVoltage
  ```

- **class myLevelIter**
  Public Member Functions

```
        // Constructor
         myLevelIter (Xtc *xtc, unsigned depth, int iDebugLevel)

        void         process( const DetInfo &di, const ... &cfg) // lots of these
        // one process function for each device configuration.

        int         process (Xtc *xtc)
```

- **class XtcSlice**
  Public Member Functions

```
        XtcSlice (std::string fname)
         ~XtcSlice ()
bool         add_file (std::string fname)
void         init ()
Result         next (Pds::Dgram *dg)
Result         skip ()
const Pds::Dgram &         hdr () const
```

- **class XtcRun**
  Public Member Functions

```
         XtcRun ()
         ~XtcRun ()
void         reset (std::string fname)
bool         add_file (std::string fname)
const char *         base () const
void         init ()
Result         next (Pds::Dgram *dg)
```

Functions

These are all declared in `main.hh` and implemented in `main.cc`.

Time functions and file name

```
int         getTime (int &seconds, int &nanoSeconds)
int         getLocalTime (const char *&time)
int         getFiducials (unsigned &fiducials)
const char *         getInputName ()
const char *         getFileName ()
void         saveName (const char *file, char *dest)
```

Configuration retrieval functions:

```
int         getAcqConfig (AcqDetector det, int &numChannels, int &numSamples, double &sampleInterval)
int         getFrameConfig (FrameDetector det)
int         getPrincetonConfig (Pds::DetInfo::Detector det, int iDevId,
                                int &width, int &height,
                                int &orgX, int &orgY,
                                int &binX, int &binY)
int         getIpimbConfig (Pds::DetInfo::Detector det, int iDevId, uint64_t &serialID,
                            int &chargeAmpRange0, int &chargeAmpRange1,
                            int &chargeAmpRange2, int &chargeAmpRange3)
int         getEncoderConfig (Pds::DetInfo::Detector det, int iDevId)
int         getFccdConfig (FrameDetector det, uint16_t &outputMode,
                           bool &ccdEnable, bool &focusMode, uint32_t &exposureTime,
                           float &dacVoltage1, float &dacVoltage2, float &dacVoltage3,
                           float &dacVoltage4, float &dacVoltage5, float &dacVoltage6,
                           float &dacVoltage7, float &dacVoltage8, float &dacVoltage9,
                           float &dacVoltage10, float &dacVoltage11, float &dacVoltage12,
                           float &dacVoltage13, float &dacVoltage14, float &dacVoltage15,
                           float &dacVoltage16, float &dacVoltage17,
                           uint16_t &waveform0, uint16_t &waveform1, uint16_t &waveform2,
                           uint16_t &waveform3, uint16_t &waveform4, uint16_t &waveform5,
                           uint16_t &waveform6, uint16_t &waveform7, uint16_t &waveform8,
                           uint16_t &waveform9, uint16_t &waveform10, uint16_t &waveform11,
                           uint16_t &waveform12, uint16_t &waveform13, uint16_t &waveform14)
int         getDiodeFexConfig (Pds::DetInfo::Detector det, int iDevId, float *base, float *scale)
int         getIpmFexConfig (Pds::DetInfo::Detector det, int iDevId,
                             float *base0, float *scale0, float *base1, float *scale1,
                             float *base2, float *scale2, float *base3, float *scale3,
                             float &xscale, float &yscale)
int         getCspadConfig (Pds::DetInfo::Detector det, unsigned &quadMask, unsigned &asicMask)
int         getCspadConfig (Pds::DetInfo::Detector det, Pds::CsPad::ConfigV1 &cfg)
```

L1Accept (event/shot) Data retrieval functions

```
int         getAcqValue (AcqDetector det, int channel, double *&time, double *&voltage)
int         getAcqValue (AcqDetector det, int channel, double *&time, double *&voltage, double &trigtime)
int         getFrameValue (FrameDetector det, int &frameWidth, int &frameHeight, unsigned short *&image)
int         getPrincetonValue (Pds::DetInfo::Detector det, int iDevId, unsigned short *&image)
int         getPrincetonTemperature (Pds::DetInfo::Detector det, int iDevId, float &temperature)
int         getIpimbVolts (Pds::DetInfo::Detector det, int iDevId,
                           float &channel0, float &channel1, float &channel2, float &channel3)
int         getFeeGasDet (double *shotEnergy)
int         getEBeam (double &charge, double &energy, // charge and energy
                      double &posx, double &posy,     // position (x,y)
                      double &angx, double &angy)     // angle (x,y)
int         getEBeam (double &charge, double &energy, // charge and energy
                      double &posx, double &posy,     // position (x,y)
                      double &angx, double &angy,     // angle (x,y)
                      double &pkcurr)                 // current
int         getPhaseCavity (double &fitTime1, double &fitTime2, double &charge1, double &charge2)
int         getPvInt (const char *pvName, int &value)
int         getPvFloat (const char *pvName, float &value)
int         getPvString (const char *pvName, char *&value)
int         getPnCcdValue (int deviceId, unsigned char *&image, int &width, int &height)
int         getEvrDataNumber ()
int         getEvrData (int id, unsigned int &eventCode, unsigned int &fiducial, unsigned int &timeStamp)
int         getEncoderCount (Pds::DetInfo::Detector det, int iDevId, int &encoderCount)
int         getDiodeFexValue (Pds::DetInfo::Detector det, int iDevId, float &value)
int         getIpmFexValue (Pds::DetInfo::Detector det, int iDevId, float *channels,
                            float &sum, float &xpos, float &ypos)
int         getCspadQuad (Pds::DetInfo::Detector det, unsigned quad, const uint16_t *&pixels)
int         getCspadQuad (Pds::DetInfo::Detector det, unsigned quad, const Pds::CsPad::ElementV1 *&data)
```

Control data retrieval functions

```
int         getControlPvNumber ()
int         getControlPvName (int pvId, const char *&pvName, int &arrayIndex)
int         getControlValue (const char *pvName, int arrayIndex, double &value)
int         getMonitorPvNumber ()
int         getMonitorPvName (int pvId, const char *&pvName, int &arrayIndex)
int         getMonitorValue (const char *pvName, int arrayIndex, double &hilimit, double &lolimit)
```

Advanced interface

```
int         getEpicsPvNumber ()
int         getEpicsPvConfig (int iPvId, const char *&sPvName, int &iType, int &iNumElements)
int         getEpicsPvValue (int pvId, const void *&value,
                             int &dbrype, struct tm &tmTimeStamp, int &nanoSec)
void         fillConstFrac (double *t, double *v, unsigned numSamples,
                           float baseline, float thresh, TH1 *hist)
void         fillConstFrac (double *t, double *v, unsigned numSamples,
                           float baseline, float thresh, double *edge, int &n, int maxhits)
```

Program control functions

```
void         usage (char *progname)
void         makeoutfilename (char *filename, char *outfilename)
void         anarun (XtcRun &run, unsigned &maxevt, unsigned &skip, int iDebugLevel)
int         main (int argc, char *argv[])
```

myana.cc and myana.hh

Functions

```
void         beginjob ()
void         beginrun ()
void         begincalib ()
void         event ()
void         endcalib ()
void         endrun ()
void         endjob ()
```

myana_morefeatures.cc and myana.hh

Functions

```
void         beginjob ()
void         beginrun ()
void         begincalib ()
void         event ()
void         endcalib ()
void         endrun ()
void         endjob ()
```