PRS/Muon Activities & Status of the Track-Finder Prototype

D. Acosta
University of Florida

PRS/Muon Group Activities

Track-Finder test software status

Study of CLCT patterns

HLT Milestones

The June HLT milestones are:

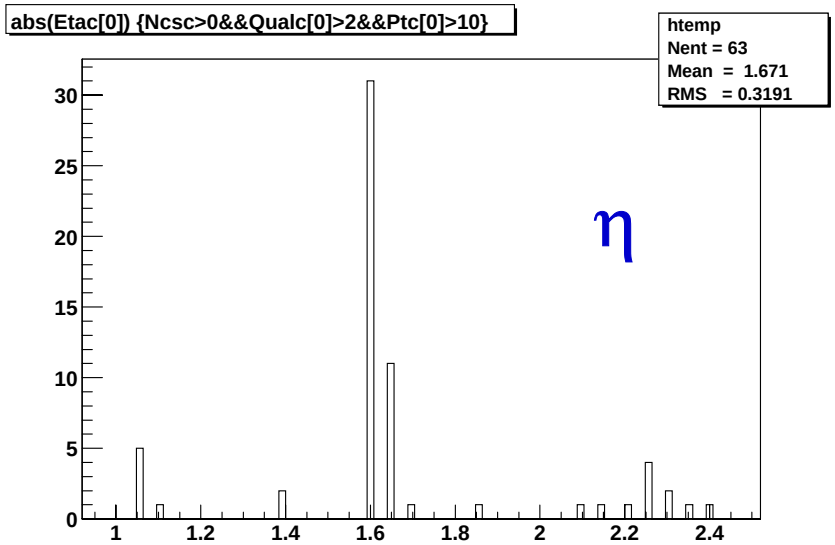
- Complete HLT selection for high-lumi scenario
- HLT results on B physics
- CPU analysis for high lumi selection
- Repeat on line selection for low-lumi

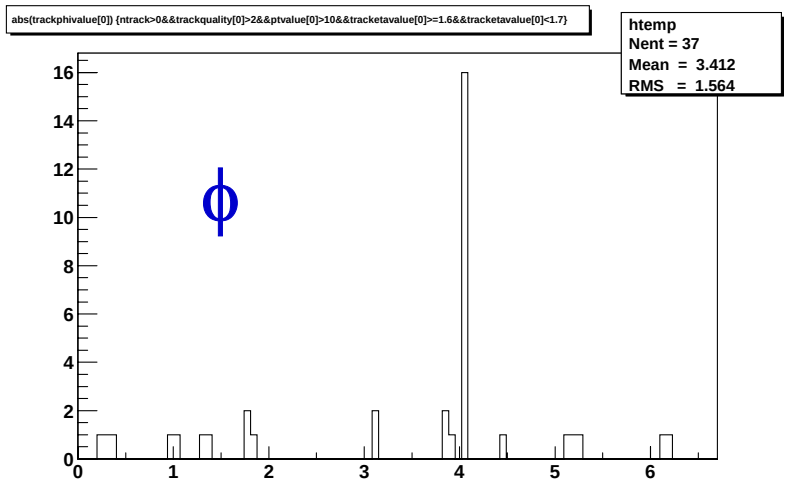
Must have results in DAQ TDR by early September!

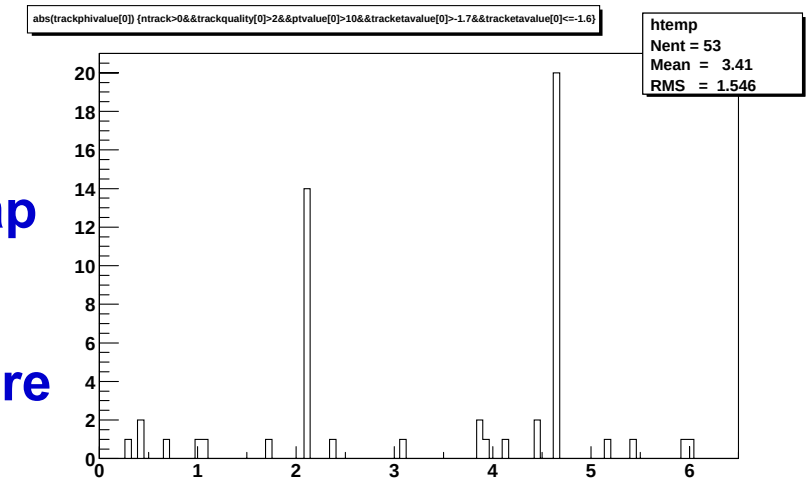
We don't have these results yet, but the current status and L1 results were reported at the June CMS Week

- HLT Muon code had severe crashes, infinite loops, and memory leaks that prohibited collecting any statistics on our HLT algorithms
- After monumental debugging effort, crashes traced to incorrect use of "ReferenceCounted" objects
  - User must never delete, even if performed new!
- In L1 results, rate spike appeared at η=1.6
  - New holes in geometry?
- Problems were "fixed" for ORCA_6_2_0 release

L1 CSC Rate Spike







- Contributes ~1 kHz to L1 rate
- Spike occurs in ϕ and η!
- Region of crack between barrel/endcap
- Traced to ambiguity in p_T assignment for low p_T muons (or punch-through)
- Fixed in CSC Track-Finder (but not sure why this is a problem only now)
- UCD may have student look into this

Bumps along the way…

The "fixed" Muon RootTreeMaker in ORCA_6_2_0 still had a large memory leak (200–500kB/event)

- ➔ Analysis stopped at CERN (batch nodes were dying)
  - Later was found that batch nodes were incorrectly reporting memory usage by a factor 3!
- ➔ Muon HLT code alone was shown to have a leak of "only" 16 kB/event when released
  - So is it because events have more occupancy with pile-up, or is it because of jet/tau/pixel code?

But since then, a new release of the reconstruction software was made, ORCA_6_2_2, to solve some problems with the barrel reconstruction

So we had a two-fold plan:

- ➔ Finish previous Root Tree production at INFN to get us a set of baseline results for HLT milestone
  - Done
- ➔ Finish new HLT reconstruction code and re-process
  - Root Tree production ongoing. Expect results by ~ Aug.30

Muon Analysis at Fermilab

- Request made to get the Muon Federations copied from CERN
  - ➔ Pt4 single muon sample was highest priority
  - ➔ Pt1, Pt10, W, Z, and t-tbar to follow
  - ➔ $Z \rightarrow \mu\mu$ (on-peak and above) already available
- Fermilab participating in the latest Muon re-processing
- Root Trees will be copied from CERN as well, once available
- US users thus have at least one local choice for an analysis center, in addition to CERN
- Mechanism to obtain FNAL visitor id and computer accounts remotely works well

DAQ TDR

I'm currently writing the muon sections of the DAQ TDR

- Ch.13:
  - Simulation, raw data formats, and basic detector reconstruction (e.g. CSC cluster fitting a la Gatti formula)
- Ch.14:
  - HLT methods and performance (the meat)

Hope to have drafts finished by September deadline

You're welcome to read and comment…

Sector Processor Test Software

This summer we resurrected the test software used in the Florida tests of 2000

- **The SP2000 still works! (BGAs still work!)**

Code is in a CVS repository, and has the following packages and subpackages:

- **ORCA**
  - **A standalone version used in the test to validate the hardware results. This has been fully updated to the latest SR and SP firmware. SR LUTs defined with help from Slava. Still need to prepare some data files for test.**
- **LB**
  - **bedtest: National "SCANEASE" software ported to download Xilinx FPGAs using JTAG via Bit3 VME interface**
  - **jam: Same thing for Altera**
  - **LoadLookUps: Downloads SP LUTs**
  - **svf2evf, datToBin: utility routines to convert file formats**
  - **LoadChips: the original JAVA GUI written by Atamanchouk**

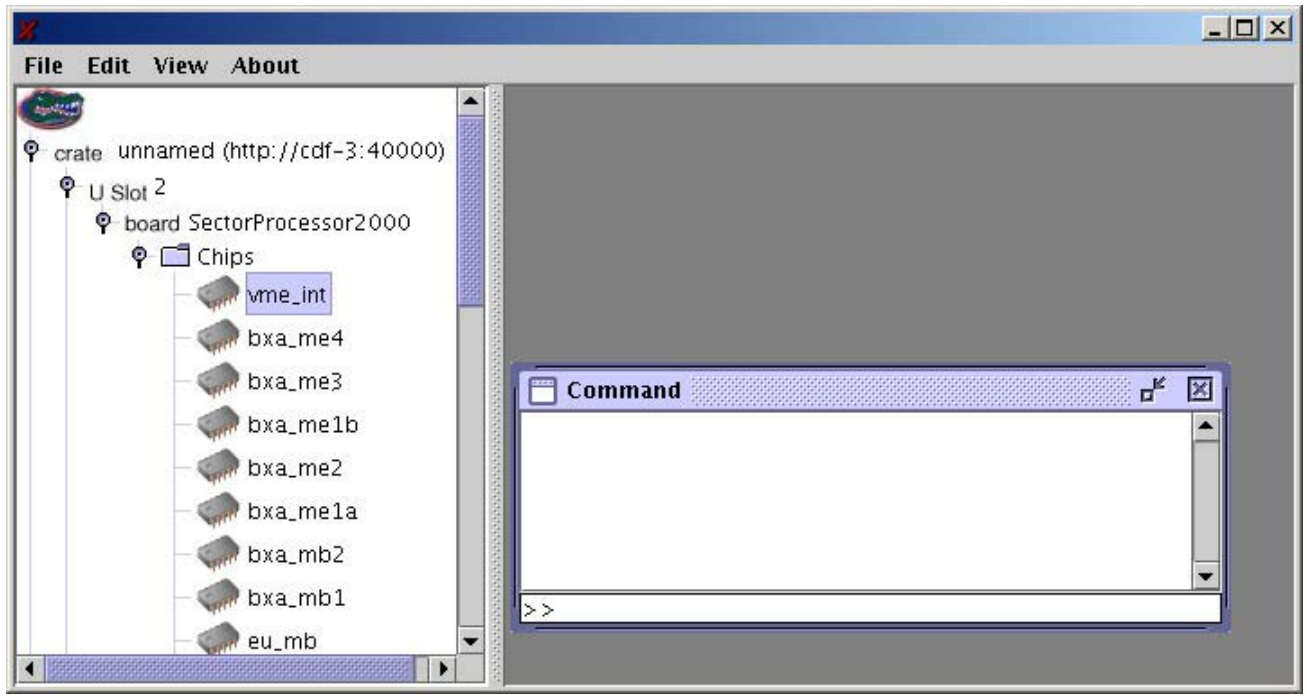
Porting of Test Software

- The "LB" package has been ported from MSWindows to Linux by Holger Stoeck – uses a common Makefile scheme
- We used the Linux Bit3 driver modified by C.Schwick
  - ➔ Successfully works on Linux 2.4 kernel
    - v1.0 from SBS does not, nor does v2.0 beta version
- Works with new SBS 620
- Time penalty of using Linux driver over Windows roughly a factor 2
  - ➔ Will study compiling driver with optimization and without debug
  - ➔ Will try faster machine than 200 MHz

Crate Loader

A summer student from Ohio, Paul Pfeiffer, spent some time generalizing the Java GUI to load more than one card in a crate and more than one crate

- ➔ **This is to prepare for a real chain test down the road, and to become compatible with XDAQ**
- ➔ **Keeps track of the FPGA and LUT configuration files, calls low-level downloading programs in LB package**



Plans

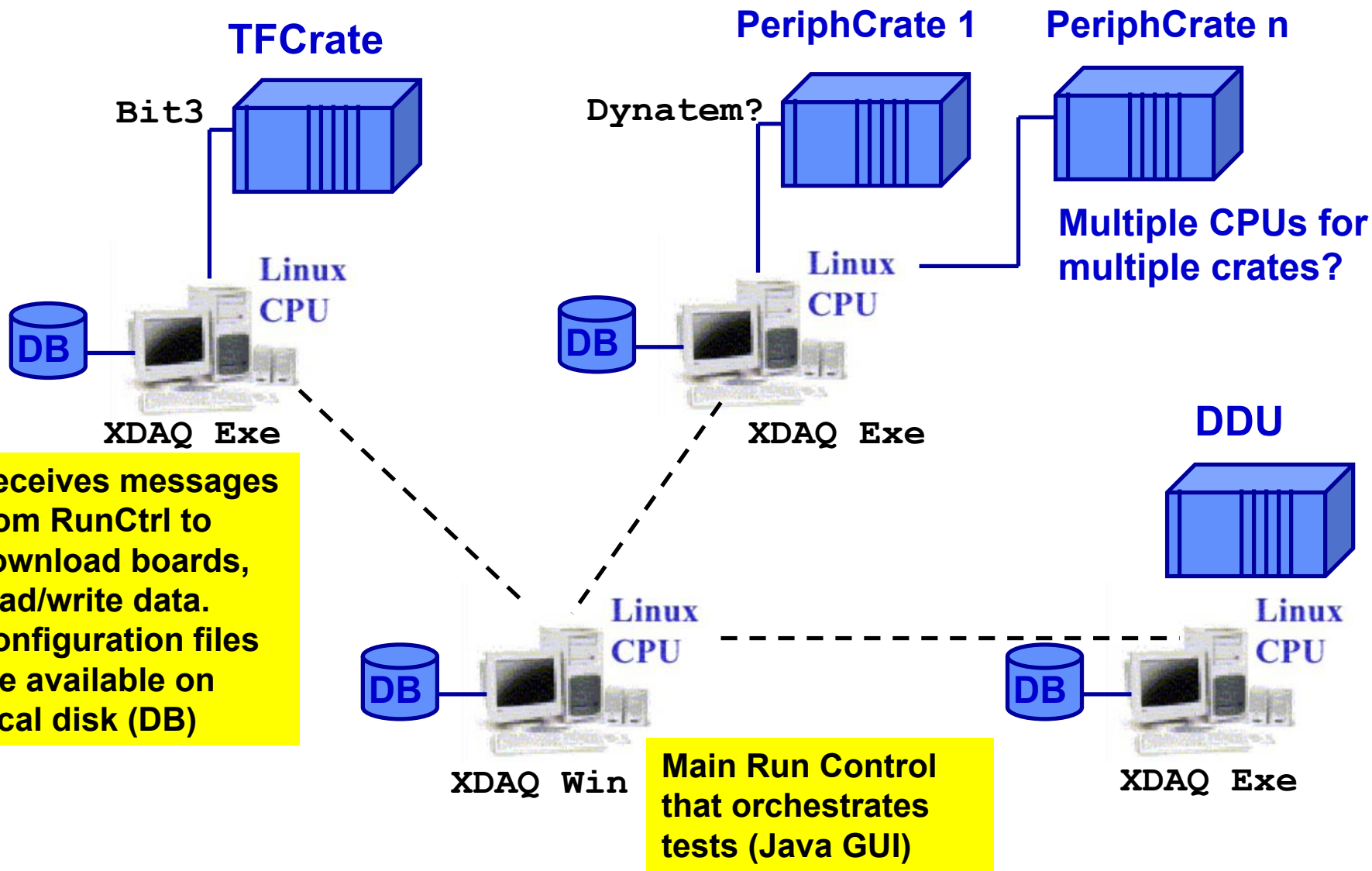
Short-term:

- We probably can start our tests of a single SP board using the existing software with some "slight" modification
  - New FPGAs and LUTs to download. New JTAG interface?
  - New VME registers to read/write data

Long-term:

- We want to perform chain tests of several different boards, and perhaps even multiple crates.
- Modify LB packages to be compatible with the Hardware Access Library of XDAQ
  - Provides a common VME interface in the user code
  - Another time penalty?
- Separate the LB packages into the "XDAQ Executive" (which runs low level programs) and the JAVA GUI into the "XDAQ Win" (which becomes the Run Control)
  - This will facilitate communication between multiple crates and multiple CPUs if needed
  - Need a good example beyond a single register read/write

Possible XDAQ Implementation



CLCT Pattern Study

Bobby has been looking into using the CLCT patterns to improve the Track-Finder p_T resolution, and ultimately reduce the trigger rate

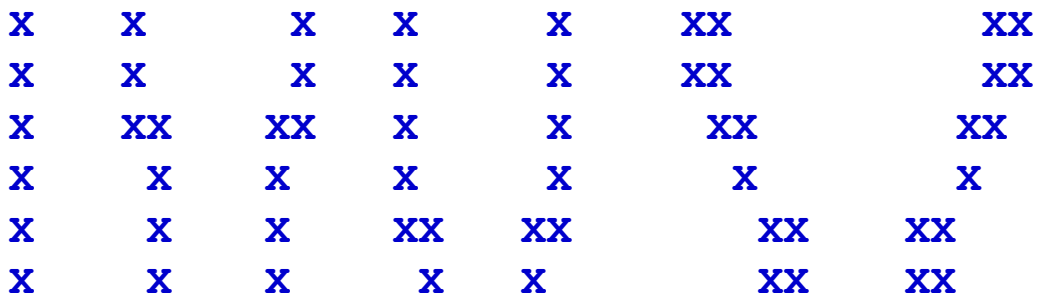
In particular, it would be useful to know if we can trigger effectively with LCTs in just two stations using the bend angle in ME1. Right now we require 3 stations to trigger without RPC, and we don't have ME4.

This study also addresses hardware requirements:

- How many patterns should be implemented in CLCT?
- Do we need di-strip patterns?
- How many bend angle bits are needed by the SP?
  - We reserved 5, but use 0
- How much additional logic space do we need in the SP FPGA to accommodate the bend angle cuts?
- Does the p_T assignment memory need to be made larger?

Patterns Used

Bobby used a variation of Jason's set of 7 patterns:

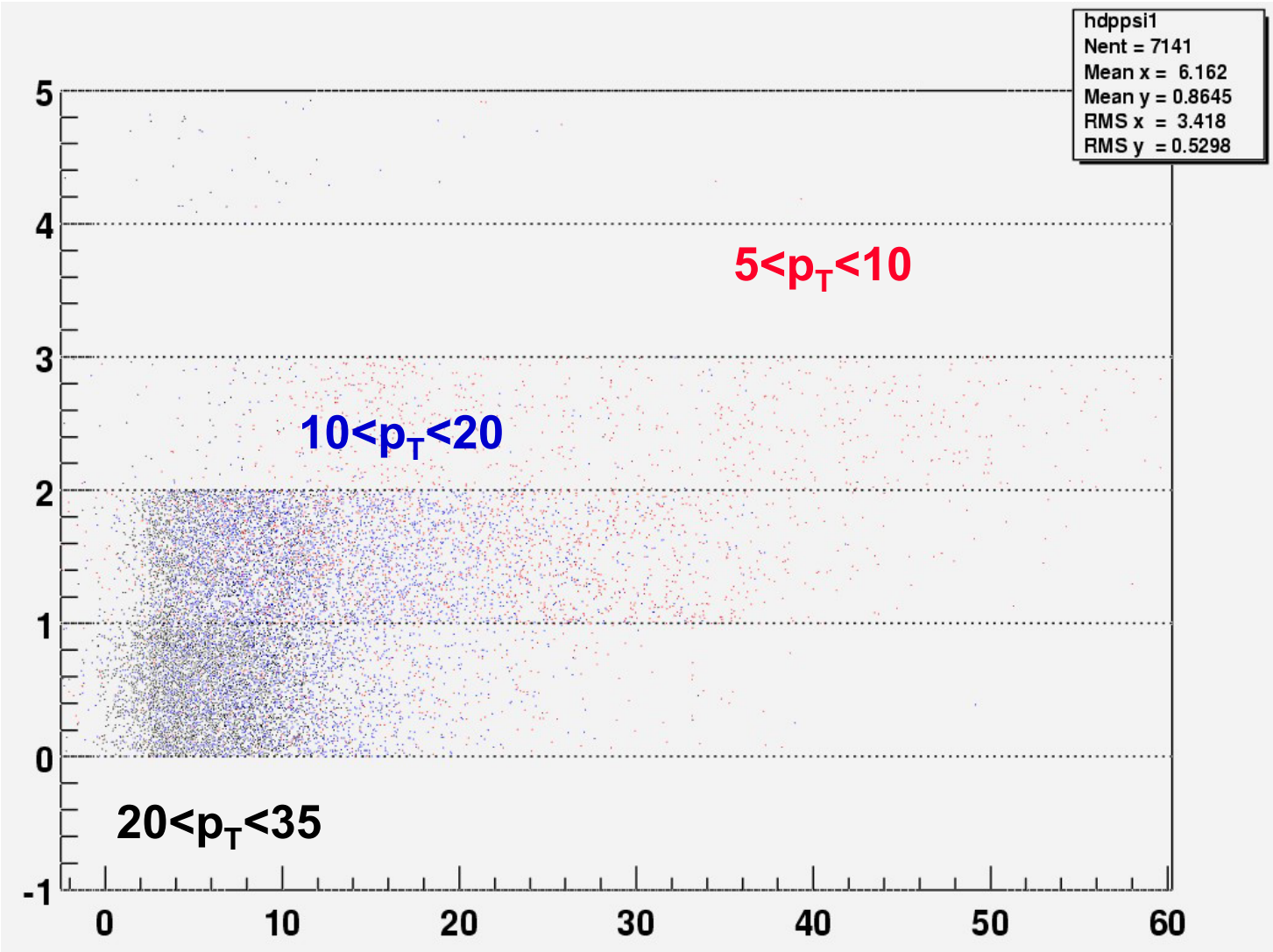


Two hits in same layer implies "OR"

➔ ψ=0 1 -1 1 -1 2 -2

- ➔ Units of bend angle are in number of ½-strips in extent
- ➔ Last two patterns are a variation of what Jason had, and actually represent ψ=2,3,4
- ➔ Technical problems didn't allow us to separate this list out into more patterns (need help from Jason/Slava)
- ➔ With this list, 5% of LCTs with $5<p_T<100$ were from di-strip patterns (4% $5<p_T<10$)
- ➔ Uses just 3-bits to label patterns

Correlation Plot



P_T Assignment

For 2 stations, we parameterize the p_T-dependence of $\Delta\phi$ as: $\Delta\phi = a(\eta)/p_T + b(\eta)/p_T^2$

Bobby tried fitting this function for different slices of the bend angle Ψ, as Ming did 4 years ago. The resulting p_T resolution strongly depended on how he did the fits (binning, least squares, chi-squared, etc.)

The resulting p_T resolution often came out worse for than it was originally before trying to add extra information!

- $5<p_T<35$, $1.2<\eta<2$: resolution = 24% before, 25% after
- $5<p_T<9$, $1.2<\eta<2$: resolution = 22% before, 25% after
- Probably the metric for a good fit should be the resulting p_T resolution, not the chi-square, etc.

Brute force approach:

- Just call everything with Ψ=2 low p_T (7 GeV, in fact)
- $5<p_T<10$, $1.2<\eta<2$: resolution = 22% before, 19% after

Simulation Conclusion & Plans

It's clear that adding bend angle information should improve the p_T resolution, and thus reduce the rate

- ➔ Just a matter of how to best use it analytically

I would like Bobby to study patterns separately labeled for bend angles of 2,3 and 4 ½ -strips

- ➔ It is very low p_T muons (<5 GeV) that contribute most of the rate, so adding more information about large bend angles may help separate low from high p_T
- ➔ Could also apply cuts on ψ based on η and $\Delta\phi$

Probably we would propose using 3-bits, and no more than 4-bits, for the number of useful patterns

We next need to port the Bunch Crossing Analyzer that Alex wrote to the full ORCA package, and study it's performance

Bobby meanwhile also should be writing test software…