Разработка программного обеспечения для системы сбора данных электромагнитного калориметра детектора Belle II

Семинар ОФВЭ ИЯИ РАН, 20 ноября 2023 г.

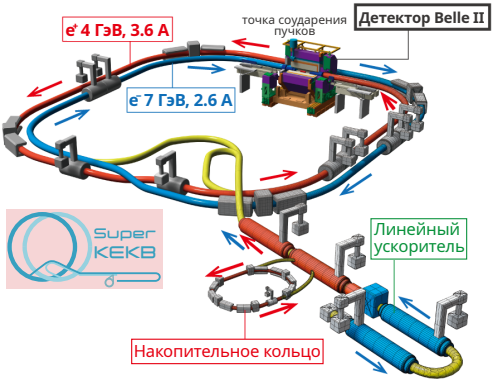
Докладчик: Ремнев Михаил Анатольевич

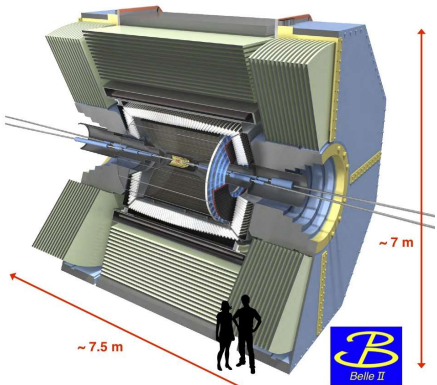
Специальность: 1.3.2 Приборы и методы экспериментальной физики

Научный руководитель: д.ф-м. н., г.н.с. ИЯФ СО РАН Кузьмин Александр Степанович

Эксперимент Belle II

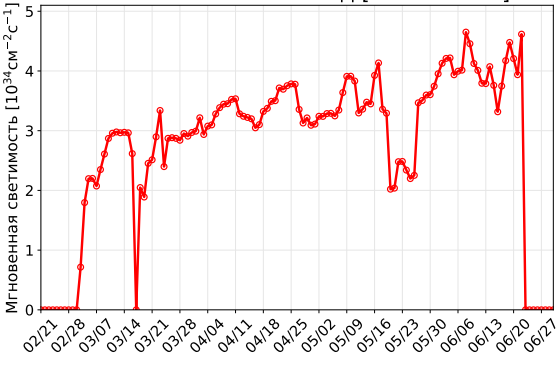
- Цель эксперимента: исследование CP-нарушения в распадах B- и D-мезонов, физики B- и D-мезонов, τ-физики, поиск Новой физики.
- Эксперимент проводится на SuperKEKB — асимметричном e^+e^- коллайдере с энергиями пучков $E_{e^-} = 7$ ГэВ, $E_{e^+} = 4$ ГэВ.
- Проектная светимость $6 \cdot 10^{35} \text{с}^{-1}\text{см}^{-2}$ в 30 раз выше, чем в эксперименте Belle.
- Электромагнитный калориметр (ECL) — важная система детектора Belle II для измерения энергии нейтральных частиц.

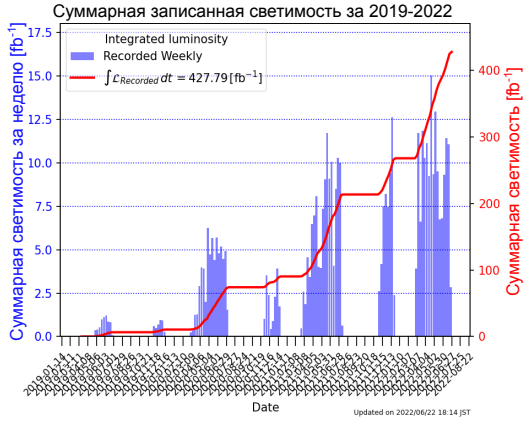




Текущий статус эксперимента

- Первые физические данные были набраны в 2017 году.
- В 2019 году эксперимент вступил в основную фазу — набор данных со всех подсистем детектора.
- В июне 2020 года была достигнута и превышена рекордная светимость эксперимента Belle.
- Опубликованы первые физические результаты, ведётся анализ набранных данных.





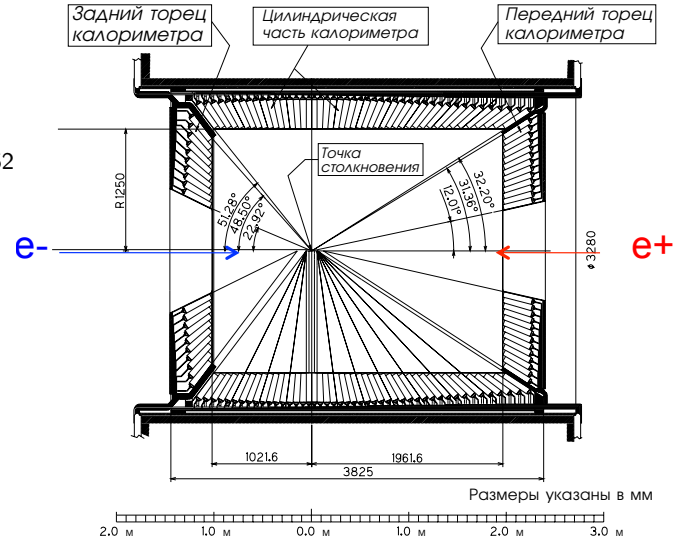
Сейчас SuperKEKB остановлен для плановых технических работ и продолжит работу в марте 2024 года.

Электромагнитный калориметр (ECL)

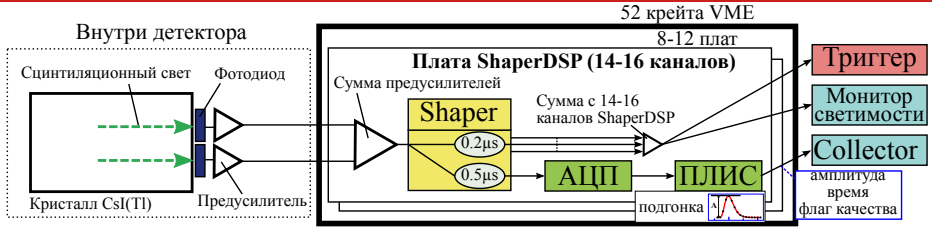
- Калориметр состоит из 8736 кристаллов CsI(Tl) (6624 в цилиндрической части, 1152 в переднем торце, 960 в заднем торце).

Основные функции электромагнитного калориметра (ECL):

- Измерение энергии и координат γ в широком диапазоне энергий.
- Выработка сигналов для нейтрального триггера.
- Измерение светимости.



Электроника калориметра

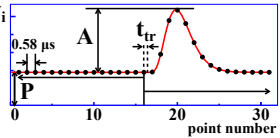


1. Данные с ECL поступают в модули ShaperDSP, где выполняется формирование, оцифровка и аппроксимация сигнала.
2. Модули ShaperDSP укомплектованы ПЛИС, в проекте которых по приходу триггера происходит подгонка данных с 8-16 каналов. Используется метод минимизации χ^2.
3. Коэффициенты, использующиеся в подгонке, вычисляются заранее на основе калибровки формы сигнала. Их объём составляет ~0.5 МБ на плату.

$$\chi^2(A,p,t_{tr}) = \sum_{i,j} (y_i - Af(t_i - t_{tr}) - P)\, S_{ij}^{-1}\, (y_j - Af(t_j - t_{tr}) - P) \to min$$

$$S_{ij} = \overline{(y_i - \overline{y})\,(y_j - \overline{y})}$$

$f(t)$ — функция отклика



$$Af(t_i - t_{tr} - \Delta t) = Af(t_i - t_{tr}) - A\Delta t f'(t_i - t_{tr}) = Af(t_i - t_{tr}) + Bf'(t_i - t_{tr})$$

где t_{tr} — триггерное время, Δt — подгоняемый параметр

$$\sum_{i,j} f_i S_{ij}^{-1}(y_j - Af_j - Bf'_j - P) = 0 \qquad A = \sum_i \alpha_i y_i$$

$$\sum_{i,j} f'_i S_{ij}^{-1}(y_j - Af_j - Bf'_j - P) = 0 \qquad B = \sum_i \beta_i y_i \Rightarrow \Delta t = -B/A$$

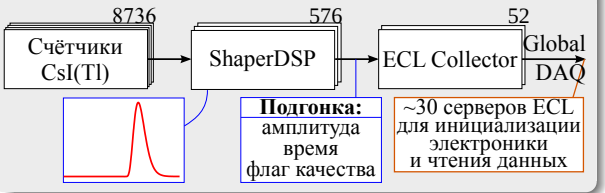
$$\sum_{i,j} S_{ij}^{-1}(y_j - Af_j - Bf'_j - P) = 0 \qquad P = \sum_i \gamma_i y_i$$

Система сбора данных ЭМ калориметра (ECL)

Для 576 плат ShaperDSP объём коэффициентов составляет ~300 МБ. Кроме того, конфигурация ShaperDSP и ECLCollector включает ~80'000 регулярно обновляющихся параметров:

- Калибровочные коэффициенты для быстрой суммы каналов ShaperDSP, использующиеся для триггера и монитора светимости.
- Энергетические пороги на запись данных и на подгонку времени.
- Параметры алгоритмов обработки данных.

Поток данных с калориметра



Параметры системы сбора данных и ECL

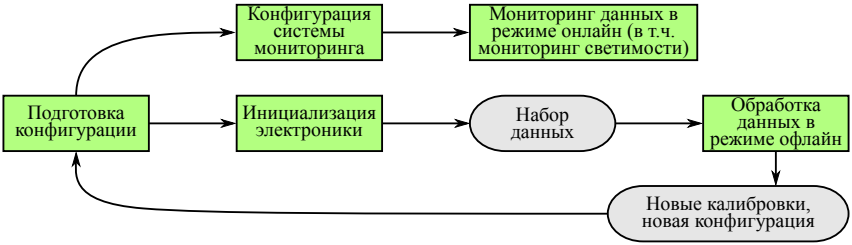
Частота первичного триггера на проектной светимости	30 кГц
Данных ECL на событие (32 бита/канал: амплитуда, время, качество фита)	12 кБ
$\sigma E/E$ при $E > 1$ ГэВ	~2%

Программное обеспечение системы сбора данных (DAQ) должно автоматизировать процесс управления конфигурацией ECL и удовлетворять следующим требованиям:

1. Обеспечение стабильности сбора данных.
2. Обеспечение качества данных.
3. Организация системы по управлению сбором данных.
4. Организация рабочего процесса по поддержке системы.

Цель и задачи

Цель работы — разработка программного обеспечения (ПО) для системы сбора данных (DAQ) электромагнитного калориметра (ECL), в частности систем медленного контроля и мониторинга.

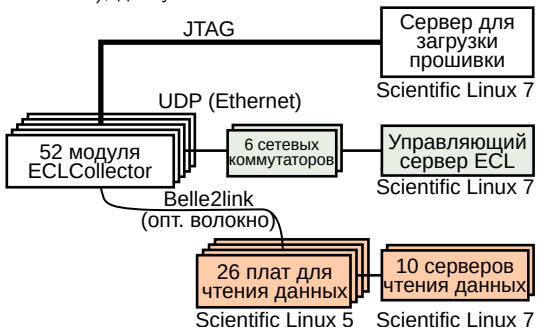


1. Разработка ПО для инициализации электроники.
2. Разработка ПО для управления конфигурациями калориметра.
3. Разработка модулей системы медленного контроля.
4. Создание системы мониторинга качества данных ECL в режиме онлайн.
5. Разработка системы DAQ для монитора светимости, отдельного модуля ECL.
6. Разработка инструментов для диагностики качества данных ECL в режиме офлайн.

Инициализация электроники (1/4)

- Перед началом считывания данных необходимо подготовить электронику к работе, проведя инициализацию модулей ECLCollector и ShaperDSP.
- Для этого нужно использовать соединения по JTAG, Ethernet и оптическому волокну, согласованно запуская процессы на 38 серверах (12 стандартных серверов и 26 плат с Scientific Linux 5).
  - Основные шаги инициализации могут выполняться как по Ethernet, так и по оптическому волокну. Однако есть малый набор функций (таких как работа с флэш-памятью), доступных только по Ethernet.

1. Программа инициализации должна поддерживать как соединение по Ethernet, так и по оптическому волокну.
2. Для ускорения процесса инициализации модули ECLCollector нужно инициализировать параллельно.
3. Чтобы быстро находить ошибки, должно вестись детально настраиваемое логирование.
4. Процесс инициализации должен быть гибким. Должна поддерживаться загрузка параметров инициализации из разных источников.

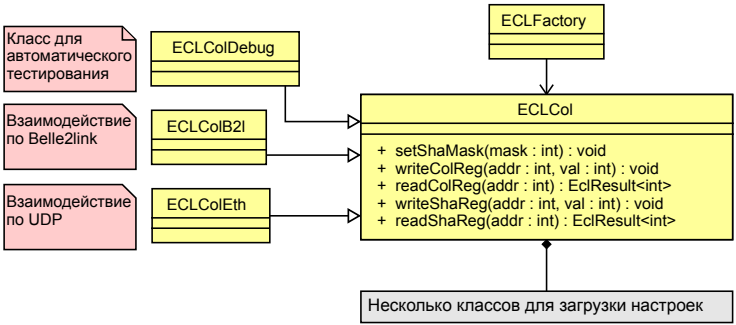


Инициализация электроники (2/4)

- Процедура инициализации должна легко модифицироваться и выполняться за короткий промежуток времени (< 5 минут).
  Поэтому была разработана библиотека C++, интегрирующая всё ПО для инициализации ECL.
- Разработанная библиотека включена в репозиторий медленного контроля.

Эта библиотека позволяет абстрагироваться от:

- Способа инициализации (Ethernet/Belle2link).
- Цели логирования (DAQ DB, файл, stdout).
- Хранилища конфигурации (DAQ DB provider, DAQ DB, файл).



- Архитектура библиотеки позволяет легко настраивать и расширять процесс инициализации.
- Библиотека может выполнять реорганизацию низкоуровневых запросов, что позволяет повысить быстродействие на $\sim$30%.

Инициализация электроники (3/4)

- При изменениях в системе сбора данных (обновляется прошивка модулей ECL, меняются платы для чтения данных) требуется модифицировать библиотеку инициализации.
- Важно тщательно проверять код каждой новой версии на работоспособность.
- Чтобы минимизировать риск ошибки, был разработан класс ECLColDebug и несколько автоматизированных тестов с его использованием.

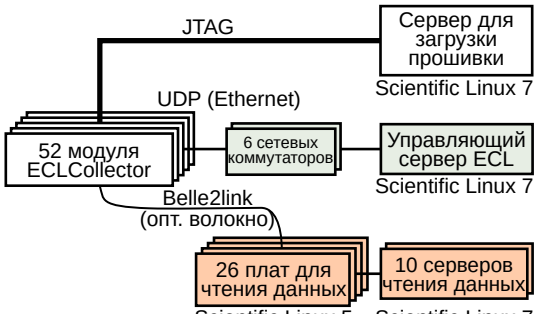
- Класс ECLColDebug эмулирует работу некоторых стандартных функций прошивки ECLCollector, в частности запись регистров по прямой и косвенной адресации.
- Это позволяет автоматически тестировать новые версии библиотеки инициализации.
- Любая утилита из библиотеки может быть запущена в тестовом режиме.
- При сборке проекта доступен опциональный шаг, в котором утилита gcov из GNU Compiler Collection определяет процент покрытия тестами, позволяя понять, какие фрагменты кода не были проверены автоматически.

Отчёт о покрытии тестами

Filename	Line Coverage		Functions	
ecl_col.cc	**22.4 %**	100 / 446	**36.7 %**	11 / 30
ecl_col.h	**50.0 %**	3 / 6	**0.0 %**	0 / 2
ecl_col_debug.cc	**84.5 %**	49 / 58	**77.8 %**	7 / 9
ecl_event_listener.h	**100.0 %**	1 / 1	**-**	0 / 0

Инициализация электроники (4/4)

- Разработанная библиотека используется утилитами на высокоуровневых языках программирования, которые параллельно инициализируют все модули ECL, используя соединения по JTAG, Ethernet и оптическому волокну.
- Инициализацию можно выполнять для заданной группы модулей (к примеру, для отдельной триггерной группы).
- Полная инициализация ECL занимает ~1 минуту.
- Большинство (~90%) известных сбоев в процессе инициализации автоматически исправляются без вмешательства эксперта.



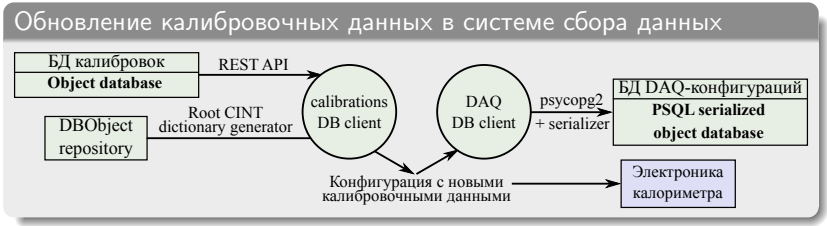
- Для инициализации ECL требуется большое число параметров, конфигурация должна храниться в базе данных.

Управление конфигурациями калориметра (1/2)

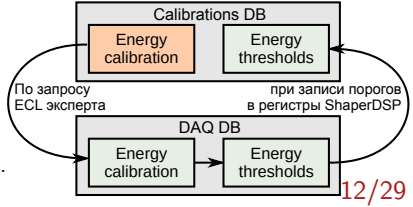
В эксперименте Belle II используются две базы данных:

- Calibrations DB (БД калибровок)
- DAQ DB (БД с текущей конфигурацией электроники)

Из-за различий в выполняемых задачах, эти две БД имеют существенно разную структуру, поэтому были разработаны утилиты, позволяющие конвертировать информацию между ними.

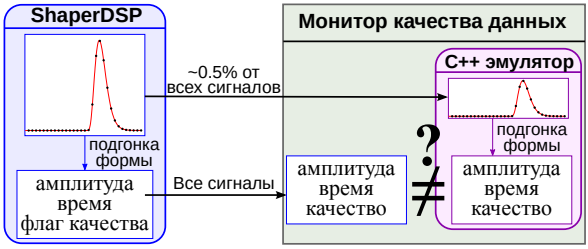


- Утилиты зависят от минимального количества сторонних модулей, что упрощает их использование на любом сервере в сети DAQ.
- Конфигурация синхронизуется в обе стороны:
  - На основе калибровочных коэффициентов в DAQ DB генерируется новая конфигурация электроники.
  - При записи конфигурации в электронику обновляется БД калибровок. Записанные параметры используются в моделировании.



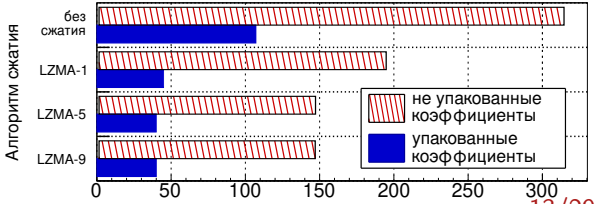
Управление конфигурациями калориметра (2/2)

- Подгонка формы сигналов, поступающих с каналов калориметра, осуществляется в ПЛИС 576 модулей ShaperDSP.
- Ионизирующее излучение и шумы в электронике могут вызвать сбои в логике ПЛИС, которые необходимо оперативно выявлять.
- Поэтому ~0.5% сигналов повторно подгоняется в программе-эмуляторе, встроенной в монитор качества данных (DQM).



Реализована процедура синхронизации массивов коэффициентов, использующихся в алгоритме подгонки, между DQM и электроникой.

- Чтобы ускорить передачу коэффициентов и уменьшить их размер, был разработан специальный алгоритм для их упаковки.
- Его использование позволило в 3 раза улучшить эффективность сжатия данных. (~150 МБ → ~40 МБ).



Модули медленного контроля

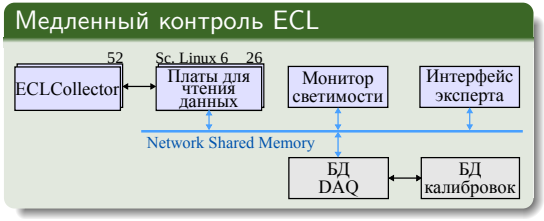
Медленный контроль в эксперименте Belle II ориентирован на следующие задачи:

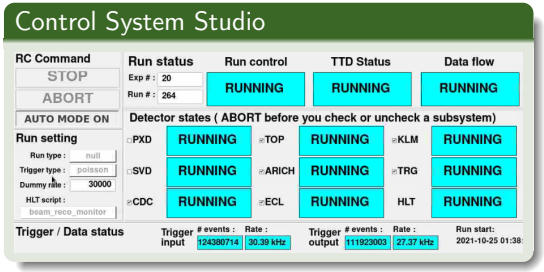
- Запуск инициализации электроники.
- Запуск и остановка чтения данных.
- Отображение статуса сбора данных.
- Проведение калибровок.
- Мониторирование температуры, влажности и т.д.

Для их реализации используется фреймворк Network Shared Memory 2 (NSM2), написанный на языке C.

Для отображения информации с NSM2 используется фреймворк Control System Studio.

- Виджеты написаны на Java.
- Внутренняя логика на JPython.





- Чтобы интегрировать медленный контроль с программами на высокоуровневых языках программирования, разработан модуль PyNSM2, использующий Python для доступа к узлам NSM2.

Реализация PyNSM2 (1/2)

Было рассмотрено несколько вариантов реализации PyNSM2. Важными критериями было отсутствие внешних зависимостей и относительная простота поддержки.

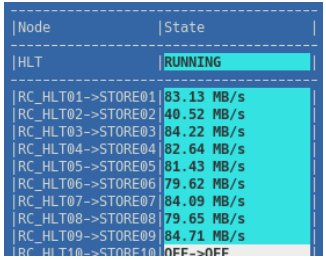
1. Python C API.
   - Требует заголовочные файлы Python (зачастую доступны по умолчанию).
   - Сравнительно сложен в разработке.
   - Требует дополнительный шаг сборки.
2. Boost.
   - Требует библиотеку Boost, которая на некоторых серверах в сети Belle II недоступна.
   - Требует дополнительный шаг сборки.
3. SWIG.
   - Требует утилиту SWIG (почти везде доступна).
   - Требует написания промежуточных swig-файлов.
   - Требует дополнительный шаг сборки.
4. Модуль ctypes.
   - Требует Python ctypes (доступен по умолчанию с версии Python $\geqslant$ 2.6).
   - Требует написания промежуточных файлов для структур C (structs).
   - Не требует дополнительного шага сборки.

Модуль ctypes относительно легко поддерживается и при этом не требует отдельного шага сборки. При этом ожидается потеря производительности, но тесты показали, что она не значительна для утилит медленного контроля (основная потеря производительности: дополнительные 0.3 секунды на запуск приложения).

Реализация PyNSM2 (2/2)

- Конечная реализация использует библиотеку ctypes вместе с pycparser, который используется для генерации определений функций и структур C.
- В результате, обновления NSM2 легко включаются в PyNSM2 за счёт автоматической обработки кода на языке C.
- Где возможно, PyNSM2 использует те же участки памяти, что NSM2, позволяя пропускать конверсию данных и тем самым улучшать быстродействие модуля.

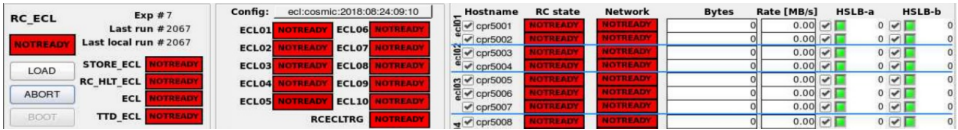
```
import nsm2

nsm2.init('MY_PROCESS')
nsm2.register('ok')
nsm2.register('error')
nsm2.register('vset')
state = nsm2.vget('RUNCONTROL', 'rcstate')
```



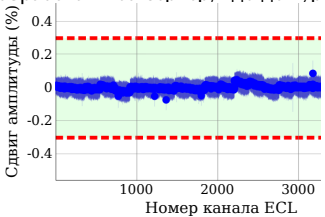
- PyNSM2 добавлен в репозиторий медленного контроля.
- Реализован набор автоматизированных тестов.
- Разработанная библиотека используется для отправки автоматических оповещений о статусе сбора данных.

Калибровка по тестовому сигналу (1/2)

- Чтобы обеспечить стабильность электроники, необходимо регулярно проводить процедуру калибровки.
  - Для этого дежурные ежедневно проводят специальные локальные заходы.
  - Эта процедура выполняется в графическом интерфейсе, использующем CS Studio.

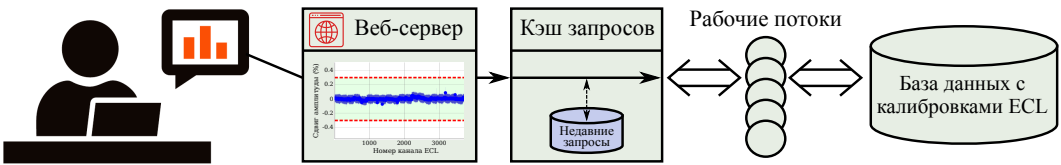


- Программа выполнения калибровок основана на стандартном клиенте управления локальными заходами Belle II, к которому было добавлено несколько важных для ECL функций:
  - Отслеживается, что все модули ECLCollector используют одну конфигурацию.
  - Информация о набранных калибровках записывается в отдельную БД.
- Обработка калибровочных данных полностью автоматизирована.
  - Автоматически выполняется обработка и перенос калибровочных данных.
  - Разработан веб-сервер, где дежурные могут посмотреть результаты по всей истории калибровок.



Калибровка по тестовому сигналу (2/2)

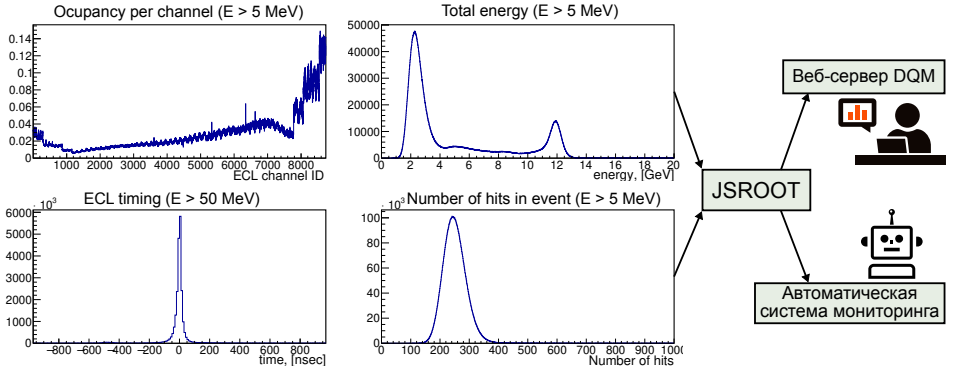
1. Для отображения калибровочных данных используется Javascript-библиотека Dygraphs, предоставляющая наилучшее быстродействие в отображении графиков.
2. Запросы к базе данных кэшируются для ускорения доступа к актуальной калибровочной информации.
3. Чтобы быстро обрабатывать запросы от произвольного числа подключенных экспертов, используется пул рабочих потоков, которые параллельно работают с базой данных.



- Запросы соответствуют стандарту REST API, поэтому при необходимости этими же данными легко могут пользоваться другие веб-сервера в сети Belle II.

Монитор качества данных (DQM)

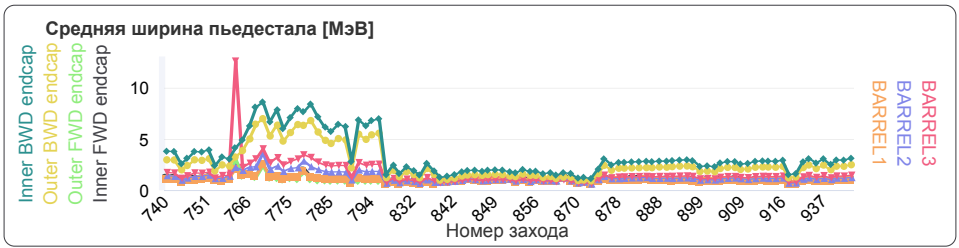
Чтобы обеспечивать корректность записываемых данных, в Belle II используется монитор качества данных (DQM).



- Используется ∼20 гистограмм для мониторинга различных параметров ECL.
- Основные распределения отображаются в пультовой, остальные регулярно проверяются дежурными по ECL.
- Интегральная информация передаётся в систему медленного контроля и используется в мониторинге.

Интегральная информация по качеству данных

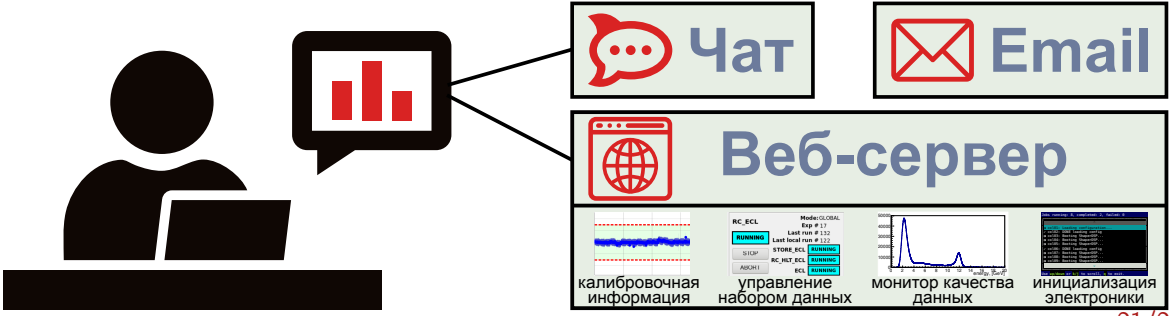
- На отдельный сервер передаётся интегральная информация о ширине пьедесталов в зависимости от времени.
- Графики ширины пьедесталов используется для формирования первичных выводов о влиянии пучкового фона на качество данных ECL.



- Эти же величины передаются в систему медленного контроля и отображаются в пультовой ускорителя.

Обеспечение стабильности сбора данных

- Эксперты по ECL круглосуточно отслеживают статус калориметра, обеспечивая работоспособность сбора данных.
- Для оперативного реагирования на возможные сбои системы разработан веб-сервер, предоставляющий доступ ко всем описанным выше инструментам.
- Разработана автоматическая система оповещений:
  - Проверяется корректность гистограмм качества данных и состояние серверов.
  - Предоставляются оповещения по email и в чате Belle II.
- Написана детальная документация и набор инструкций для дежурных по ECL.



Обнаружение и диагностика ошибок

- Поскольку ПО для системы сбора данных ECL состоит из большого числа компонент, важно иметь детальную документацию, позволяющую дежурным оперативно находить сбои в системе.
- Помимо основной документации, написан ряд инструкций для стандартных процедур диагностики ошибок.
- Регулярно проводится инструктаж для дежурных экспертов по ECL.

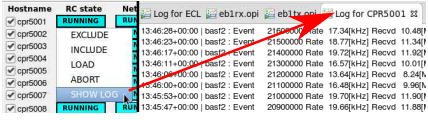
b2ecl bot <b2ecl-ecl01@____kek.jp>
Dear ECL expert,

At Thu Jul 26 00:20:02 JST 2018, automatic checkup script detected that disk space usage for ecldaq@eclpc13:/media/Backup_Data rose above the threshold of 90%.

Detailed status:
ecldaq@eclpc13:/media/Backup_Data : 93%

ecl_bot
ShaperDSP corruption was detected in ECL
logic summary DQM histogram
Bins with errors (crate,shaper) = (6, 11).

Please ask CR shifter to stop the run.
After the run is stopped,
run **`ecl-init-shaper 45`** script ...

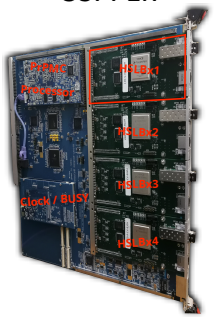


- В тексте автоматических оповещений по email и в чате всегда указаны шаги по исправлению проблемы или ссылки на документацию.
- Созданы утилиты для доступа к логам различных компонент системы сбора данных. Добавлено отображение лога ошибок в GUI управления заходами.

Обновление глобальной системы сбора данных

- В 2022 году была существенно обновлена общая система сбора данных, считывание данных теперь проводится платами PCIe40, имеющие более высокую пропускную способность по сравнению с предшествующими платами COPPER.
- Это означало изменения в ПО инициализации и медленного контроля.
- Используя существующие решения, перенести все разработанные библиотеки с COPPER на PCIe40 оказалось относительно просто и не заняло много времени.

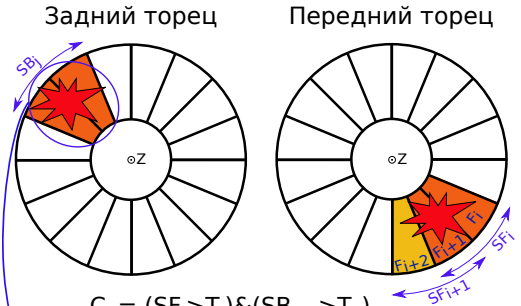
COPPER

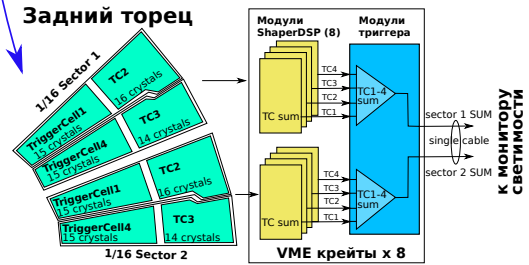


PCIe40

- Обновлены низкоуровневые и высокоуровневые утилиты инициализации, с сохранением обратной совместимости.
- Обновлён графический интерфейс и все модули медленного контроля.
- Обновлён распаковщик событий.

Монитор светимости (LOM)





- Одна из важных задач, которую позволяет решить ECL – измерение светимости.
  Для этого быстрые суммы входных сигналов с ShaperDSP отправляются в модуль монитора светимости (LOM).
- LOM определяет светимость по событиям e^+e^--рассеяния.
- Критерии детектирования события: большое энерговыделение в торцах и правильная топология (энерговыделение в противоположных секторах).

- Разработан пакет программ для передачи данных с LOM в систему медленного контроля.

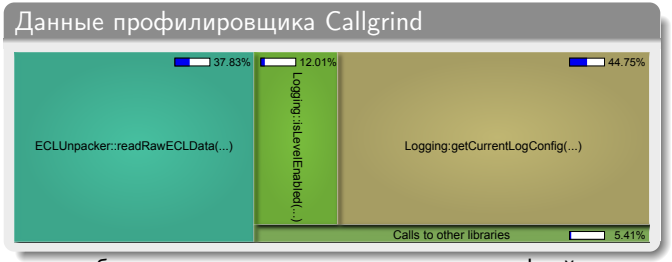
Монитор светимости (LOM)

- Разработана библиотека для взаимодействия с модулем онлайн-монитора светимости (LOM).
- Разработан промежуточный сервер, осуществляющий кэширование запросов к LOM и осуществляющий запись информации о светимости на диск.
- Реализован экспорт данных в систему медленного контроля.
- Автоматизирована процедура энергетической калибровки LOM.

Обработка данных с калориметра (1/2)

- Чтобы использовать заметную долю данных для мониторинга, важно, чтобы обработка информации со всех систем проходила максимально быстро.
  - Данные с систем детектора считываются в сыром универсальном формате, потом происходит распаковка этих данных, что может занимать долгое время.
- Для этого была проведена оптимизация распаковки сырых данных с ECL.

- С помощью фреймворка Valgrind проведено профилирование кода распаковщика событий.
- Определены наиболее критичные для оптимизации функции.



- Были обнаружены проблемы оптимизации с логированием в фреймворке анализа данных Belle II. Предложено и добавлено несколько исправлений к ядру фреймворка.
  Это позволило повысить скорость распаковщика данных с ECL на ~50%.
- Была оптимизирована передача сырых данных, что привело к дополнительному ускорению распаковщика на ~20%.

Обработка данных с калориметра (2/2)

- Информации с монитора качества данных может быть недостаточно в некоторых задачах, относящихся к мониторингу:
  - исследование стабильности электроники,
  - исследование влияния пучкового фона на ECL.
- В таких случаях необходимо программное обеспечение для детального исследования низкоуровневых данных с калориметра.

- Разработан пакет программ для детальной диагностики электроники ECL по сохранённым данным, использовавшийся в большом числе задач:
  - Тестирование новых прошивок ECLCollector и ShaperDSP.
  - Анализ долговременной стабильности электроники.
  - Исследование влияния фона инжекции на ECL.
  - Совместный анализ данных ECL и логики триггера.
- Обработка может быть легко разбита на произвольное число независимых задач и запущена на вычислительном кластере.
- Данные можно комбинировать с информацией из БД DAQ, БД калибровок и информацией с монитора светимости.

Заключение

Разработано ПО для системы сбора данных электромагнитного калориметра.

- Подготовлены программные модули для синхронизации и управления конфигурацией электроники сбора данных в двух БД эксперимента Belle II.
  - Создан алгоритм упаковки, улучшающий сжатие конфигурационных данных в ~3 раза.
- Разработана система инициализации электроники калориметра.
  - Низкоуровневые модули объединены в фреймворк на языке C++.
  - Высокоуровневые модули на языке Python запускают инициализацию асинхронно.
  - Система инициализации регулярно используются дежурными экспертами для диагностики и исправления проблем с калориметром.
- Разработанное ПО интегрировано в систему медленного контроля.
  - Реализована библиотека PyNSM2, позволяющая взаимодействовать с системой медленного контроля с использованием языка Python.
  - Разработана система сборки, автоматически адаптирующая PyNSM2 к изменениям в ПО медленного контроля.
- Для дежурных разработан веб-сервер, объединяющий все необходимые утилиты.
- Подготовлено ПО для считывания данных с монитора светимости.
- Оптимизирован код обработки данных с калориметра.
- Разработан пакет программ для детальной диагностики электроники калориметра по сохранённым данным.

Публикации по теме работы

Основные результаты диссертации опубликованы в пяти публикациях, из них пять в изданиях, рекомендуемых ВАК при Минобрнауки России:

1. CsI(Tl) pulse shape discrimination with the Belle II electromagnetic calorimeter as a novel method to improve particle identification at electron–positron colliders / S. Longo, ..., M. Remnev, ... // Nuclear Instruments and Methods in Physics Research Section A – 2020. – Vol. 982, nr. 0168–9002. – P. 164562. – URL: https://doi.org/10.1016/j.nima.2020.164562
2. Data acquisition system for Belle II electromagnetic calorimeter / A. Kuzmin, M. Remnev, D. Matvienko, ... // Journal of Instrumentation. – 2020. – Vol. 15, – nr 7. – P. C07020. – URL: https://doi.org/10.1088/1748-0221/15/07/C07020
3. Data acquisition system for the calorimeter of the Belle II detector / A. Kuzmin, M. Remnev, D. Matvienko, Y. Usov [et al.]. — Текст : электронный // Physics of Atomic Nuclei. — 2021. — Vol. 84, nr. 1. — P. 42–44. — URL: https://doi.org/10.1134/S1063778821010257
4. Trigger slow control system of the Belle II experiment / C.-H. Kim, ... M. Remnev, ... // Nuclear Instruments and Methods in Physics Research Section A – 2021. – Vol. 1014, nr. 0168-9002. – P. 165748. – URL: https://doi.org/10.1016/j.nima.2021.165748
5. Development of data acquisition system for Belle II electromagnetic calorimeter / V. Aulchenko, A. Bobrov, B.G. Cheon, A. Kuzmin [et al.]. — Текст : электронный // Nuclear Instruments and Methods in Physics Research Section A – 2022. – Vol. 1030 – P. 166468. – URL: https://doi.org/10.1016/j.nima.2022.166468

Резервные слайды

Специальность 1.3.2

Шифр научной специальности: 1.3.2. Приборы и методы экспериментальной физики

1. Изучение физических явлений и процессов, которые могут быть использованы для создания принципиально новых приборов и методов экспериментальной физики.
2. Разработка новых принципов и методов измерений физических величин, основанных на современных достижениях в различных областях физики и позволяющих существенно увеличить точность, чувствительность и быстродействие измерений.
3. Разработка и создание научной аппаратуры и приборов для экспериментальных исследований в различных областях физики.
4. Развитие квантовой теории измерений.
5. Исследование фундаментальных ограничений на точность измерений.
6. Разработка и создание экспериментальных установок для проведения экспериментальных исследований в различных областях физики.
7. Разработка и создание новых приборов и аппаратурных комплексов для исследований в области астрономии и астрофизики.
8. **Разработка и создание средств автоматизации физического эксперимента.**
9. Разработка методов математической обработки экспериментальных результатов.
10. Моделирование физических явлений и процессов.
11. Разработка и создание лечебно-диагностических методик и аппаратурных комплексов для биомедицинских исследований.

Схема электроники калориметра

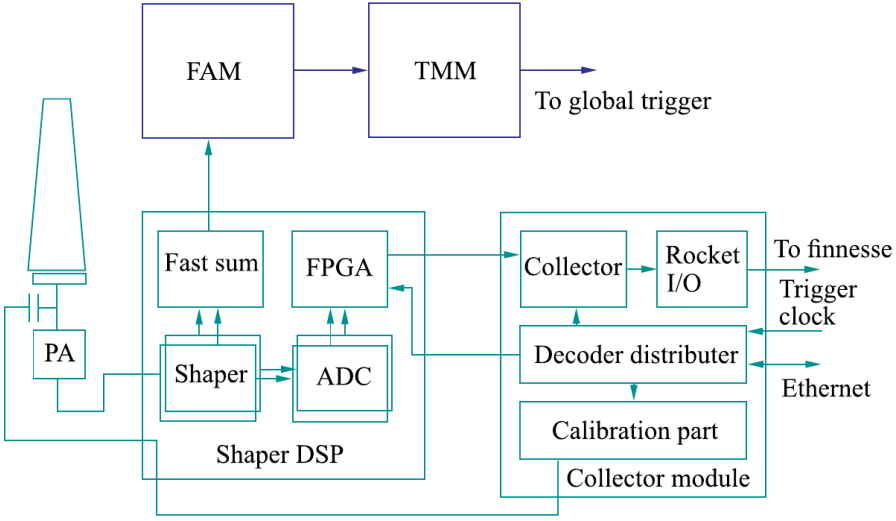


Рис. 3. Схема электроники калориметра

Детальное заключение

- Разработано ПО для системы сбора данных электромагнитного калориметра.
- Подготовлены программные модули для синхронизации и управления конфигурацией электроники сбора данных в двух БД эксперимента Belle II.
  - Конфигурация легко может корректироваться для разных типов заходов.
  - История конфигурации архивируется и может быть в дальнейшем использована для моделирования калориметра с заданными настройками.
  - Разработаны решения для задания более детальной тестовой конфигурации.
  - Создан алгоритм упаковки, специально подстроенный под использующуюся структуру конфигурационных данных. Его использование позволило увеличить эффективность сжатия в три раза, не теряя при этом в быстродействии.
- Разработан фреймворк, объединяющий все функции инициализации электроники ECL.
  - Архитектура фреймворка позволяет абстрагироваться от используемого протокола.
  - Разработаны инструменты автоматизированного тестирования.
- Подготовлены высокоуровневые утилиты инициализации
  - Поддерживается большое число дополнительных параметров работы.
  - Утилиты регулярно используются дежурными экспертами для диагностики и исправления проблем с калориметром, а также при тестировании новых версий прошивки.
  - Процедура инициализации может работать независимо от основной системы медленного контроля, развёртывая свою сеть взаимодействующих процессов.
- Приложения интегрированы с фреймворком Network Shared Memory 2.
  - Реализована библиотека pyNSM2 для взаимодействия с системой медленного контроля с использованием языка Python.
  - Подготовлено несколько системных демонов, управляющих электроникой калориметра.
  - Расширен графический интерфейс для запуска калибровок.
  - Создана отдельная база данных с информацией по набранным калибровочным заходам.

Детальное заключение

- Для экспертов по ECL разработан веб-сервер, объединяющий все необходимые утилиты для быстрого доступа.
  - Все основные процедуры диагностики и устранения проблем с электроникой калориметра были внесены в документацию дежурных.
  - Разработано несколько приложений для мониторинга качества данных с калориметра, реализована система автоматических оповещений.
- Подготовлено ПО для считывания данных с монитора светимости и передачи информации о светимости в систему медленного контроля.
- Разработаны и дополнены некоторые модули для обработки данных с калориметра.
  - Разработаны исходные версии модулей временной калибровки, оптимизирована распаковка данных и реализованы процедуры для первичной обработки данных в калибровке нелинейности.
  - Для детального исследования качества данных реализован набор управляющих скриптов Python, с отдельной библиотекой для структурированной записи выходных данных.

Упаковка коэффициентов для фитирования сигнала

Большинство массивов DSP коэффициентов F_i могут быть представлены в виде

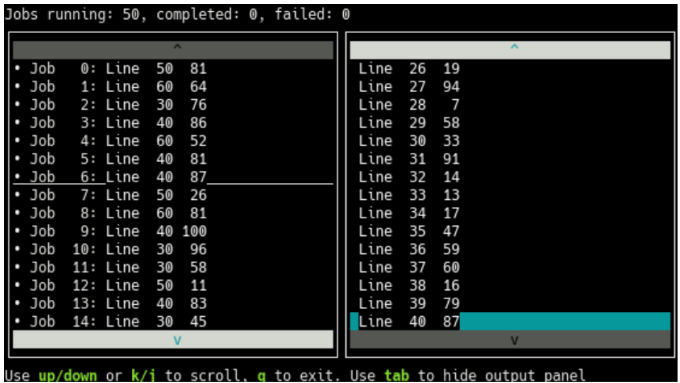
$$F_{16n+m} = F_m + f_{16n+m}, \qquad n = 0 \ldots 191,\ m = 0 \ldots 15 \tag{1}$$

где для почти каждого i выполняется условие $|f_i| < 16$. Это означает, что массивы 16-битных чисел из $16 \cdot 192$ элементов могут быть реорганизованы как массив 16-битных чисел F_m на 16 элементов и массив 5-битных чисел f_i на $16 \cdot 191$ элементов. В результате, за счёт исключительно упаковки коэффициентов достигается следующий уровень сжатия:

$$\frac{16 \cdot 16 \cdot 192}{16 \cdot 16 + 5 \cdot 16 \cdot 191} \approx 3.2$$

На практике, алгоритм упаковки достигает уровня сжатия 2.9, потому что некоторые массивы DSP коэффициентов не могут быть представление в виде (1). Тем не менее, этот алгоритм хорошо сочетается вместе со стандартными алгоритмами сжатия данных. Использование его совместно с LZMA позволяет достичь уровня сжатия 7.9

Клиент наблюдения за статусом инициализации электроники



- Mouse support
- ANSI color codes support
- Works on many different terminals

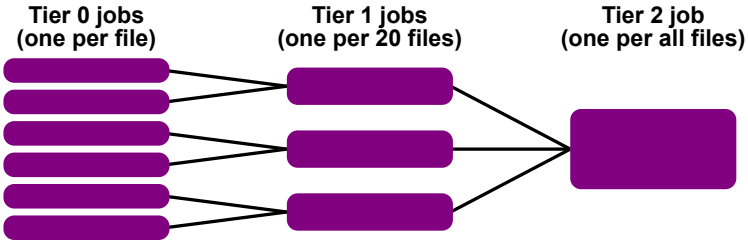


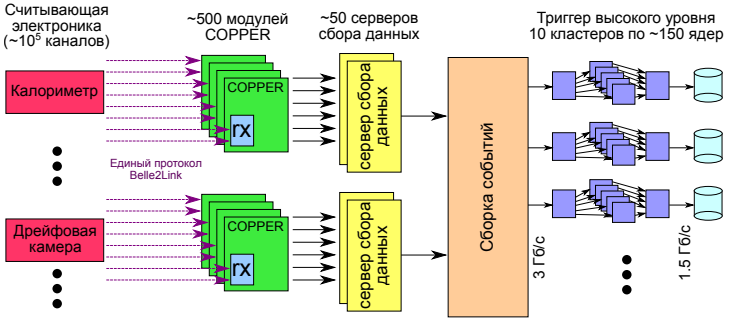
Рис.: Процесс калибровки, выполняющийся в несколько этапов.

Контактные данные

Ремнев Михаил Анатольевич
m.a.remnev@inp.nsk.su
mikhail.a.remnev@gmail.com

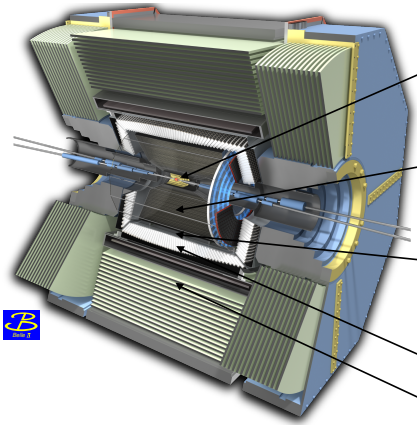
Система сбора данных Belle II

- Считывающая электроника индивидуальна для каждой подсистемы, но выходная информация должна передаваться в цифровом виде по оптическим каналам в заданном формате.



- Оцифрованные данные с подсистем считываются платами COPPER (Common Pipelined Platform for Electronics Readout).
- Промежуточные сервера выполняют первичную сборку событий и взаимодействуют с системой управления заходами.
- Собранное событие отправляется на триггер высокого уровня (HLT), где происходит дальнейшая обработка данных и отбор событий.

Детектор Belle II



Вершинный детектор (VXD)
- 2 внутренних слоя: пиксельный детектор (PXD)
- 4 внешних слоя: полосковый детектор (SVD)

Дрейфовая камера (CDC)
- He (50%), C_2H_6 (50%)

Идентификация частиц:
- Баррель: времяпролётный детектор (TOP)
- Торцы: черенковский детектор (ARICH)

ЭМ калориметр (ECL)
- Кристаллы CsI(Tl)

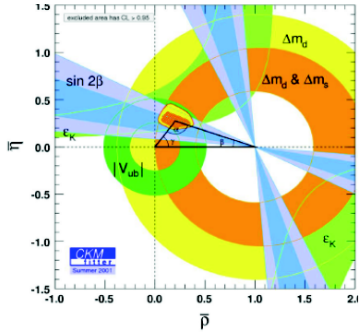
Детектор K_L и мюонов (KLM)

- Частота триггера на проектной светимости: 30 кГц.
- Поток данных со всех подсистем кроме PXD: 1.4 ГБ/с.
- Кроме систем детектора, непосредственно регистрирующих частицы, важными компонентами являются система триггера и система сбора данных (ССД).

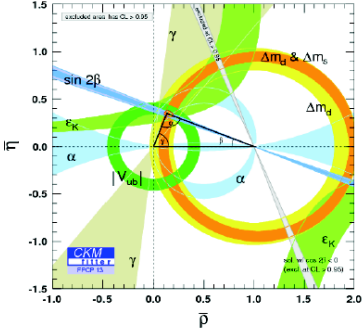
Основные результаты Belle

- Первое наблюдение нарушения CP в распадах B-мезонов.
- Измерение параметров CKM матрицы и углов треугольника унитарности.
- Наблюдение смешивания D-мезонов.
- Открытие новых адронов. Более 20 новых частиц было открыто на Belle (включая экзотические состояния Y(3862), четырёх кварковые состояния чармония и боттомония).
- Измерение редких мод распада B, D-мезонов и τ-лептона.
- Поиск новой физики за пределами стандартной модели.

EPS 2001



FPCP2013



Скорость счёта физических событий

- Проектная светимость $L = 8 \cdot 10^{35}\text{см}^{-2}\text{с}^{-1} = 800$ nb.
- Скорость счёта $e^+e^- \approx 24$ кГц.
- Скорость счёта $B\overline{B} \approx 840$ Гц.

- Текущая светимость $L = 3 \cdot 10^{34}\text{см}^{-2}\text{с}^{-1} = 30$ nb.
- Скорость счёта $e^+e^- \approx 0.9$ кГц.
- Скорость счёта $B\overline{B} \approx 30$ Гц.

$e^+e- \rightarrow$	Сечение (nb)
$b\overline{b}$	1.05
$c\overline{c}$	1.30
$s\overline{s}$	0.35
$u\overline{u}$	1.39
$d\overline{d}$	0.35
$\tau^+\tau^-$	0.94
$\mu^+\mu^-$	1.16
e^+e^-	∼29 (eff, Belle II)

Сечения при $\sqrt{(s)} = M(\Upsilon(4S))$.
(BaBar physics book, p. 75)

- Для подавления числа сохраняемых событий e^+e^- используется prescaling.
  (На данный момент 1/100 в триггере первого уровня)

HLT prescaling	
bhabha_calib	0.143
hadron_calib	0.179
cosmic_calib	0.179
mumu_calib	0.357
gamma_gamma_calib	0.643

Эффект Тушека, рассеяние на остаточном газе,
При высокой светимости – пучковое рассеяние (rad Bhahba рассеяние на малые углы создаёт фон).

Machine Design Parameters

parameters		KEKB		SuperKEKB		units
		LER	HER	LER	HER	
Beam energy	E_b	3.5	8	4	7.007	GeV
Half crossing angle	φ	11		41.5		mrad
# of Bunches	N	1584		2500		
Horizontal emittance	ε_x	18	24	3.2	4.3	nm
Emittance ratio	κ	0.88	0.66	0.27	0.31	%
Beta functions at IP	β_x^*/β_y^*	1200/5.9		32/0.27	25/0.30	mm
Beam currents	I_b	1.64	1.19	3.6	2.6	A
beam-beam param.	ξ_y	0.129	0.090	0.087	0.081	
Bunch Length	σ_z	6.0	6.0	6.0	5.0	mm
Horizontal Beam Size	σ_x^*	150	150	10	11	um
Vertical Beam Size	σ_y^*	0.94		0.048	0.063	um
Luminosity	**L**	**2.1×10^{34}**		**8×10^{35}**		**$cm^{-2}s^{-1}$**

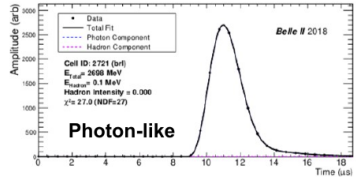
CsI(Tl) Pulse Shape Discrimination

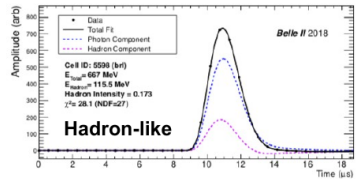
Reconstruction of neutral particles is an important capability of B-factories

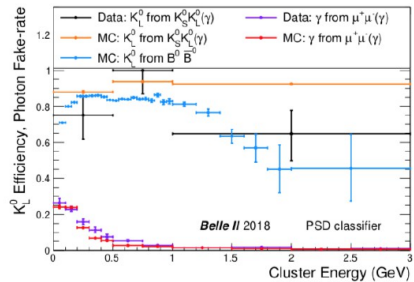
e.g. $B \rightarrow J/\psi K_L^0$ measurement of sin(2β)

- Calorimeter-based neutral hadron discrimination exploits difference in time structure of scintillation light from hadrons compared to EM showers

PhD thesis work of Savino Longo (UVic)
NIM A paper in preparation



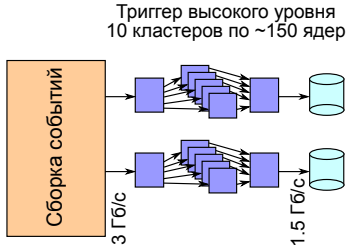


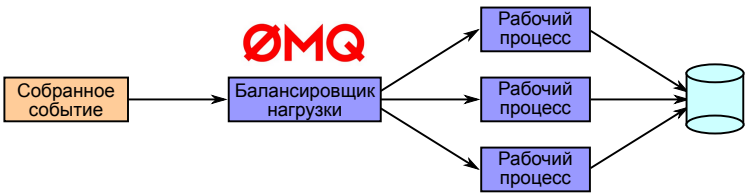


Fit ECL waveforms with templates for photon and hadron components to obtain “Hadron Intensity”

Обработка данных в триггере высокого уровня (HLT)

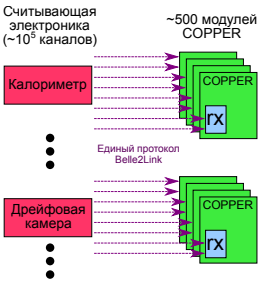
- Собранные события параллельно обрабатываются на серверах триггера высокого уровня (HLT).
- После обработки отбрасываются события, не прошедшие требования, остальные записываются на диск.
- Координация серверов HLT выполняется через Zero MQ, программную библиотеку для межпроцессного взаимодействия.

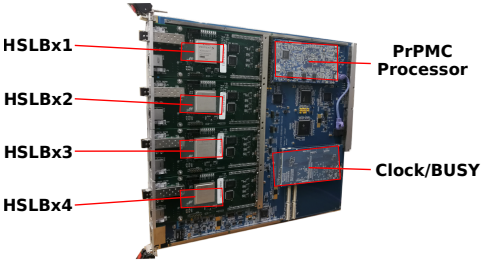




- Исходно для межпроцессного взаимодействия использовались примитивы UNIX, однако они работали менее стабильно и с примерно таким же быстродействием.
- Кроме того, Zero MQ имеет встроенный механизм для балансировки загруженности параллельных процессов.

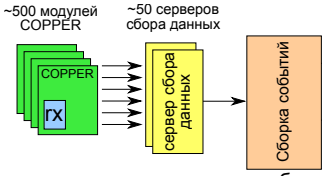
Модули COPPER (Common Pipelined Platform for Electronics Readout)





- Модуль COPPER включает в себя 1–4 платы для чтения данных (HSLB, High Speed Link Board) по протоколу Belle2Link.
- Если буфер близок к переполнению, плата может послать сигнал BUSY, приостанавливающий триггер.
- На плате установлен процессор, исполняется Scientific Linux 5.
  - Это позволяет напрямую интегрировать COPPER в сеть медленного контроля/управления заходами.

Медленный контроль



На всех COPPER и серверах ССД запущено программное обеспечение медленного контроля, выполняющего следующие задачи:

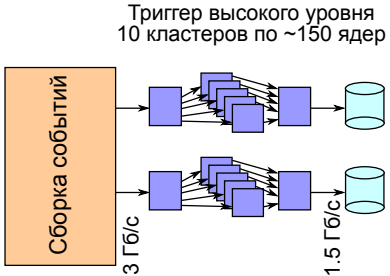
- Мониторирование температуры, влажности, ...
- Инициализация электроники.
- Запуск и остановка чтения данных.
- Проведение калибровок.
- Отображение статуса сбора данных.

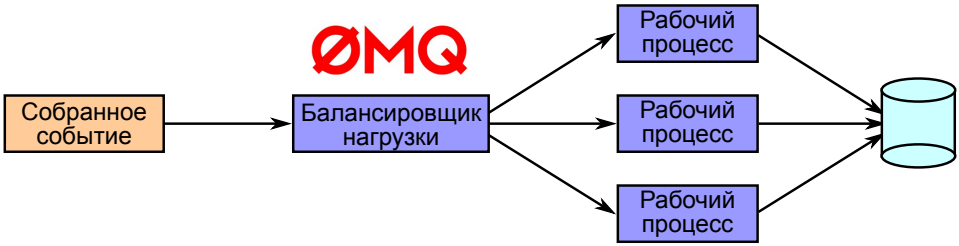
Для медленного контроля используются фреймворки EPICS и NSM2.

- Интерфейс управления заходами в Belle II.
- В отдельных окнах можно смотреть текущий поток данных от каждого COPPER.

Триггер высокого уровня (HLT)

- Собранные события параллельно обрабатываются на серверах триггера высокого уровня (HLT).
- После обработки отбрасываются события, не прошедшие требования, остальные записываются на диск.
- Координация серверов HLT выполняется через Zero MQ, программную библиотеку для межпроцессного взаимодействия.



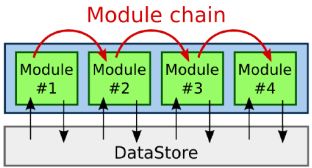


- Чтобы отфильтровывать данные фона, рабочие процессы HLT выполняют реконструкцию событий.

Реконструкция событий — переход от данных с каналов электроники к физическим

Триггер высокого уровня (HLT)

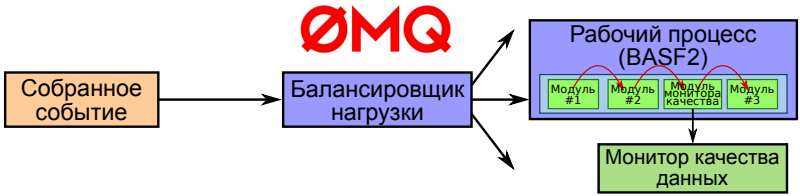
- Программы для реконструкции событий сложны в разработке и требуют больших вычислительных ресурсов.
- Для упрощения процесса разработки был создан фреймворк BASF2 (Belle Analysis Software Framework 2).



BASF2 — модульный фреймворк. Каждый этап реконструкции выделен в отдельный независимый модуль.
Гибкое задание цепочек модулей позволяет использовать BASF2 в различных сценариях:

- Реконструкция событий в HLT.
- Моделирование физических процессов.
- Анализ данных.

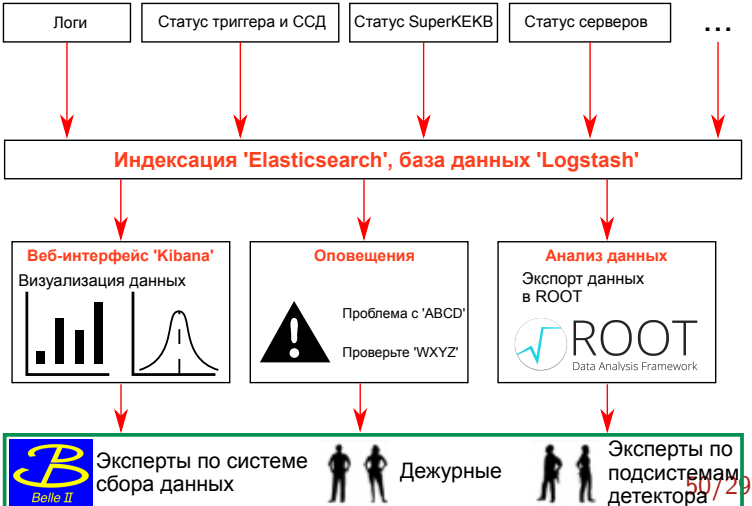
Такая схема работы позволяет HLT высылать в монитор качества данных информацию о событии между заданными шагами реконструкции.



Мониторирование статуса системы

Система сбора данных (ССД) имеет множество компонент, стабильность работы которых должна постоянно отслеживаться. С этой целью была разработана программная надстройка для мониторирования ССД.

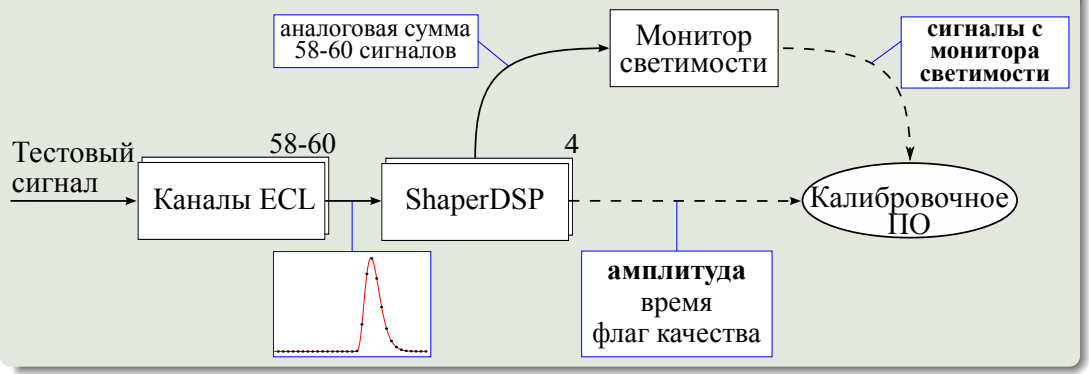
- Для мониторинга системы сбора данных используется стек программ ELK (Elasticsearch + Logstash + Kibana).
- ELK используется для визуализации данных медленного контроля, к примеру эффективности ССД.



Энергетическая калибровка монитора светимости

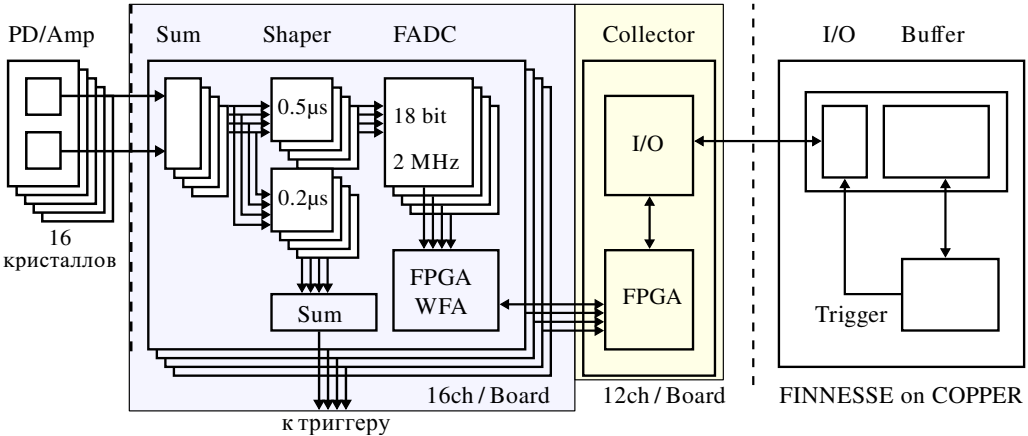
Energy calibration of LOM module is based on testpulse data.

Dataflow for single LOM channel during calibration



- Waveforms from luminosity monitor are fitted, resulting fit amplitude is compared against total amplitude of ECL channels.
- LOM calibration coefficients are then determined based on ECL energy calibration coefficients.

Электроника калориметра



Система сбора данных эксперимента Belle II

- В случае эксперимента Belle II поток данных меньше, чем для детекторов LHC, поэтому запись данных идёт по сигналу триггера первого уровня (L1).
- Тем не менее, ожидаемая высокая частота триггера требует использование дополнительного программного триггера высокого уровня (HLT).

Эксперимент	Число каналов	Частота триггера (L1 $\rightarrow$ HLT)	Поток данных (L1 $\rightarrow$ HLT)	Мёртвое время
СНД	2.4K	1 кГц	20 МБ/с	6%
КМД-3	15K	1 кГц	90 МБ/с	5%
KLOE	23K	10 кГц	50 МБ/с	2.2%
BESIII	30K	4 кГц	56 МБ/с	5%
Belle	150K	0.5 кГц	15 МБ/с	10%
CLEO III	400K	1 кГц	25 МБ/с	2%
Belle II	8.4M	30 кГц $\rightarrow$ 6 кГц	18 ГБ/с $\rightarrow$ 3 ГБ/с	3.4%
LHCb	1.1M	40 кГц $\rightarrow$ 200 Гц	6 ГБ/с $\rightarrow$ 50 МБ/с	2.5%
ALICE	15M	500 кГц $\rightarrow$ 2 кГц	15 ГБ/с $\rightarrow$ 2 ГБ/с	20%
CMS	78M	750 кГц $\rightarrow$ 7.5 кГц	55 ГБ/с $\rightarrow$ 1 ГБ/с	3%
ATLAS	100M	100 кГц $\rightarrow$ 1 кГц	160 ГБ/с $\rightarrow$ 1.6 ГБ/с	1%

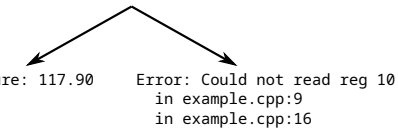
Научная новизна

- Детектор Belle II является уникальной установкой, соответственно требующей уникального программного обеспечения для управления системой сбора данных.
  - Новая организация баз данных потребовала разработки новой системы управлениями конфигурациями калориметра.
  - Разработана система сборки для модуля PyNSM2, которая автоматически обновляет соответствие структур между файлами исходного кода на языке C и на языке Python.
- Реализован алгоритм упаковки DSP-коэффициентов, достигающий более эффективных результатов по сравнению с универсальными алгоритмами сжатия.
- На языке C++ реализована Rust-подобная схема обработки ошибок, которая на 12% быстрее стандартного механизма обработки исключений C++ (при частоте ошибок 0.1%).
- Реализован набор простых в использовании макросов, позволяющих экспортировать функции C++ для использования в легковесном интерпретируемом скриптовом языке.

Rust-подобная схема обработки ошибок

```cpp
Result<int> readRegister(int addr)
{
  // 0.1% chance for register reading to fail
  if (rand() % 1000 == 0)
    return ERROR("Could not read reg " + to_string(addr));
  else
    return rand() % 256;
}

Result<double> readTemperature()
{
  double temp = TRY(readRegister(10)) * 0.7 - 55;
  return temp;
}
```

```cpp
int main()
{
  Result<double> result = readTemperature();
  if (result.isOk())
    printf("Temperature: %lf\n", result.getData());
  else
    printf("Error: %s\n", result.getError().c_str());

  return 0;
}
```



Идея заключается в использовании класса Result, реализованного наподобие std::variant, который может содержать либо возвращаемые данные, либо информацию об ошибке (как правило std::string). Макрос TRY использует возможности компилятора GCC (statement expression), чтобы обрывать дальнейшую обработку выражения, если в функции чтения регистра возникла ошибка. Помимо улучшения в производительности без усложнения программы, эта система интегрирована с расширениями компилятора GNU Compiler Collection, позволяя ещё на стадии сборки проекта обнаруживать места, где не выполняется обработка ошибок, что в результате улучшает стабильность кода.

Интерпретатор конфигурационных скриптов

```
// Вывести первую строку указанного файла
FUNCTOR(head, string filename)
{
  ifstream f(filename);
  string line;
  getline(f, line);
  return line;
}
```

```
$ ./example_cui
> head "/etc/hosts"
127.0.0.1 localhost
>
```

Функция 'head' может быть вызвана из командной строки.

Рисунок 11 — Пример кода, определяющего функтор для функции head

Чтобы уже на ранней стадии разработки предоставлять API для доступа к основным функциям приложения, на языке C++ была разработана простая библиотека генерации именованных функторов на стадии компиляции. Любая функция C++ или метод объекта могут быть в одну строку экспортированы в API, существенно упрощая интеграцию с системой DAQ.

В особых случаях, при настройке электроники для тестовых заходов или при использовании новых версий прошивки, возникает необходимость задать дополнительные параметры, не учтённые в текущей схеме. Поэтому был раз- работан скриптовый язык, позволяющий детально определять дополнительные действия, выполняемые на стадии установки конфигурации.

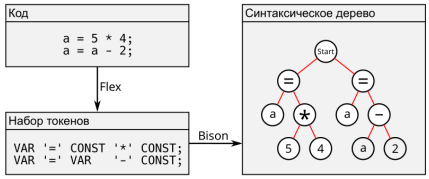


Рисунок 13 — Обработка скриптового языка