Test vectors and trigger simulation

- Reminder of scope of work
- Summary of work since Mainz
- Plans
- Details of CP-chip test vectors
- Experience with debugging FPGA
- The mythical FTE

Scope of the online trigger simulation

- To predict expected data at the level that can be read-out to DAQ/VME
  - No sub–40MHz information
  - Algorithms simulated at ‘conceptual’ level
  - Hopefully relatively quick, for use on-the-fly
- To provide a common interface for test-vector generation
  - Generate data that can be fed into
    - a) Hardware
    - b) Simulation to compare results

Summary of work since Mainz

- Some improvements/extensions to framework
- Development of GTM model
  - Before had ‘full’ CPM + crate simulation
- Addition of several sets of CP-FPGA test-vectors
- Code put into RAL cvs repository
- Various bits of tidying up
  - Obeys more ATLAS coding conventions
- Debugging FPGA code with James
  - First practical test of code
  - VHDL vs my simulation useful for picking up errors

Still to do… (lots)

- More CP test-vectors
- Full integration of Bill's work
- Extensive comments and User Documentation
- Serious thoughts about integration with rest of world
  - Only interaction so far through flat files!

CP FPGA test-vectors

- For straight FPGA simulation and GTM set-up we have test-vectors for:
  - ‘Physics’ data (Alan’s original vectors)
  - Random data (carefully tuned)
  - Threshold behaviour
  - Sum testing
- To do:
  - Full BC-demux treatment

Experience with VHDL simulation and real hardware

- Methodology:
  - I sent input file to James
  - James sent results to me
  - FPGA seen as a ‘black-box’ with James as intermediary!
- Looking from the outside:
  - 108 inputs at 160 MHz
  - 40 outputs at 40 MHz
  - ie huge loss of information
- Still possible (but difficult) to diagnose obvious large errors, eg
  - Incorrect overflow behaviour in CP FPGA
  - Reversal of eta in GTM set-up
  - Data for one cell in wrong place in GTM set-up
    - Required VHDL to look internally at signals
- Less obvious problems require more work
  - OK in VHDL, possible in final system?

Conclusion

- Debugging FPGAs is going to be tough!
  - Some of you probably knew this already
  - Need to get as much right with VHDL before we get the hardware
  - Control functions tougher than real-time signals
- Have we got enough tools (manpower) to do it?
  - Chip-scope?
  - Extended (board-level) VHDL simulations?
- Need well tested stable code before we are deluged with lots of boards

The Mythical FTE, eg S.J.Hillier – 100% Level-1 'online software'

- I could have chosen anyone – I just happen to know more about me!
- I chose a particularly exceptional period:
  - 10/9/01 to 9/11/01
  - 2 months, 45 working days
- Activities estimated to nearest day
- 100% sometimes should be interpreted as ~ 10%

Meetings
- attendance – 16
- preparation/minutes – 4

Teaching
- contact hours – 4
- preparation – 6
- marking – 3

Holiday– 5

Admin – 1

Software Development – 6