Scanning the log files

For the first scan! I used the auto-replay log files.

- These are located at
  - /adaqfs/home/a-onl/tritium/replay/t2root/Rootfiles/log
- I scanned for the the following keywords:
  - "warning","Warning","WARNING","ERROR","error","Error"
- Then I separated them into the following types
  - "Other","Scaler","overflow", "Output","out of bounds","Resolution lock", "Decoder","CODA"
- Each type will get its own output file.

Log file scanner

- Example of the statement stored into the output file:

/adaqfs/home/a-onl/tritium/replay/t2root/Rootfiles/log/848.log
THaOutput::Init: WARNING: Block OldTrackL.* does not match any variables.

/adaqfs/home/a-onl/tritium/replay/t2root/Rootfiles/log/705.log
Warning:: Scalers are handled by event handlers now
Warning::GenScaler:: (1,0) using default num 32 channels
GenScaler:: ERROR: (1,5) inconsistent number of chan.
Warning:: Scalers are handled by event handlers now

What do the errors mean?

Warning:: Scalers are handled by event handlers now

Replay script

```
380
381     //====================================
382     //  Set up Analyzer and replay data
383     //====================================
384     ReplayCore(
385                runnumber,        //run #
386                numevents,        //-1=replay all;0=ask for a number
387                50000,            //default replay event num
388                RNAME,            //output file format
389                ODEF.Data(),          //out define
390                CUTS.Data(),          //empty cut define
391                bScaler,          //replay scalar?
392                bHelicity,        //repaly helicity
393                fstEvt,           //First Event To Replay
394                QuietRun          //whether ask user for inputs
395                );
396
```

ReplayCore

```
117   // step 1: Init analyzer
118   cout<<"replay: Init analyzer ..."<<endl;
119   THaAnalyzer* analyzer = THaAnalyzer::GetInstance();
120   if( analyzer ) {
121     analyzer->Close();
122   } else {
123     analyzer = new THaAnalyzer;
124   }
125   gHaCuts->Clear();
126
127   //Enable scalers
128     if (EnableScalar)
129       {
130       cout<<"replay: Enabling Scalars"<<endl;
131       analyzer->EnableScalers();
132       }
133
134   //Enable Helicity
135   if (EnableHelicity)
136     {
137       cout<<"replay: Enabling Helicity"<<endl;
138       analyzer->EnableHelicity();
139     }
```

ThaAnalyzer Class

```
//_____________________________________________________________________
void THaAnalyzer::EnablePhysicsEvents( Bool_t b )
{
  fDoPhysics = b;
}

//_____________________________________________________________________
void THaAnalyzer::EnableScalers( Bool_t )
{
  cout << "Warning:: Scalers are handled by event handlers now"<<endl;
}

//_____________________________________________________________________
void THaAnalyzer::EnableSlowControl( Bool_t b )
{
  fDoSlowControl = b;
}

//
```

Warning:: Scalers are handled by event handlers now…..
Is a issue with the conversion from analyzer 5 to 6. Since we enable scalers through other means, this is not an issue for us!

Warning::GenScaler:: (1,0) using default num 20 channels

Replay script

```
Tritium_THaScaler100EvtHandler* rEndscaler = new Tritium_THaScaler100EvtHandler("EndRight","HA scaler event type 100");
gHaEvtHandlers->Add(rEndscaler);
```

Tritium_Tritium_THaScaler100EvtHandler

```
while(p<pstop)
  {if (scalers[j]->IsSlot(*p)==kTRUE)
    {scalerloc[j]->found=kTRUE;
     ifound=1;
 goto found1;}
```

IsSlot function in GenScaler.C

```
Bool_t GenScaler::IsSlot(UInt_t rdata) {
  /// Check if this word is the header for the slot we are looking for
  /// Get the number of channels in this module from the header and
  /// save so that bank version of LoadSlot can skip over this module if
  /// it is not the correct one.
  Bool_t result;
  static Bool_t firsttime=kTRUE;
  static Bool_t firstwarn=kTRUE;
  result = ((rdata & fHeaderMask)==fHeader);
  fNumChan = (rdata&fNumChanMask)>>fNumChanShift;
  if (fNumChan == 0) {
    fNumChan=fgNumChanDefault;
    if (firsttime) {
      firsttime = kFALSE;
      cout << "Warning::GenScaler:: (" << fCrate << "," << fSlot << ") "
        "using default num "<<fgNumChanDefault << " channels" << endl;
    }
  }
```

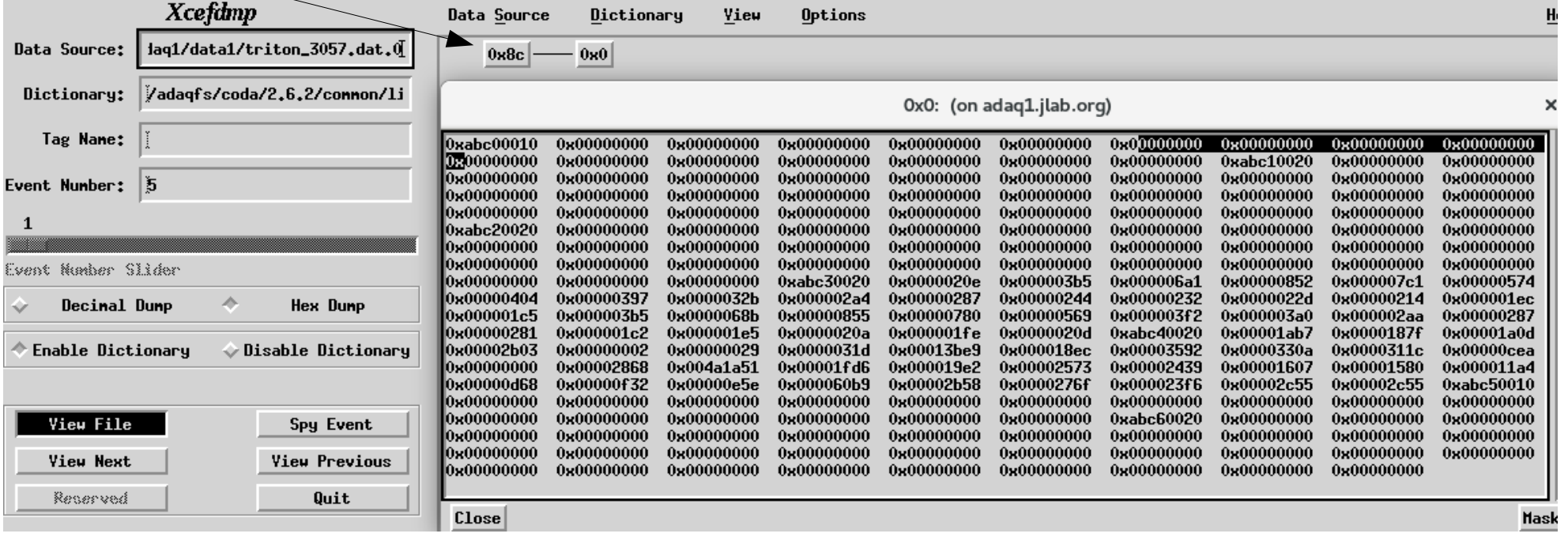
fNumChanMask = 0xff;
fNumChanShift = 0;

Warning::GenScaler:: (1,0) using default num 20 channels

1st – ca 2nd -8c10cc 3rd -c8 4th – 100 5th - abc00010

- Ca = 202 in dec → Total number of words for the event?
- C8 = 200 in dec → Total number of words remaining?
- (abc00010) → abc = left arm, first 0 = slot (0) – next four digits is hex for the number of channels. So 0010 = 16 channels.
- 100 = 256 in dec → ?????? This causes the warning to pop!! Not the header of a scaler… Warning is firing when not looking at scaler.

Looking at the binary for run 3057



GenScaler:: ERROR: (1,5) inconsistent number of chan.
GenScaler:: ERROR: (10,3) inconsistent number of chan.
Same files as last error!

IsSlot function in GenScaler.C

```
if (result && fNumChan != fWordsExpect) {
  if (fNumChan > fWordsExpect)
    fNumChan = fWordsExpect;

  // Print warning once to alert user to potential problems,
  // or for every suspect event if debugging enabled
  if( firstwarn || fDebugFile ) {
    cout << "GenScaler:: ERROR:  (" << fCrate << "," << fslot << ") "
         << "inconsistent number of chan."<<endl;
    //        DoPrint();
    firstwarn = false;
  }
}
return result;
```

```
result = ((rdata & fHeaderMask)==fHeader);
```

fHeaderMask = 0xff00;

fHeader = fSlot << 8;

For abc5 → stored as 1151 in the scaler file on adaq ,but in our DB file stored as 3800.

In data stream the number of channels set is 16 so 16 != 32 from DB.

For ceb3 stored as 3800 in adaq, but Our DB → 1151 with 16 channels.

Decoder:: WARNING: Fastbus module in (roc,slot) = (3,1) found in data but NOT in cratemap !

CodaDecoder.cxx

```
  if ( fbfound[index] && !fMap->slotUsed(iroc, islot)) {
    if (fDebugFile) *fDebugFile << "FB slot in data, but NOT in cratemap  (bad!).  roc = "<<iroc<<"   slot = "<<islot<<endl;
    slotstat[index]=3;
  }
for (Int_t iroc=0; iroc<MAXROC; iroc++) {
  if ( !fMap->isFastBus(iroc) ) continue;
  for (Int_t islot=0; islot<MAXSLOT; islot++) {
    Int_t index = MAXSLOT*iroc + islot;
    if (slotstat[index]==3) cout << "Decoder:: WARNING:  Fastbus module in (roc,slot) = ("<<iroc<<","<<islot<<")  found in data b
  }
}
```

From the Debug file in CodaDecoder

```
FB slot NOT in data, but in cratemap  (bad!).  roc = 2   slot = 18
FB slot NOT in data, but in cratemap  (bad!).  roc = 2   slot = 31
FB slot in data, but NOT in cratemap  (bad!).  roc = 3   slot = 1
FB slot in cratemap and in data.  (good!).  roc = 3   slot = 16
FB slot in cratemap and in data.  (good!).  roc = 3   slot = 17
FB slot in cratemap and in data.  (good!).  roc = 3   slot = 18
FB slot in cratemap and in data.  (good!).  roc = 3   slot = 19
FB slot in cratemap and in data.  (good!).  roc = 3   slot = 20
FB slot in cratemap and in data.  (good!).  roc = 3   slot = 21
FB slot NOT in data, but in cratemap  (bad!).  roc = 3   slot = 22
FB slot in cratemap and in data.  (good!).  roc = 3   slot = 31
FB slot in data, but NOT in cratemap  (bad!).  roc = 4   slot = 1
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 3
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 4
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 5
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 6
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 7
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 8
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 9
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 10
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 11
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 17
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 18
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 22
FB slot in cratemap and in data.  (good!).  roc = 4   slot = 31
FB slot in data, but NOT in cratemap  (bad!).  roc = 5   slot = 1
FB slot in cratemap and in data.  (good!).  roc = 5   slot = 16
FB slot in cratemap and in data.  (good!).  roc = 5   slot = 17
```

1 CODA Error Runs: 2274,2770

/THaAnalyzer.cxx

```
    cout << "Counter summary:" << endl;

    first = false;

  }

  cout << setw(w) << GetCount(i) << "  " << text << endl;
```

/THaAnalyzer.h

```
//---------------- inlines --------------------------
inline UInt_t THaAnalyzer::GetCount( Int_t i ) const
{
  return fCounters[i].count;
```

/THaAnalyzer.cxx

```
{ kDecodeErr,        "decoding error" },
{ kCodaErr,          "CODA errors" },
{ kRawDecodeTest,    "skipped after raw decoding" },
```

/THaAnalyzer.cxx

```
//_____________________________________________________________________
Int_t THaAnalyzer::ReadOneEvent()
{
  // Read one event from current run (fRun) and raw-decode it using the
  // current decoder (fEvData)

    default:
      Incr(kCodaErr);
      break;
```

```
Int_t status = THaRunBase::READ_OK;

status = fRun→ReadEvent();

switch( status ) {
   case THaEvData::HED_OK:     // fall through
   case ThaEvData::HED_WARN:
   case ThaEvData::HED_ERR:
   case ThaEvData::HED_FATAL:
   case THaRunBase::READ_EOF:    // fall through
   case THaRunBase::READ_FATAL:
```

THaSlotData: Warning in loadData: channel 125 out of bounds, ignored, on crate 4 slot 31

- Runs 776,777,779,870,876,910,913,2007,2408
- For 2408 crate 5 slot 31
- From DB both slots have 96 channels

THaSlotData.cxx

```
if (chan < 0 || chan >= (int)maxc) {
  if (VERBOSE) {
    cout << "THaSlotData: Warning in loadData: channel ";
    cout <<chan<<" out of bounds, ignored,"
         << " on crate " << crate << " slot "<< slot << endl;
  }
```

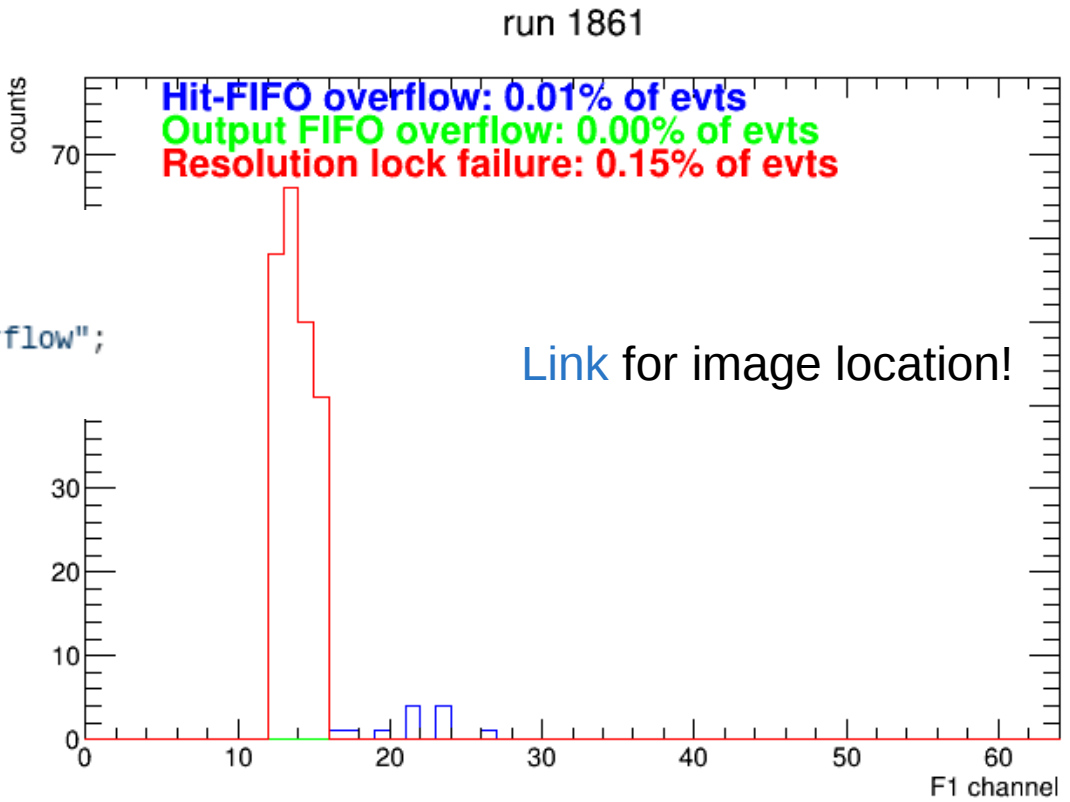
THaOutput::Init: WARNING: Block _____ does not match any variables.

- L.tr.z → should be L.tr.vz or could use L.tr.*
- OldTrackL.* -> Bool_t bOldTrack=kFALSE;
  - Old track is turned off, but still in odef!
- LeftEDTM_s2 → scaler values need to be added into their DB also.
- FbusLurb* → Need to add unrastered beam class into replay
- RightFB_remote → scaler values need to be added into their DB also.
- DR.ltClock* → Decoder values need to be added into their DB
- DR.ltL1* → Decoder values need to be added into their DB

Warning: F1 TDC Slot (Ch) = 14(22) Output FIFO overflow
Warning: F1 TDC Slot (Ch) = 14(21) Hit-FIFO overflow
Warning: F1 TDC Slot (Ch) = 14(14) Resolution lock failure!

```
if (okslot && f1slot!=30 && ((*loc) & DATA_CHK) != F1_RES_LOCK ) {
  if(nwarnings<10) {
     cout << "\tWarning: F1 TDC " << hex << (*loc) << dec;
     cout << "\tSlot (Ch) = " << f1slot << "(" << chan << ")";
  }
```

```
const UInt_t F1_HIT_OFLW = 1<<24; // bad
const UInt_t F1_OUT_OFLW = 1<<25; // bad
const UInt_t F1_RES_LOCK = 1<<26; // good
const UInt_t DATA_CHK = F1_HIT_OFLW | F1_OUT_OFLW | F1_RES_LOCK;
```




```
 if ( (*loc) & F1_OUT_OFLW ) {
     fWarnings[chSlot] |= 1UL << 1; // 2nd bit
     if(nwarnings<10) cout << "\tOutput FIFO overflow";
 }
```

Link for image location!

```
 if ( (*loc) & F1_HIT_OFLW ) {
     fWarnings[chSlot] |= 1UL << 0; // 1st bit
     if(nwarnings<10) cout << "\tHit-FIFO overflow";
 }
```

```
if ( ! ((*loc) & F1_RES_LOCK ) ) {
    fWarnings[chSlot] |= 1UL << 2; // 3rd bit
    if(nwarnings<10) cout << "\tWarning: F1 TDC " << hex << (*loc) << dec << "\tResolution lock failure!";
}
```