Concept of a Software Trigger for an Experiment at Tesla

G. Eckerlin, R.Gerhards, R.Heuer, P. LeDu, G. Quast

Outline

- Introduction
- TESLA
- General Concept
- Detector Layout
- Data Acquisition
- Conclusion

ECFA DESY
Linear Collider Workshop

Introduction

TESLA is one of the candidates for a TeV range linear collider

- 2nd ECFA/DESY study for a Linear Electron-Positron Collider
  - Worldwide Studies on Physics and Detectors for a Linear e^-e^+ Collider
  - TESLA Conceptual Design Report (May 1997)
  - TDR for TESLA is due in 2001
  - The proposed concept has implications on the detector design

Most material taken from various studies

- ECFA/DESY study
  - www.desy.de/Conferences/ecfa-desy-98
- World-wide study (last meeting)
  - www.cern.ch/Physics/LCWS99

Evolution of Trigger and Data Acquisition Architecture

- Increasing use of “commodity” components
  - take advantage of industry standards
  - rapidly developing network technologies

    GBit Ethernet

    Switched networks

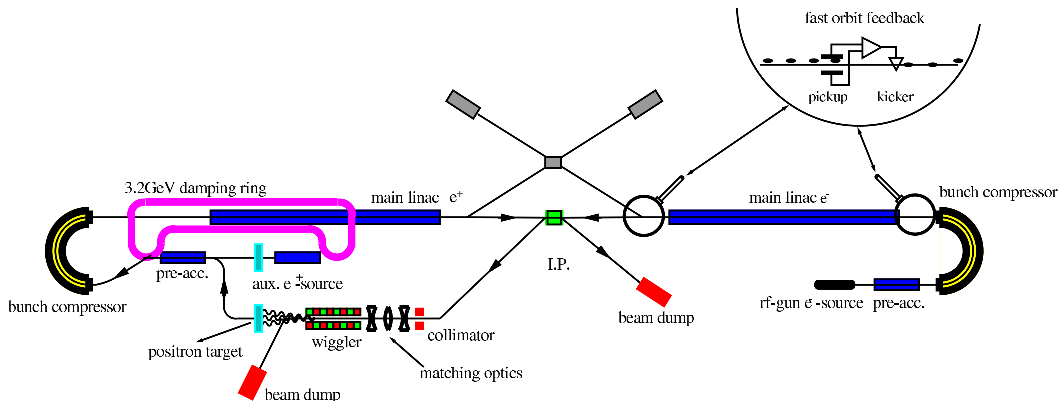
    SAN
  - mainframe like PCs :

    GByte RAM

    GHz CPUs

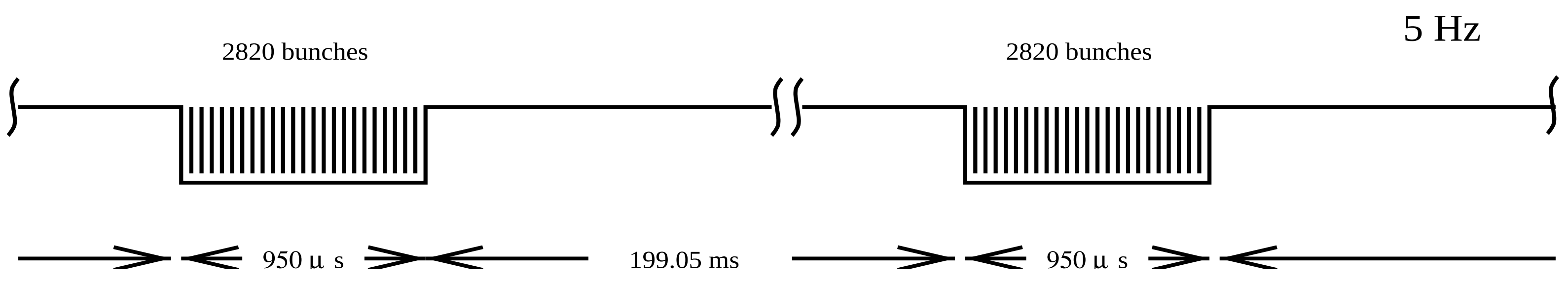
    500MByte/s bus speed

TESLA



- **Options:**
  - $90 < \sqrt{s} < 800$ GeV
  - e^+e^-, e^-e^-, $e\gamma$, $\gamma\gamma$
  - e(TESLA) p(HERA)
  - Luminosity : $3.1 \bullet 10^{34}$ $cm^{-2}s^{-1}$ (at 500GeV)

TESLA operation conditions



- 2820 bunches grouped in trains at a rate of 5 Hz
- Long time between bunch trains: 199 ms
- Rather long time between bunches: 337 ns
- Rather long bunch trains (same order as detector readout time): 1ms

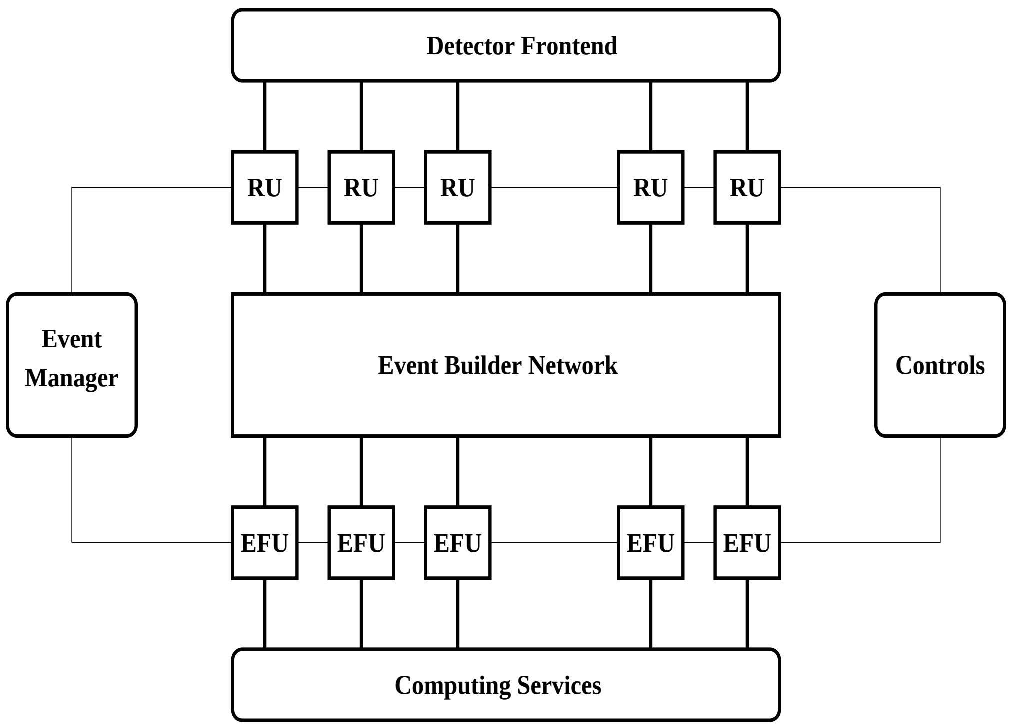
Trigger and DAQ Concept

- AIM:
  - fully efficient and flexible trigger
  - deadtime free readout

    $\Rightarrow$ *no data loss*
  - easy maintenance
  - scalability
  - exploit TESLA operation mode

- CONCEPT:
  - readout and store data of complete bunch train into pipeline
    - no trigger interrupt
    - 1 ms active front-end pipeline
  - perform zero suppression and/or data compression
    - manageable data volumes online
  - apply software selection between trains
    - full event data information of complete train available
    - store classified events according to (physics) needs

Trigger and DAQ Concept Overview



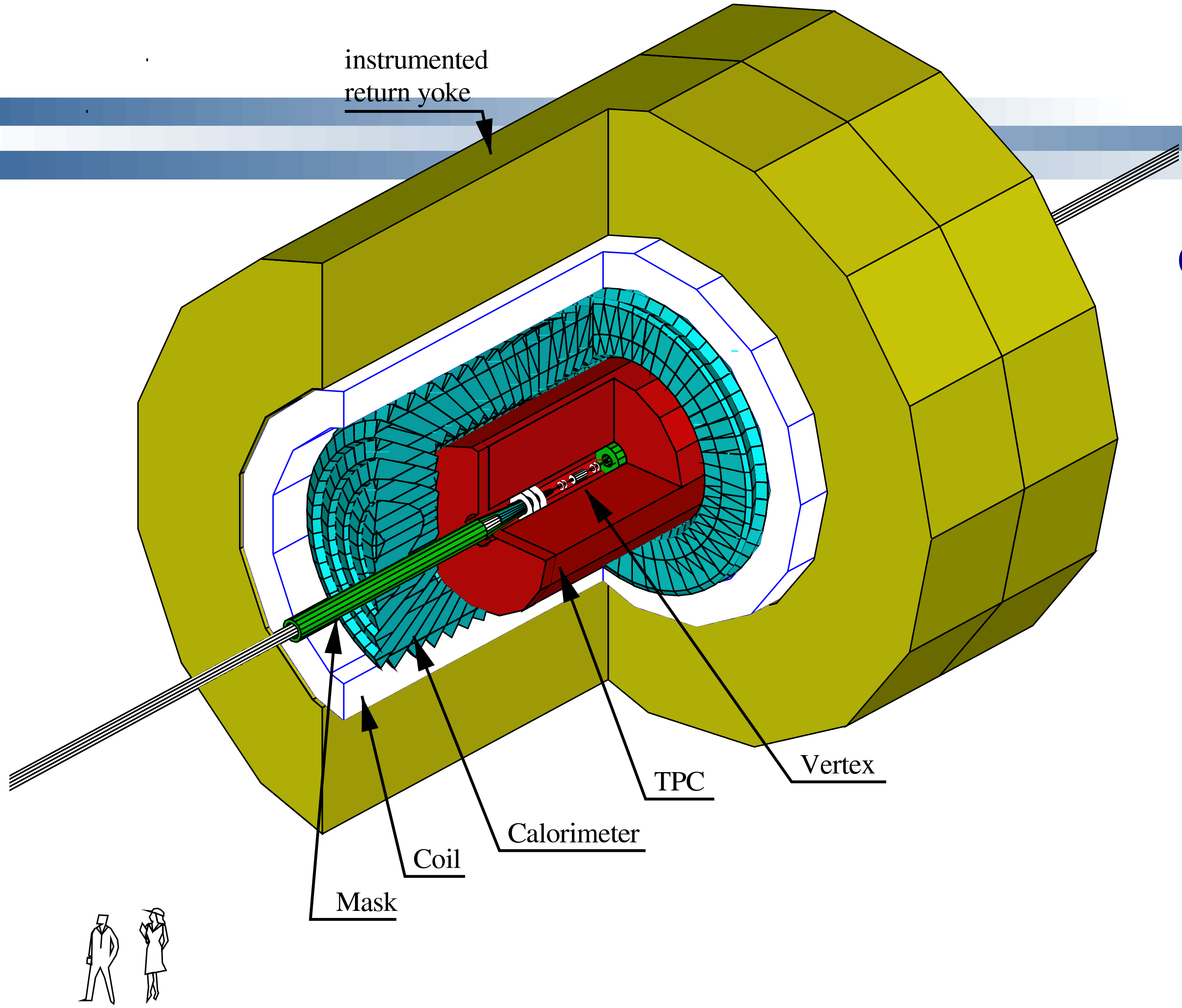
- 1 (or virtually 2) level DAQ system
  - deadtime free
  - no conventional (hardware) L1
  - Readout Unit reads complete train
  - Full event data is built for a train
  - Event selection in Event Finder Unit
  - Store data from ‘bunches of interest‘

Very similar to CMS architecture without L1

Trigger and DAQ Concept Advantages

- Flexibility
  - programmable to a large extent
  - full event information available
  - unforeseen (background or physics) rates easily accommodated
- Ease of maintenance (cost effectiveness)
  - off-the-shelf technology (memory, switches, processors, ...)
  - commonly used operating systems
  - high level programming languages
  - on-line computing resources usable for off-line applications
- Scalability
  - modular system
- Efficiency
  - all data is looked at
- Portability
  - off-line software in on-line environment

Detector Concept



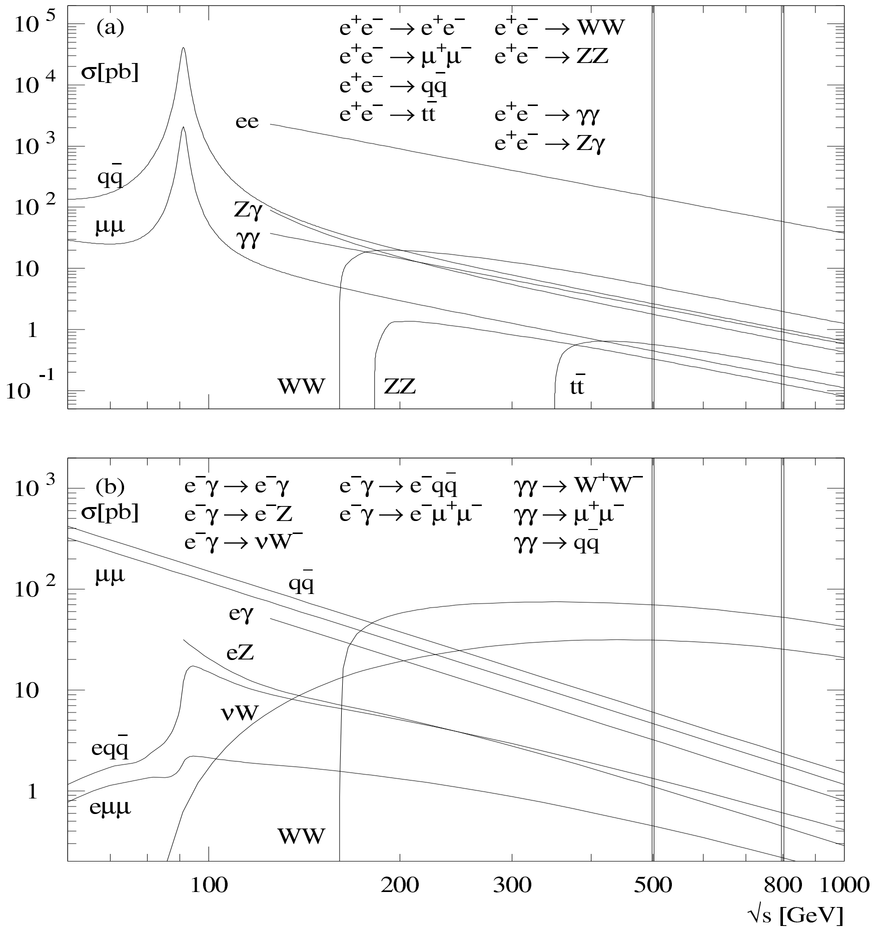
General concept:

- large hermetic detector
  - 5 layer vertex (CCD) ($800 \cdot 10^6$ channels)
  - main tracking (TPC) ($1.5 \cdot 10^6$ channels)
  - ECAL (SiW) ($32 \cdot 10^6$ channels)
  - HCAL, IRON, ...
- total $8.5 \cdot 10^8$ channels

Implications on the Detector Concept

- Constraints on detector readout technologies
  - readout 1 ms continuously
    - digitising during pulse to keep occupancy small (VTX)
    - no active gating for the TPC
    - ...
- Pattern recognition for complete bunch trains
  - parallel processing of trains
    - need farm of EFU
    - one train per EFU
- Feasibility depends on occupancy, in particular in inner VXD layer
- High bandwidth in front-end and event builder
  - few GB/s

Physics and Background Rates



Physics

- $e^+ e^- \to X$ 0.0002/BX
- $e^+ e^- \to e^+ e^- X$ ($\gamma\gamma$) 0.7/BX

Beam induced background:

- neutrons 2000n/BX
- e+e- pairs
  - VXD inner layer 1000 hits/BX
  - TPC 15 tracks/BX
  - ECAL 3000 cells/BX

⟹ *Background dominated !*

⟹ *300 MByte per train (1.5GByte/s)*

Finder Strategy

- Two step process
  - event building and formatting
    - minimize data transfer
  - event classification/finder
    - asynchronous
    - tagging of interesting event ("bunch of interest")
    - maybe several "virtual" levels
- Issues
  - calibration/alignment
  - algorithms
  - selection strategy

Comparison with other Experiments

	present colliders	LHC	LC
event building	250MBytes/s	50GBytes/s	1.5GBytes/s
event processing (reconstruction, filter)	~1s (H1,HERA–B,...)	~1s	~1s
data volume	100TByte/year (Tevatron,...)	1PByte/year	300TByte/year
processing units (for L2/L3)	few hundred (HERA–B)	few thousand	few hundred

- Requirements in the ballpark of experiments which come into operation shortly

Z Factory

- **Physics Rate**
  - e+e- $\rightarrow$ X ~300 Hz
  - track multiplicity ~30
    - 1200 charged tracks per train
    - 0.5 charged tracks per bunch

- **Background Rates**
  - assumed to be similar to rates expected at 500 GeV

- **Results**
  - no significant change in data volume for front-end event building
  - increased finder output due to larger physics rate

- **Note**
  - multi-hadron every 50th bunch
    - expect for TPC all bunches with hits
    - conventional L1 type of trigger won't work deadtime free

Conclusions

Base-line Concept

Deadtime-free software selection

- Considerable flexibility both in design of physics algorithms and in the choice of data used by these algorithms
  - rare physics and unfavourable event topologies can be kept
  - event data storage under control of event filter algorithms written according to needs of physics groups
  - more efficient trigger
- Event building rates are much smaller than at LHC and comparable to those at present experiments
  - feasible - (almost) today
  - less demanding than for LHC
- expected development of computing technology very favourable
  - commodity hardware, high level programming languages and widely used OS will ease scalability and maintenance
  - merging on-line and off-line worlds results in more effective use of resources

$\Longrightarrow$ *Better, more efficient trigger !*

Outlook

Next steps

- watch that (sub)detector design is compatible with the trigger concept
- write a “trigger and DAQ” chapter for the TDR
- perform simulations to proof the concept
- design algorithms and selection strategies
- develop a (common) on-line and off-line computing model
  - event selection/rejection strategies
  - event analysis and processing (common resources, ..)
  - event access