GELATIO
The GERDA framework for digital signal analysis

Matteo Agostini[a], Luciano Pandola[b], Paolo Zavarise[b,c]

[a] Physik-Department E15, Technische Universität München, Germany
[b] INFN Laboratori Nazionali del Gran Sasso, Italy
[c] Dipartimento di Fisica, Università dell'Aquila, Italy

ACAT 2011, Uxbridge, London, UK
September 5-9, 2011

Outline

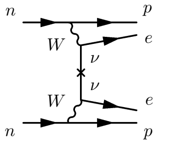
Ref: 2011 JINST **6** P08013 [arXiv:1106.1780]

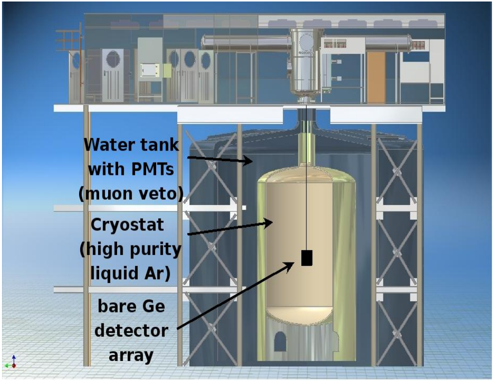
The GERDA experiment @ LNGS

Goal: search for neutrinoless double beta decay ($0\nu\beta\beta$) of ^{76}Ge

- process not allowed in the Standard Model ($\Delta L = 2$) possible only if neutrinos are **massive majorana** leptons
- very rare process: $T^{0\nu}_{1/2}(^{76}\mathrm{Ge}) \geq 10^{25} y$
- $Q_{\beta\beta} = 2039$ keV

$$^{76}_{32}\mathrm{Ge} \longrightarrow ^{76}_{34}\mathrm{Se} + 2\,e^-$$





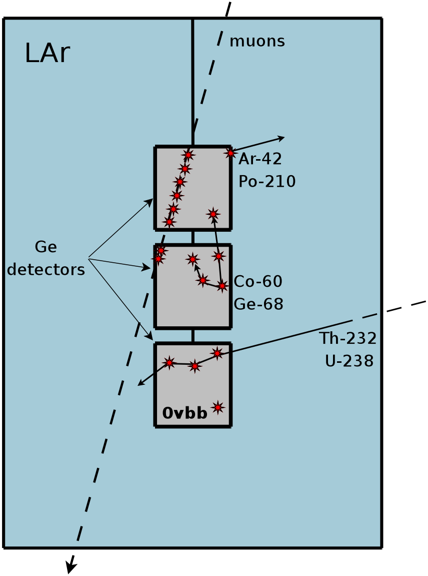
Low Background Design:

- Bare ^{enr}Ge-diodes (87% Ge-76) array in liquid Argon (LAr)
- Shield: high-purity LAr/H_2O
- Radio-pure material selection
- deep underground lab (3.8 km w.e.)

Phase I: $\sim$ 15 kg ^{enr}Ge
up to 12 detectors (8 of ^{enr}Ge)
bkg@$Q_{\beta\beta} < 10^{-2}$ cts/(keV kg y)

Phase II: additional $\sim$ 15 kg ^{enr}Ge
$\sim$ 25 new detectors
bkg@$Q_{\beta\beta} < 10^{-3}$ cts/(keV kg y)

Background



Main background sources:

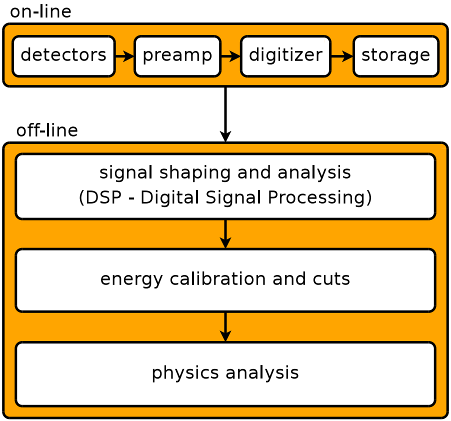
- γ from natural radioactivity (Th-232, U-238)
- γ from cosmogenic isotopes inside the detectors (Ge-68, Co-60)
- β from natural isotope of Ar (Ar-42)
- α from surface contamination (e.g. Po-210)
- non-vetoed μ

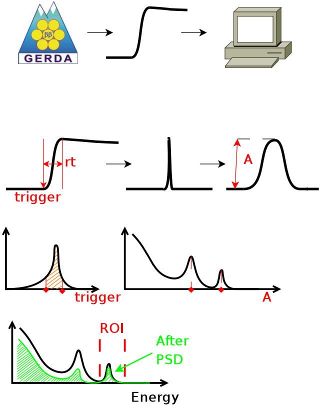
Different event topologies:

- multiple site interactions (μ, γ)
- surface interactions (α, β)
- point-like bulk-volume interactions (γ, $0\nu\beta\beta$)

Pulse Shape Discrimination (PSD) techniques can efficiently suppress background

Data stream





traces digitized @ 100 MHz 160 μs long

background	$\sim 10^2$ event/ch/h	$\sim$ 10 MB/ch/h
calibration	$\sim 10^6$ event/h	$\sim$ 10 GB/h

Analysis software requirements

GELATIO requirements

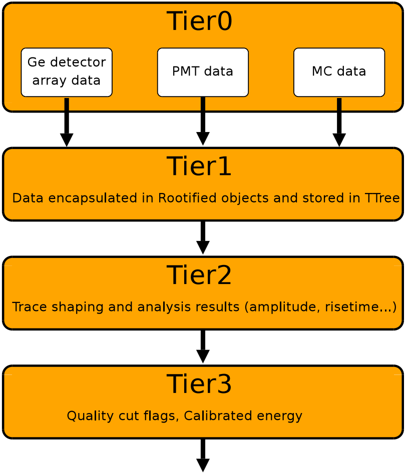
- solid, user-friendly, flexible, maintainable over a long lifetime
- good computational performances
- cross-platform compatibility
- **GERDA common platform for the full off-line analysis:**
  - decouple algorithms from data format
  - highly customizable digital signal processing (DSP)
  - multi-user analysis (sharing and comparison)

Design Paradigms

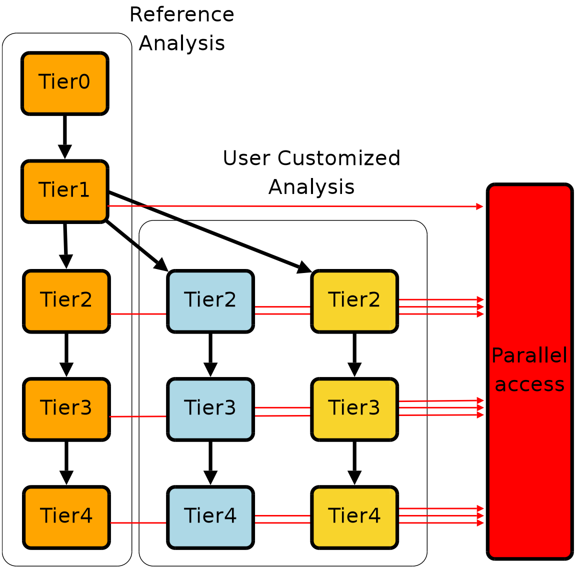
1) multi-tier data organization
2) modular digital signal processing

Multi-tier data structure

- all the raw data (Tier0) are converted into a standard format optimized for storage and analysis (Tier1)
- results of the digital signal processing and analysis are stored in the Tier2
- quality cuts and calibrations based on the Tier2 are stored in the Tier3
- more advanced analysis results (PSD, delayed coincidences. . .) can be stored in higher-level tiers.
- Blinding can be applied in the Tier0 to Tier1 conversion



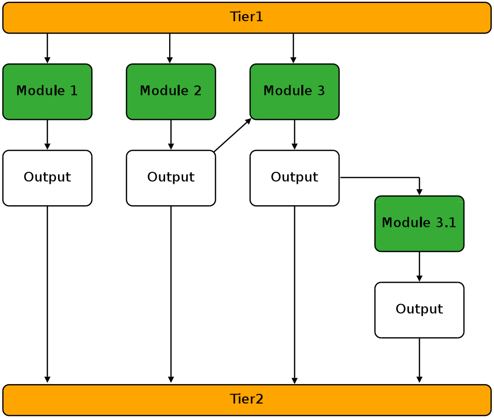
Multi-tier data structure



- alternative analysis as forks of the reference data (multiuser analysis)
- all the information available in parallel (TTree friendship mechanism)
- easy to share and compare customized analysis

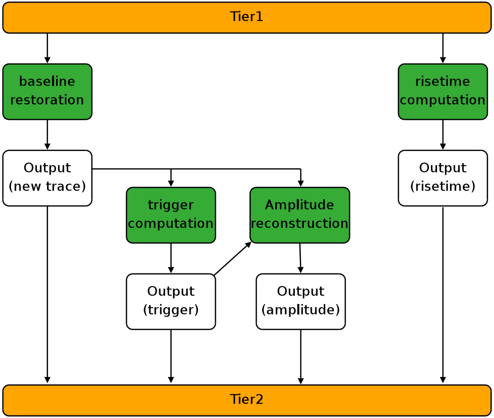
Modular digital signal processing

- each module implements a task of the DSP (trigger, energy. . .)
- computed info and traces can be used as input for other modules
- module chains and internal parameters set via ASCII file
- multiple instances of the same module are allowed
- users can implement new modules and use them together with the existing ones



Modular digital signal processing

- each module implements a task of the DSP (trigger, energy. . .)
- computed info and traces can be used as input for other modules
- module chains and internal parameters set via ASCII file
- multiple instances of the same module are allowed
- users can implement new modules and use them together with the existing ones



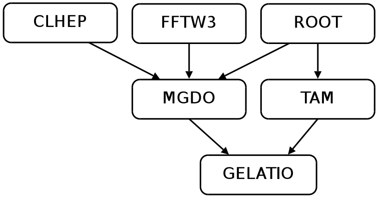
Framework structure and software dependences

GELATIO components

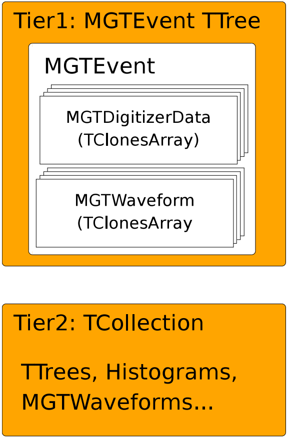
1. two main programs implemented in C++
   raw data → tier1 conversion; tier1→tier2 DSP
2. suite of Bash and Python scripts handling the data streaming through the different tiers
3. GUI implemented by using ROOT graphical components
   event display; event analysis definition and viewer

MGDO: MAJORANA and GERDA libs (containers and DSP algorithms)

TAM: ROOT package providing a modular interface for analyzing data stored in a `TTree`



Data format



Data stored in ROOT TFiles

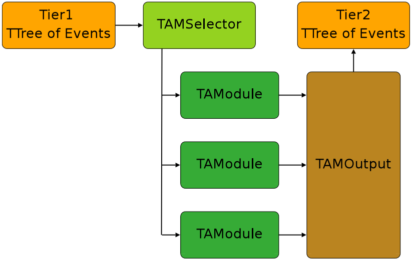
- ROOT compression
- browser
- TTree friendship
  (parallel access info in different tiers)
- exportable
- object streamers (schema evolution)

Tree Analysis Modules (TAM)

TAM: free ROOT package developed by M. Ballintijn , C. Loizides and C. Reed extending `TSelector` functionalities.
http://code.google.com/p/tree-ana-mod/

TSelector

TAM



`TSelector`

- Structured interface to process `TTrees`
  - Begin()
  - SlaveBegin()
  - Process()
  - SlaveTerminate()
  - Terminate()
- PROOF compatible
- no flexible interface for changing `TTrees` and analysis

`TAM`

- TAModules (inheriting from `TTask`) registered in TAMSelector. Same structure of TSelector (Begin(), SlaveBegin(). . .)
- TAMSelector interfaces Modules with the `TTree`
- TAMOutput handles the output (PROOF compatible)

GERDAModule interface

```
class GERDAModule : public TAModule

// Before the event loop
virtual void OnSlaveBegin(..);

// Before the channel loop
virtual void BeginEvent(..);

// Process first trace
virtual void InitializeTransforms(..);

// Process the traces
virtual void ProcessWaveform(..);

// After the channel loop
virtual void TerminateEvent(..);

// After the event loop
virtual void OnSlaveTerminate(..);
```

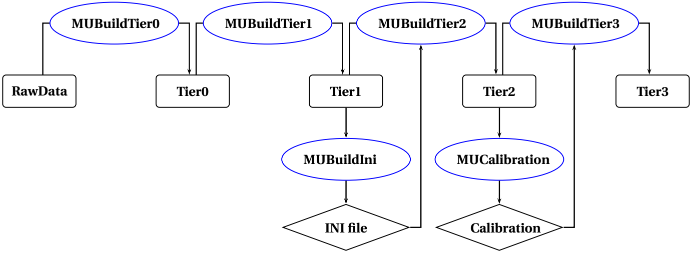
```
//GEMDBaseline.cc

void GEMDBaseline::OnSlaveBegin()
{
   // register output parameter
   fDAEvent->AddParameter<double>
      (fDAVectorName, "baseline");
}
void GEMDBaseline::ProcessWaveform
   (const MGTWaveform& waveform)
{
   // run my algorithm
   double baseline = MyAlgorithm(&waveform);

   // store the value of the parameter
   fDAEvent->SetParameter<double>
      (fDAVectorName, fTaskID,
      "baseline", baseline, fChannelNumber);
}
```

Utilities

```
File system:
/
|-- tier0
|-- tier1
|   \-- out
|-- tier2
|   |-- ini
|   |-- log
|   \-- out
|-- tier3
|   \-- out
\-- calibration
```



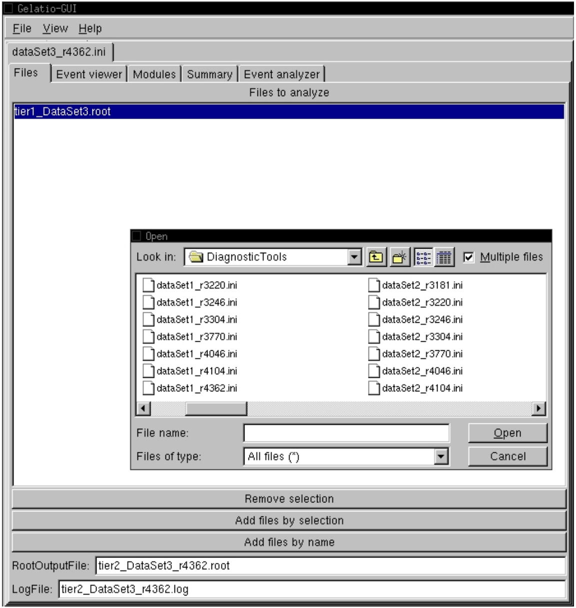
Bash/Python utilities functionalities

- Scan directories checking for new files and process them
- Store the output of each step in a well defined dir (analysis file system)
- Calibrate energy spectra

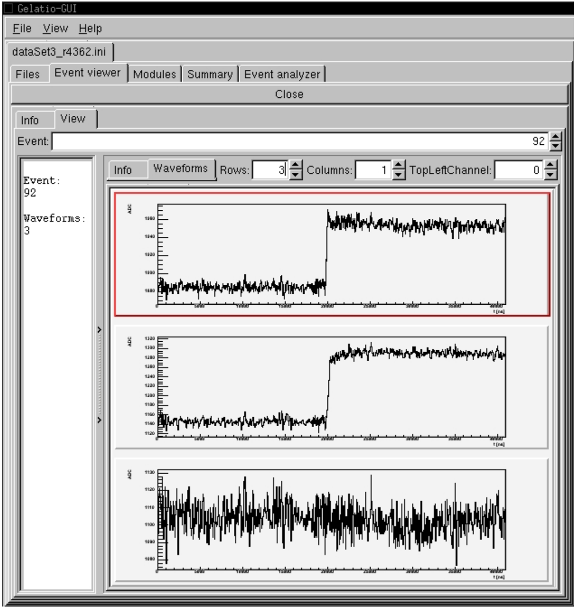
The graphical interface

GUI developed by using only ROOT graphical components (not flexible but avoids further dependence)

- file selection
- event viewer
- interactive creation of the module lists
- DSP viewer
- creation of an ascii file used to run the Tier1→Tier2 analysis



The graphical interface



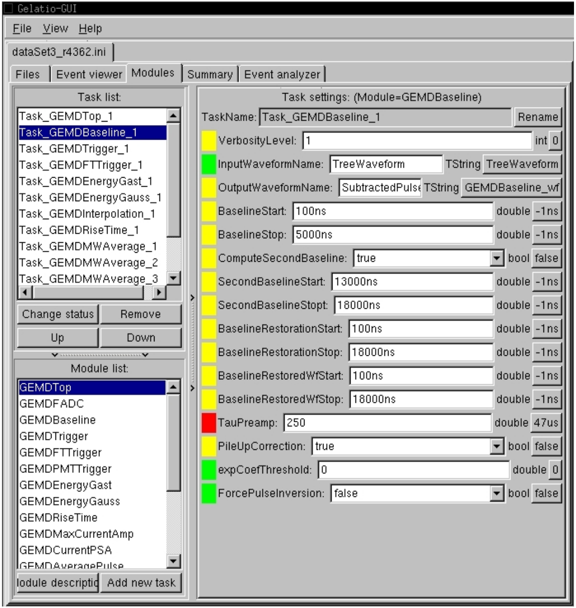
GUI developed by using only ROOT graphical components (not flexible but avoids further dependence)

- file selection
- event viewer
- interactive creation of the module lists
- DSP viewer
- creation of an ascii file used to run the Tier1→Tier2 analysis

The graphical interface

GUI developed by using only ROOT graphical components (not flexible but avoids further dependence)

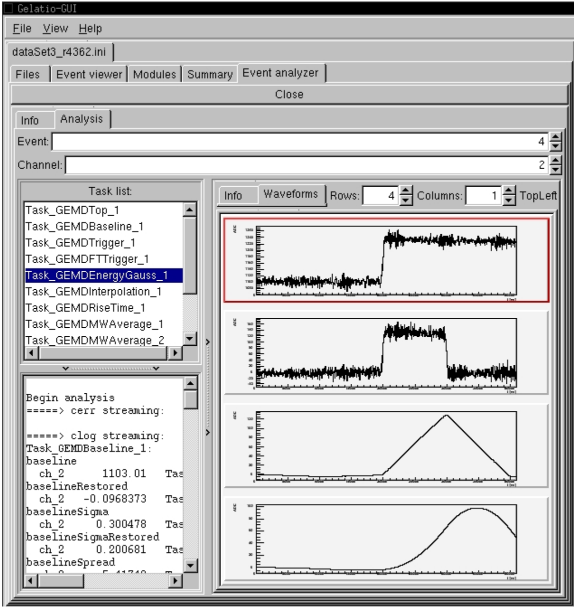
- file selection
- event viewer
- interactive creation of the module lists
- DSP viewer
- creation of an ascii file used to run the Tier1→Tier2 analysis



The graphical interface

GUI developed by using only ROOT graphical components (not flexible but avoids further dependence)

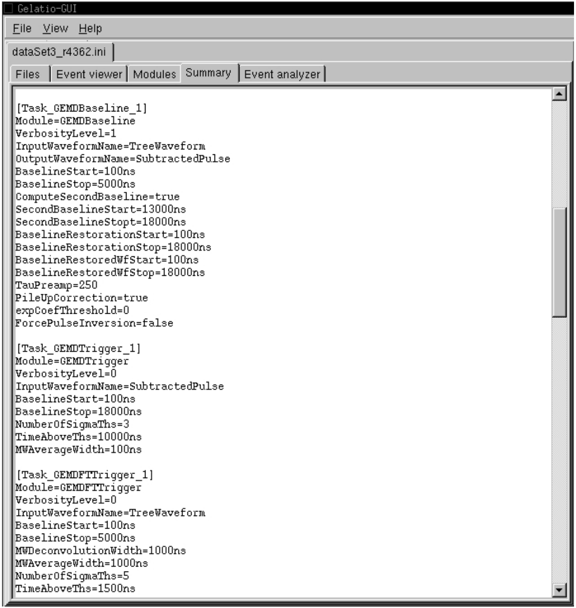
- file selection
- event viewer
- interactive creation of the module lists
- DSP viewer
- creation of an ascii file used to run the Tier1→Tier2 analysis



The graphical interface

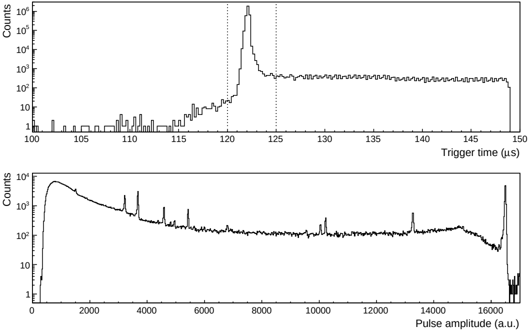
GUI developed by using only ROOT graphical components (not flexible but avoids further dependence)

- file selection
- event viewer
- interactive creation of the module lists
- DSP viewer
- creation of an ascii file used to run the Tier1→Tier2 analysis



Application and Benchmarking

GELATIO status:
- 6 digitizer formats supported
- tested multichannel stream (up to 10 channels)
- reached optimal results with the default algorithms
- more than 15 official modules
- reference data analysis for GERDA & R&D activities



▶ **raw data to tier1 conversion:** ∼ 1 hour per detector for a standard calibration (∼ 600 trace/s with a 2.40 GHz CPU, compression level 1)

▶ **Tier1 → Tier2 DSP:** ∼ 3 hours per detector with the standard analysis (∼ 200 trace/s with a 2.40 GHz CPU)

Conclusions

Summary

- developed new tool for digital signal processing and analysis
  - possible to handle different raw data formats
  - multichannel support
  - highly customizable analysis
  - user-friendly software (utilities & GUI)
  - easy to implement new algorithm
- stable releases available for the GERDA collaboration
- used for the reference GERDA analysis