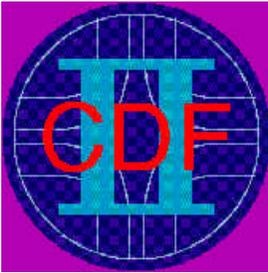


# L2 Software

- Overview
- Algorithms
- Error Handling code
- Code management
- Connection to trigger DB
- Code testing
- Plan, schedule, manpower

Peter Wittich, for L2 software types:  
Stephen Miller, Heather Ray, Masa  
Tanaka, Tom Wright



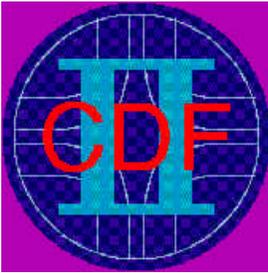
# Overview

## Basic job of processor:

1. Get data from clients
2. Evaluate triggers
3. Report trigger decision

## Implementation:

- C++ code running on  $\alpha$  processor
  - Have most strengths of C++: type safety, inheritance, inlined code
  - Don't use some of the 'costlier' aspects of C++: RTTI, exceptions, virtual inheritance, STL....
- No OS - Evaluation Board Software Development Kit (EBSDK) as pseudo-OS to provide basic services such as console I/O
- Code is compiled on DEC Workstation running OSF 4.0D and same CPU as on  $\alpha$  board.



# Overview

Code runs on  $\alpha$  processor, with a simplified event loop as follows:

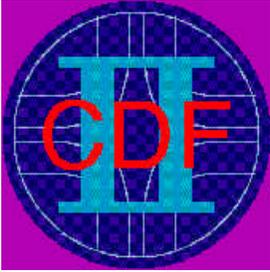
- Initiate data transfer from clients
- Manage DMA FIFOs (receive data from interface boards) to receive next event
- Reformat data into local structures
- Error checking
- Evaluate triggers
- Communicate decision to TS
- On accept, read out RECES/muon board and create TL2D

The strategy is to make the code

1. Correct
2. Robust
3. Fast

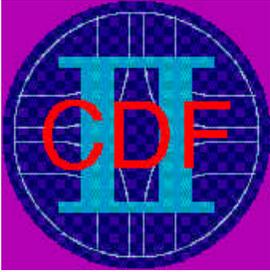
in that order.

Code to be moved to FPGA



# Algorithms

- Necessary algorithms defined by CDF note 4718.
  - ~40 triggers, covered by roughly 20 trigger objects/algorithms with differing options.
  - 4718 'lite' (all the physics, half the code?): an interim selection with somewhat fewer triggers.
- All algorithms  $\alpha$ -independent
  - Allow porting to other platforms and to offline code.



# Currently existing algorithms

We currently have code in CVS for the following sets of triggers.

## Legend:

-  tested successfully
-  being tested
-  code written, not tested
-  placeholder

- AutoAccept 
- CentralElectron 
- Photon 
- GlobalEt 
- HadronicB 
- Jet 
- Dijet 
- L2Error 
- SvtTest 
- SvtTrack 
- SvtXftTrack 



# L2 Software and CDF4718

JSM slide:

L2 options written are highlighted in Blue

Many still need to be tested

	Trigger	L2 Option
1	zerobias	<a href="#">AutoAccept</a>
2	minbias	<a href="#">AutoAccept</a>
3	singletower5	<a href="#">AutoAccept</a>
4	Jet20	<a href="#">Jet</a>
5	Jet50	<a href="#">Jet</a>
6	Jet70	<a href="#">Jet</a>
7	Jet100	<a href="#">Jet</a>
8	HighEtCentralEl	<a href="#">CentralElectron</a>
9	Muon	<a href="#">AutoAccept</a>
10	BtoPiPi	<a href="#">HadronicB</a>
11	Bs	<a href="#">HadronicB</a>
12	Met+2Jets	<a href="#">NJets</a>
13	Met	<a href="#">GlobalMissEt</a>
14	JPsiMuMu	<a href="#">AutoAccept</a>
16	JPsiEE	<a href="#">JPsiEE</a>
17	RadiativeB	<a href="#">ElectronPlusSvt</a>
18	W/ZHiggs	<a href="#">SvtTrack</a>
19	tt->jets	<a href="#">ClusterSum</a>
20	Diffraction	
20.1		<a href="#">AutoAccept</a>
20.2		<a href="#">Jet</a>
20.3		<a href="#">Jet</a>
20.4		<a href="#">AutoAccept</a>
20.5		<a href="#">AutoAccept</a>



# L2 Software and CDF4718

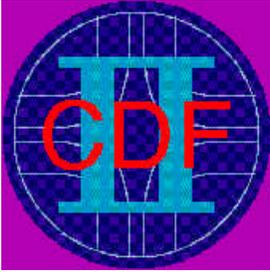
## cont..

PW 7

	Trigger	L2 Option
22	HighEtPhoton_Iso	Photon
23	UHEtPhoton	Photon
24	SHEtCluster	Photon
25	HEtDiPhoton_Iso	Photon
26	LoEtDiPhoton_Iso	Photon
27	Photon+2Jet	Photon
28	Three EmClust	Photon
29	MET + 2btags	SvtTrack
30	MET+PEM	Photon
31	.1 e+isol track	ElectronPlusTrack
31	.2 CMUP+isol track	AutoAccept
31	.3 CMX+isol track	AutoAccept
32	Dileptons	
32.1	CEM+CEM	CentralElectron
32.2	CEM+PEM	ElectronPlusPhoton
32.3	CMUP+CMUP	AutoAccept
32.4	CMUP+CMX	AutoAccept
32.5	CMX+CMX	AutoAccept
32.6	CEM+CMUP	CentralElectron
32.7	CEM+CMX	CentralElectron
32.8	PEM+CMUP	Photon
33	Two XFT Tracks	XFTTrack
34	Photon+muon (charm)	Photon
35	Z->bb	SvtTrack
36	HighPt b-jet	HighPtBjet
37	.1 CMUP+disp. track	SvtTrack
37	.2 CEM+disp. track	ElectronPlusSvt
38	Di-tau	DiTau
39	MET+tau	CentralElectron
41	HiEtPhoton_Iso	Photon
42	W (no track)	Photon
43	e+track (no e iso)	ElectronPlusTrack



Most algorithms have code written for them at varying stages of completion and testing.



# Code Testing

## Right Now:

- Currently, we have two methods of testing code
  - Run it in the detector and check results (CDF as a pulser)
  - Run Masa's test code:
    - Writes L2 raw data into  $\alpha$
    - $\alpha$  runs triggers
    - Read out TL2D and compare to expectation.

With this we can do:

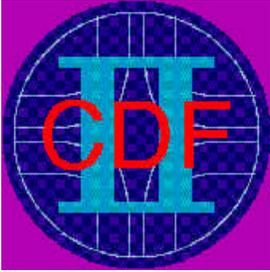
- Correctness checks
- **Simple** robustness and timing checks



## Longer Term:

- Generate test fixture to run code in AC++ environment
- L2\_Pulsar to drive test patterns (similar to Masa's code)

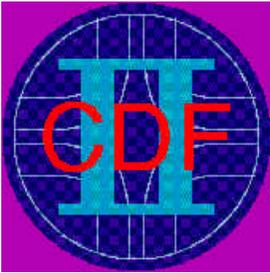
Short of using CDF, no realistic method for getting timing or performance w/o L2\_Pulsar.



# Code Performance: Timing

Only performance tests so far on prototype triggers:

- Time to run single triggers measured on single events
- Time to perform ‘bookkeeping’ measured
- Average performance goal is  $\sim 10 \mu\text{sec}/\text{event}$  for entire event loop (excludes event loading)
- Unclear (to me, anyhow) where we stand with respect to this goal.
  - However, clear paths to improving the software exist, if need be.



# Error Handling

---

L2 Error handling in place and tested

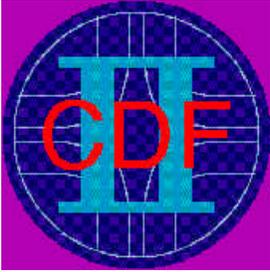
- Code generates L2 timeout for three current error conditions:
  - No L1 magic bus words sent
  - CList buffer number mismatch
  - SVT-XTRP BC mismatch

More error checks can be added pretty easily.

- On error, a message appears in run control explaining the error
- First two run by default in all triggers.

To Do:

- Implement Error triggers
- Switch from L2 timeout to pulling CDFError
  - Code exists, needs to be tested
- Implement error statistics

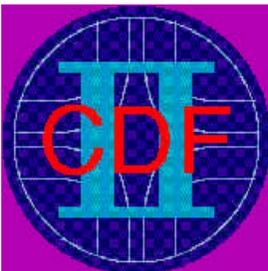


# Code Management

- Code exists in three packages
- All packages under version control system (CVS) in online DAQ repository
- Code specific to hardware implementation separated from trigger algorithms
- Trigger-table specific code limited to two two files that are created by TriggerDB and #include'd into the source code
  - This is the only connection between the trigger DB and the code

## Unfinished business:

- Need CVS package release version numbers in Trigger DB
- Need to split L2 code from L3 code - update L2 code w/o changing L3, others (affects trigger tables?)
- Adapt build procedure to use well-defined CVS version.



# Trigger DB to code

- Trigger table-specific code is included via two files:

define triggers

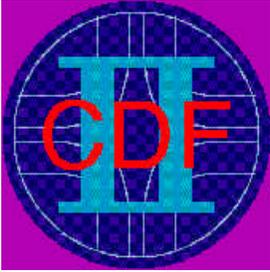
evaluate triggers

```
// TRIGGER TABLE Physics_0_01_v-48  
l2trig_787.trigger();
```

Code is auto-generated  
by Trigger DB GUI.

```
// TRIGGER TABLE Physics_0_01_v-48  
numL2Trigs = 38;  
AutoAccept l2trig_787;  
l2trig_787._l2Bit = 14;  
l2trig_787._l1Bit = 21;
```

```
void mainloop(void) // The main prog.  
{  
  // [code elided]  
  #include "L2cuts.hh"  
  
  MagicBusDMA mbusdma; //see mbusdma.h  
  TSI tsi; //see tsi.h  
  // event loop  
  while (1) {  
    // [code elided]  
    #include "L2triggers.hh"  
  }  
}
```



# Manpower/Schedule

- Identified L2 software writers:

- Stephen Miller
- Heather Ray
- Masa Tanaka
- Peter Wittich
- Tom Wright

However, it is unclear what fraction of these peoples' time will be available to write L2 software.

## Goals:

- Code for 4718 lite by mid-January 2002
- Code for all 4718 triggers by end of January 2002