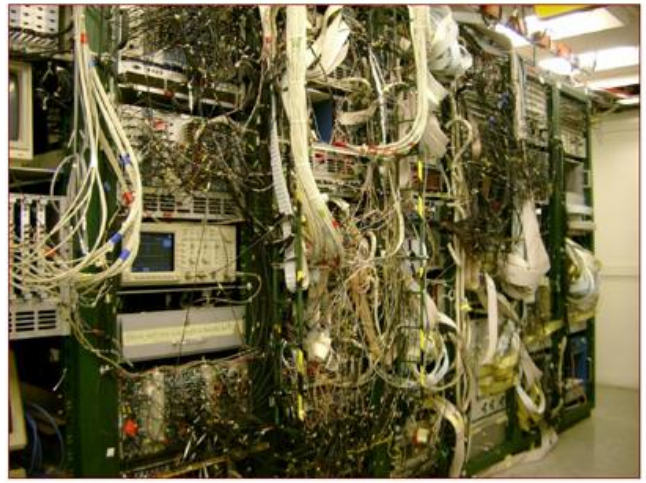
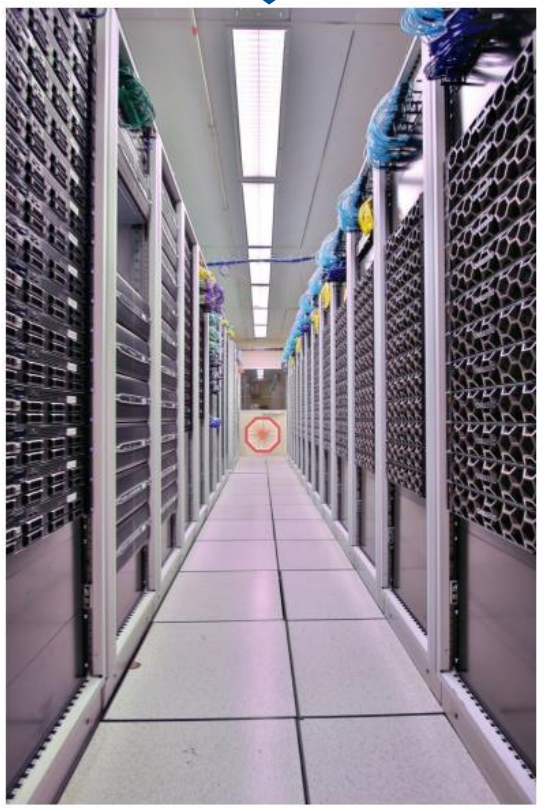
Тенденция к реализации ССД по схеме "бестриггерного чтения" в современных экспериментах ФЭЧ

Всё пишем, позже отбираем.

Введение

- В подготовке современных экспериментов по ФВЭ (NICA SPD, LHC LHCb, GSI – PANDA, CBM, NUSTAR, HL-LHC) реализация работы с данными в системе сбора данных (ССД) осуществляется по новой схеме:
  - стадию принятия решения о полезности события отодвигают на более поздний этап – на программный уровень (firmware-software).
  - таким образом, сначала всё оцифровывается, а затем отбирается
- Этому способствовало развитие IT индустрии и технологий высокопроизводительных FPGA





Задачи данного доклада

- Обзорный доклад:
  - Рассмотреть физические аспекты требующие реализации потокового чтения;
  - Познакомиться с концепцией бестриггерного чтения;
  - Посмотреть варианты реализации такой системы на примере установок – в частности, работа с данными на аппаратном и софтварном уровне;
  - Подробный разбор реализации на SPD (NICA);

Терминология. Triggerless/streaming/free-running DAQ

- Триггер (классическое толкование в ФЭЧ с точки зрения электроники) – первичная электроника (Да/Нет), амплитудные дискриминаторы

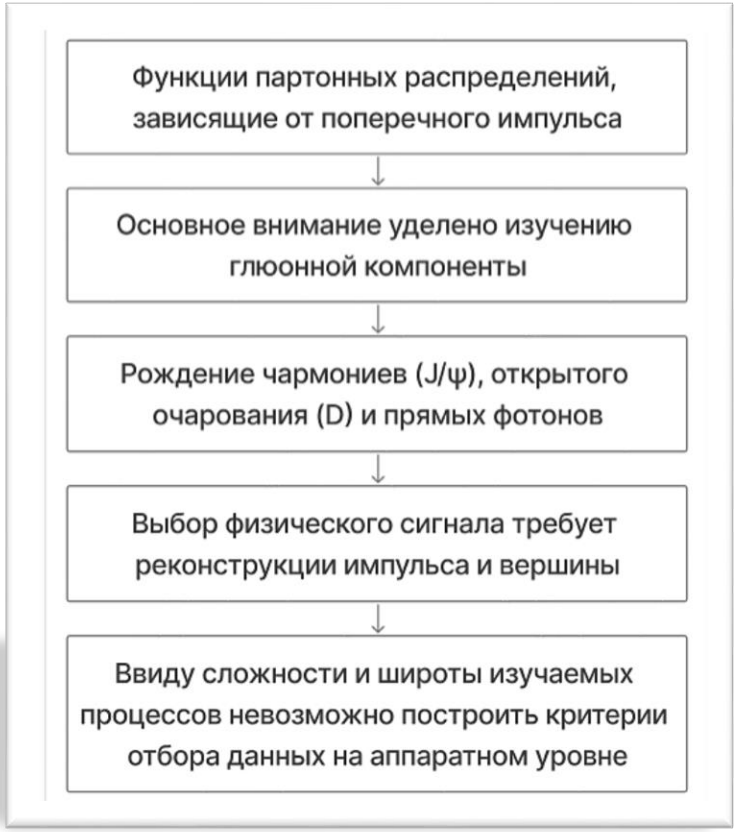
- Триггер (с точки зрения физики в ФЭЧ) – набор критериев по которым отбирается событие

- Triggerless DAQ doesn't really exist
- It just means we don't trigger on the Front-end
- We need a **good timing** scheme to in order to match data from different sources
- It also means we do **more selection in firmware/software**
- Evidently, a more complete event picture gives a better selection

Мотивация

- **Cost-efficiency**: 1 USD spent on an ASIC trigger board will be “active” only during the operation of the trigger, 1 USD spent on a CPU or GPU can be “active” round the clock
- Flexibility: new hard and software can be integrated without impacting the detector or operations
- Operations: compute for filtering does not need to be “on-prem”, *not operated by experiment, not even “owned”*

- Непрерывное считывание => непрерывная регистрация данных.
- Близость фоновых и сигнальных процессов, а также невозможность отбора событий на аппаратном уровне без предварительной реконструкции импульса частиц и положения вершины детектора, делают применение классической триггерной схемы нереализуемым.
- В 2009–2011гг. было предложено использование бестриггерной системы сбора данных для экспериментов PANDA и CBM на строящемся ускорительном комплексе FAIR. В 2013–2014 гг. бестриггерная система сбора данных была предложена для экспериментов LHCb и ALICE на LHC. Впервые данный метод был реализован в 2021 г. на экспериментах LHCb и ALICE в рамках модернизации ускорителя LHC для выполнения программы Run 3.



SPD

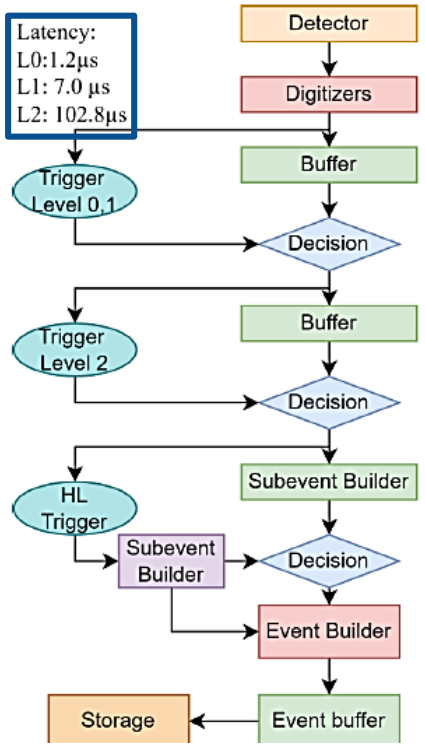
Мотивация

- Пример сравнительной схемы в SPD.
  - Минимальная задержка в передачи данных

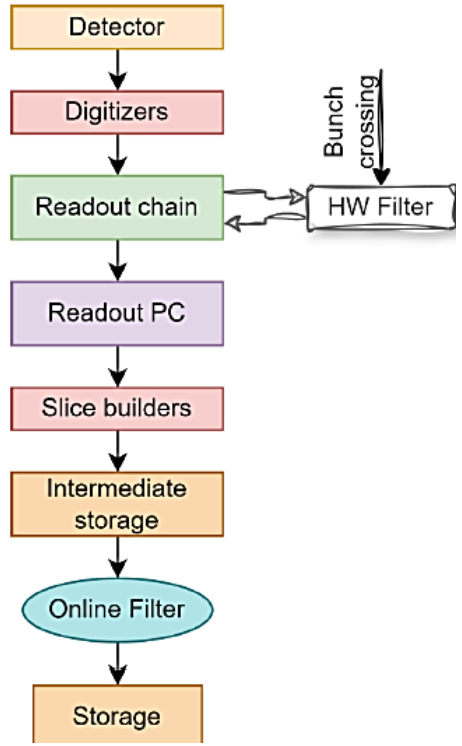
  со считывающих компьютеров
- В эксперименте SPD отказались от идеи аппаратного триггера. В нём данные с детектора считываются непрерывно. Это связано с тем, что в данном эксперименте решение о значимости «события» невозможно принять на аппаратном уровне, так как оно зависит от измерения импульса частиц и положения вершины взаимодействия.
- В контексте обработки данных – иметь возможность доступа к ним в распределённой сети (как к «сырым» данным, так и к обработанным).

Free-running (Triggerless) DAQ

Триггерная(классическая) DAQ



Triggerless DAQ



Мотивация

LHCb : From Nico-san's slides

https://indico.cern.ch/event/1037447/

Triggerless read-out: Why?

- No trigger is 100% efficient, hardware triggers always have to compromise
- A trigger-less front-end is much *less complex* and *more robust*
  - No buffering, no selection logic,
- Selection in "software" / using compute infrastructure from the data-centre world has a lot of advantages:
  - Scalability
  - Cost: costs in electronics are driven (down) by scale
  - **Cost-efficiency**: 1 USD spent on an ASIC trigger board will be "active" only during the operation of the trigger, 1 USD spent on a CPU or GPU can be "active" round the clock
  - Flexibility: new hard and software can be integrated without impacting the detector or operations
  - Operations: compute for filtering does not need to be "on-prem", *not operated by experiment, not even "owned"*

N. Neufeld CERN - Triggerless readout 22

Triggerless read-out: Why not?

- More data from the front-end
  - 0-suppression / compression / clever encoding required
  - more links → more power, more cost, more material
- Cultural shift:
  - from electronics to software engineers
  - from hardware projects to software commitments: What does your funding agency say?

!IGGER DAQ WORKSHOP : DEC. 2, 2022 21

Belle II case …

https://indico.cern.ch/event/1037447/

Triggerless read-out: Why?

- No trigger is 100% efficient, hardware triggers always have to compro
- A trigger-less front-end is much *less complex* and *more robust*
  - No buffering, no selection logic,
- Selection in "software" / using compute infrastructure from the data-centre has a lot of advantages:
  - Scalability
  - Cost: costs in electronics are driven (down) by scale
  - **Cost-efficiency**: 1 USD spent on an ASIC trigger board will be "active during the operation of the trigger, 1 USD spent on a CPU or GPU ca "active" round the clock
  - Flexibility: new hard and software can be integrated without impacting detector or operations
  - Operations: compute for filtering does not need to be "on-prem", *not operated by experiment, not even "owned"*

BB pair : 100% efficiency
This part (physics motivation) should be more studied.
(low-multiplicity event ?
Displaced vertex ₹?

This part is true even if hardware trigger is still available ?

N. Neufeld CERN - Triggerless readout 22

Triggerless read-out: Why not?

- More data from the front-end
  - 0-suppression / compression / clever encoding required
  - more links → more power, more cost, more material
- Cultural shift:
  - from electronics to software engineers
  - from hardware projects to software commitments: What does your funding agency say?

Large upgrades of FEE will be necessary.

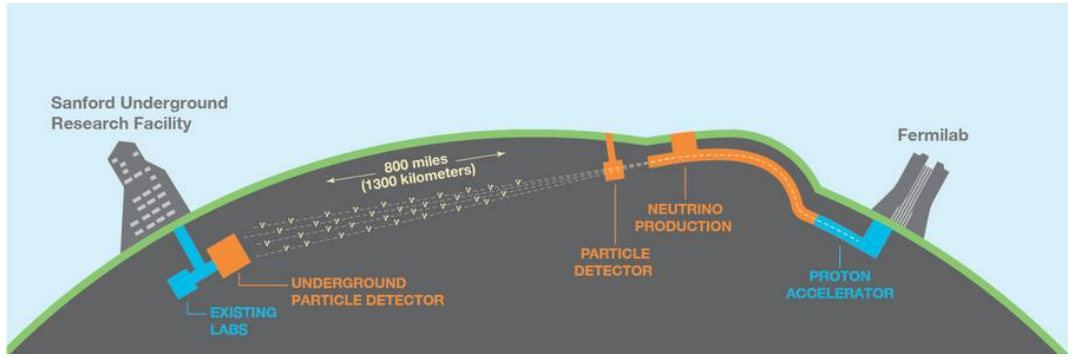
More HLT computing resource, network bandwidth,
New readout board ?

!IGGER DAQ WORKSHOP : DEC. 2, 2022 23

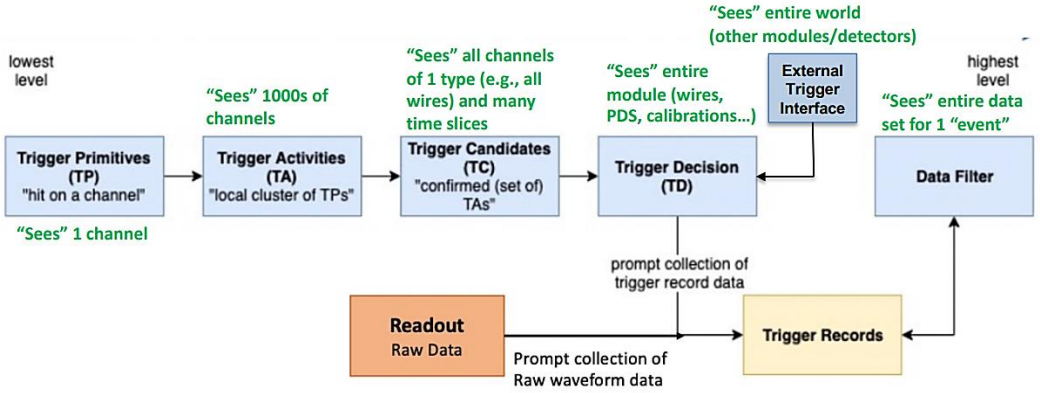
MINOS детектор в Fermilab

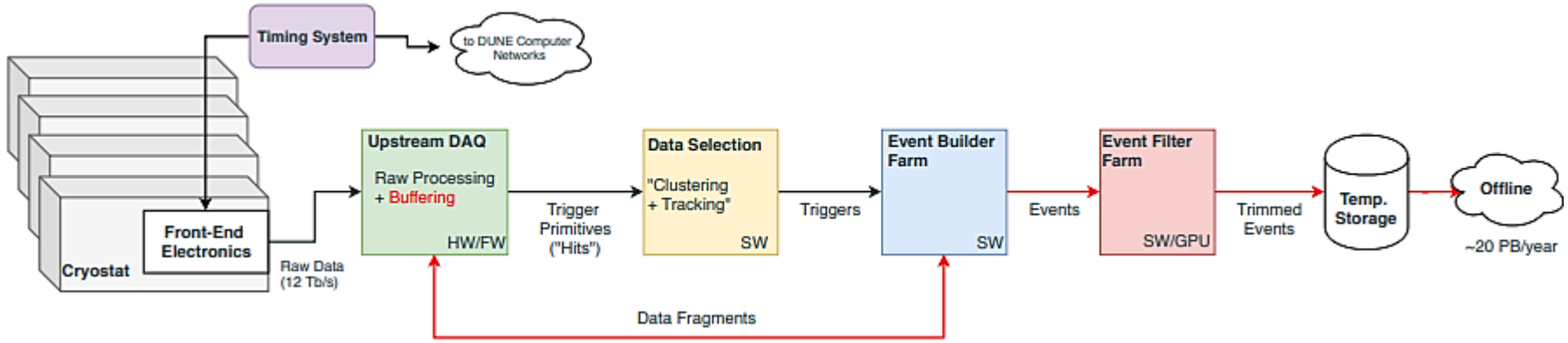
- Первые идеи использования потоковой ССД появились на экспериментах с нейтринными детекторами.
- Одна из первых бестриггерных ССД была применена в 2004 году для детектора Near Detector в эксперименте MINOS в Fermilab по изучению осцилляций нейтрино.
- Отсутствие аппаратного триггера, привязка к сигналу ускорителя fast spill, когда пучок направляется на графитовую мишень.
- Интервал 10 мкс для непрерывной оцифровки электроникой всех событий с частотой 53 МГц.
- Поток данных оценивался в 10 Мбит/с.
- 10к каналов электроники.

DUNE



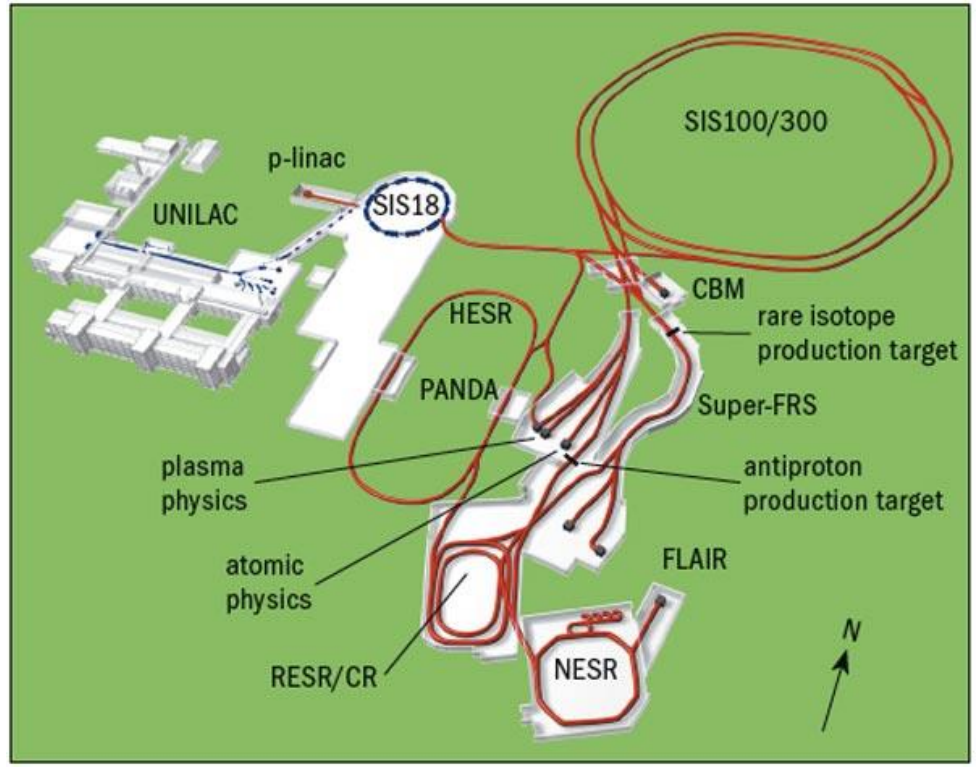
- 1.4 ÷1.5 ТБ/с – непрерывный несжатый поток данных
- 2 МГц частота дискретизации АЦП (12 бит разрешение)
- 100 секунд постоянный кольцевой буфер в оперативно памяти DRAM и на NVMe SSD дисках
- 5,4 мс: Стандартное временное окно, вырезаемое из буфера для записи одного обычного триггерного события. Оно соответствует максимальному времени дрейфа электронов в жидком аргоне
- На весь эксперимент объём данных в год ограничен
- 20 ÷30 ПБ/год

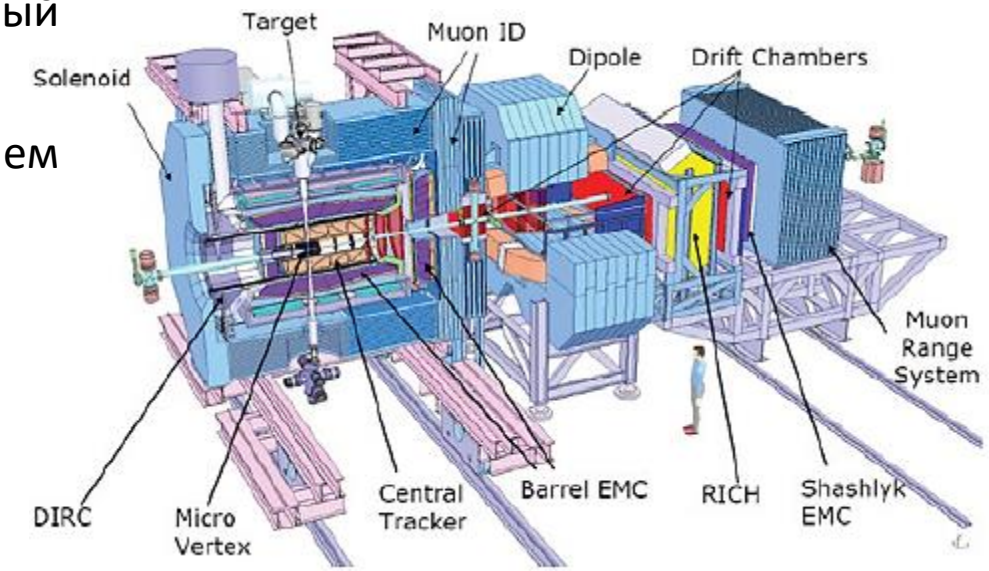




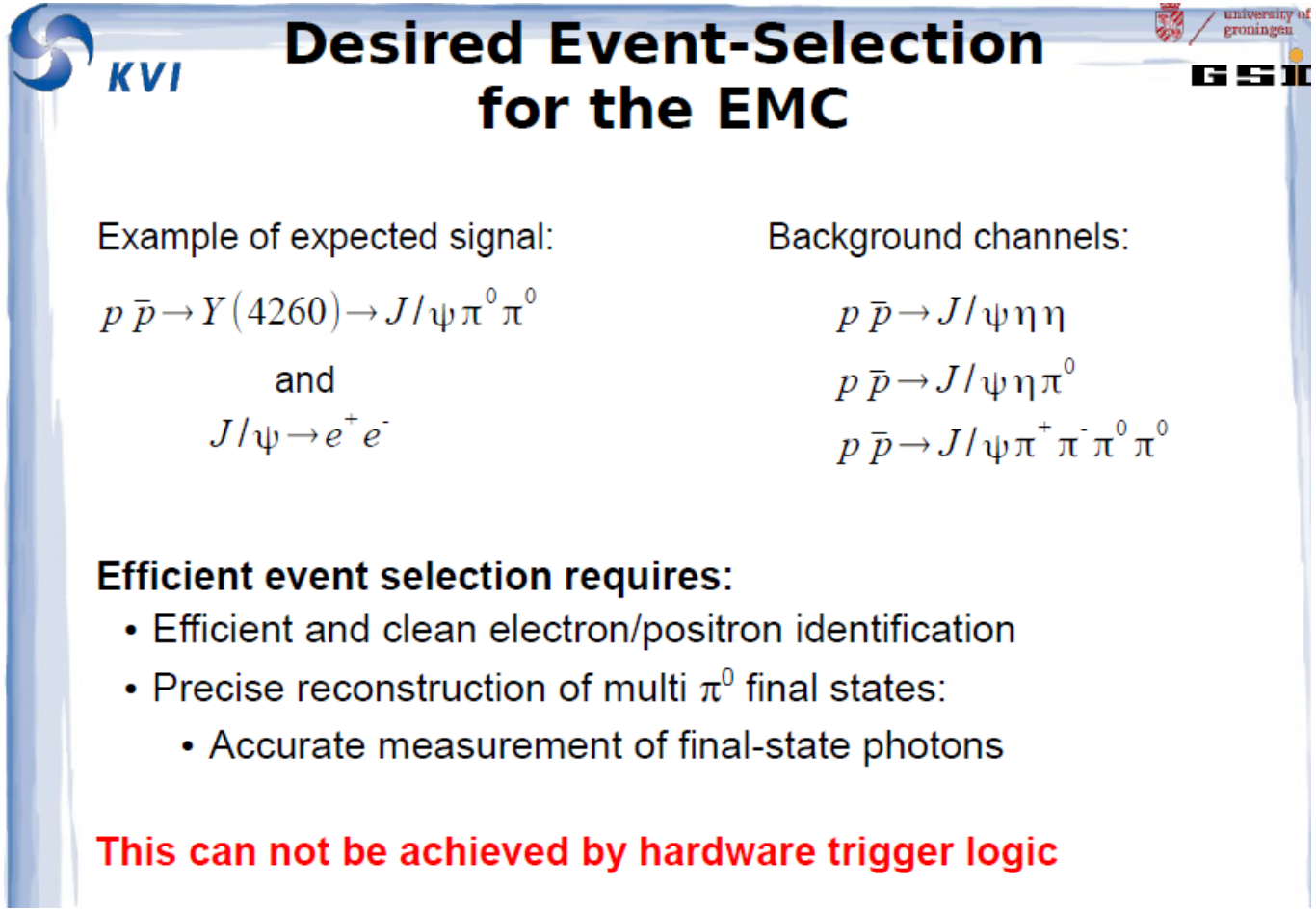
PANDA

- Именно на эксперименте PANDA предложили концепцию системы без общего триггера для коллайдера.
- Была детально проработана концепция фронт-энд электроники.
- Частота взаимодействий 20 МГц, частота срабатывания канала электроники 500 кГц.
- Доказали, что при частоте взаимодействия 20 МГц аппаратный триггер не может отбирать события.
- Концепция нарезки данных в FPGA+DDR с присвоением временных меток.
- Суммарный поток сырых данных – 200 Гбит/с – 1 Тбит/с.



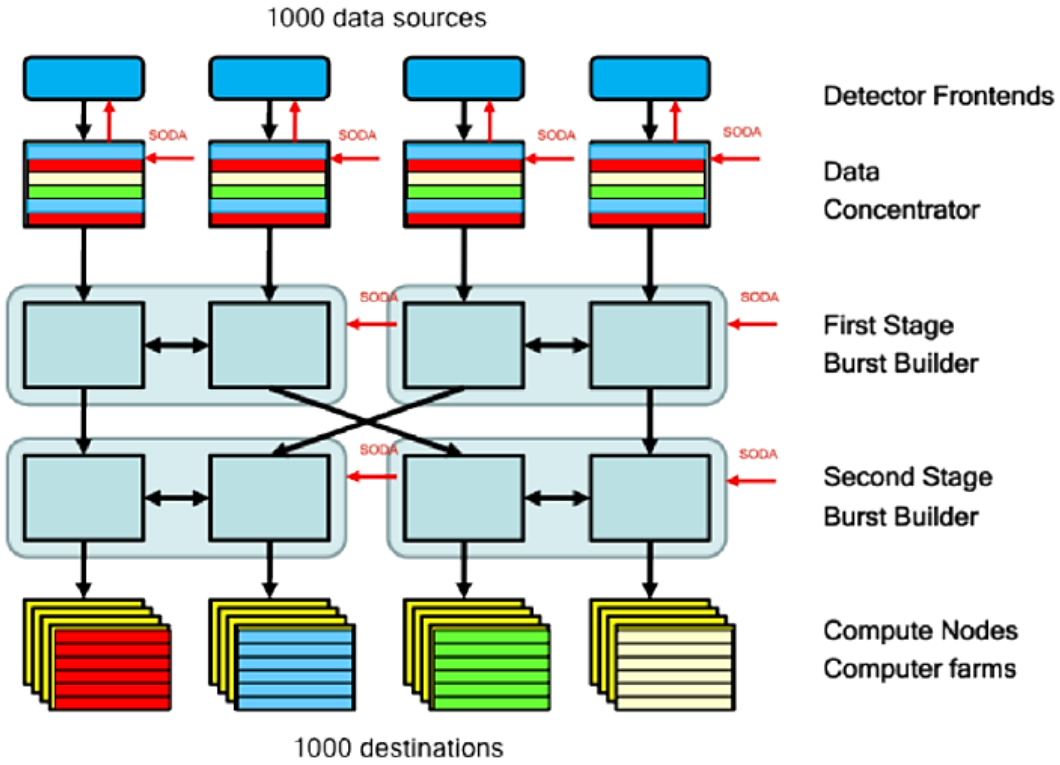


Калориметр PANDA. Физическая мотивация

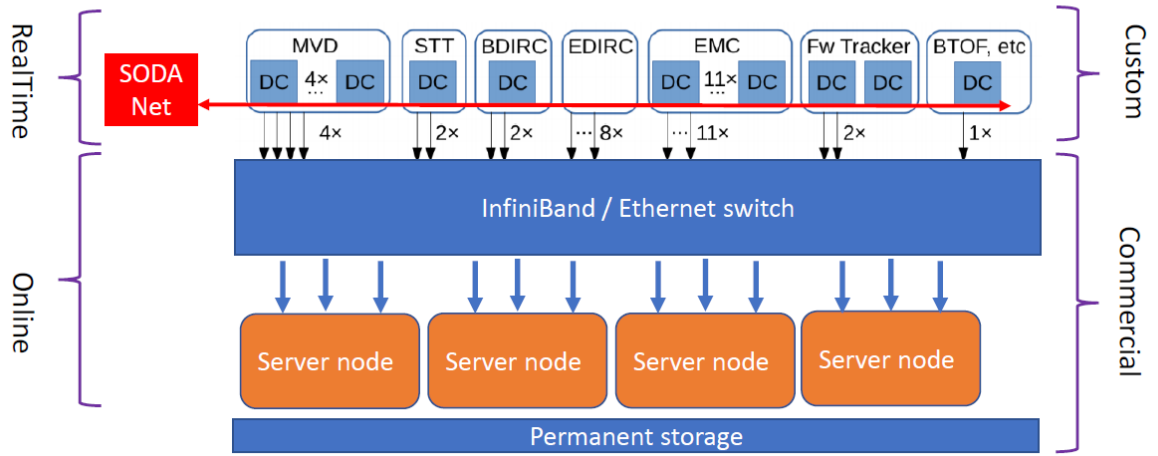


ССД PANDA

- SODA (Synchronisation of Data Acquisition Network)– система синхронизации



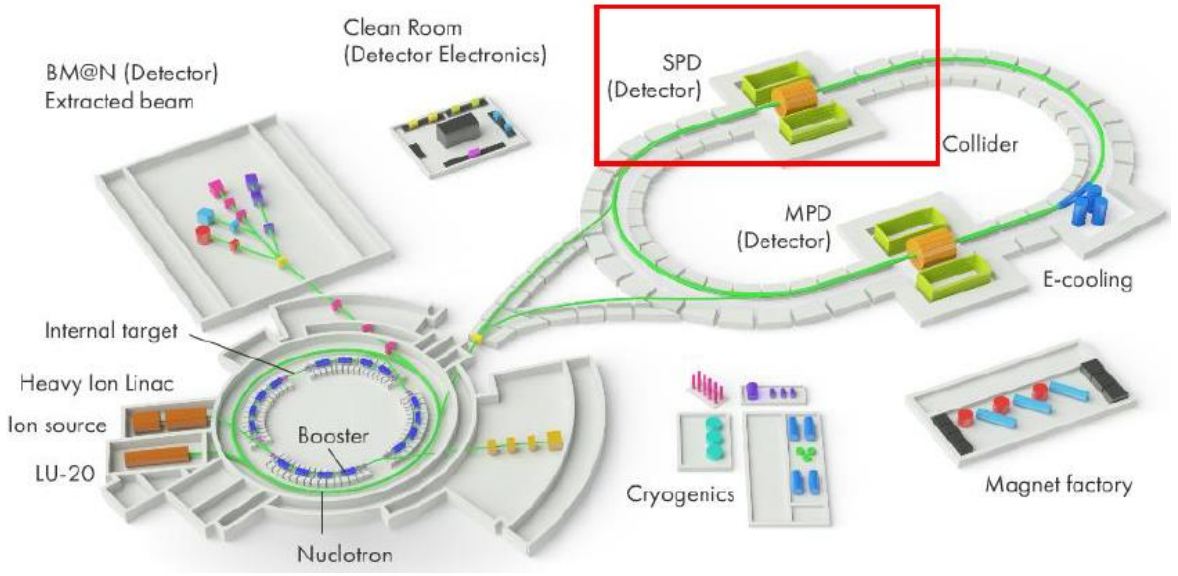
Proposed DAQ scheme



GPU Alveo и QUADRO

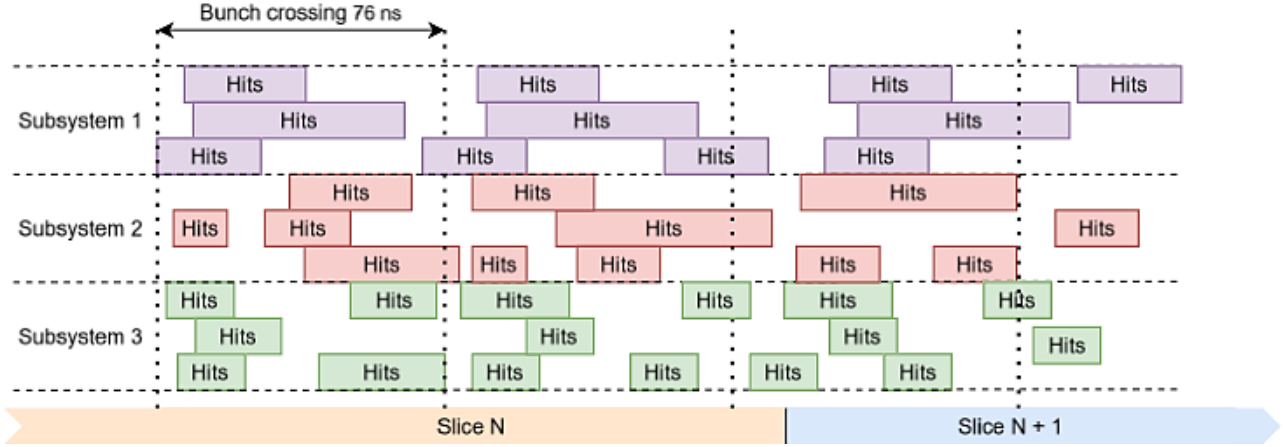
Что на SPD

- Частота столкновений пучков 3-4 МГц (на первом этапе 100 кГц).
- Период пересечения сгустков каждые 76 нс.
- Обширная физическая программа SPD исключает возможность отбора событий на аппаратном уровне триггера.
- Поток «сырых» данных будет составлять примерно 20 Гб/с с учётом упрощённого моделирования и некоторого запаса надёжности (для максимальной светимости L=10^{32}см$^{-2}$с$^{-1}$, $\sqrt{s} = 27$ ГэВ). На первом этапе поток «сырых» данных составит 1 Гб/с.



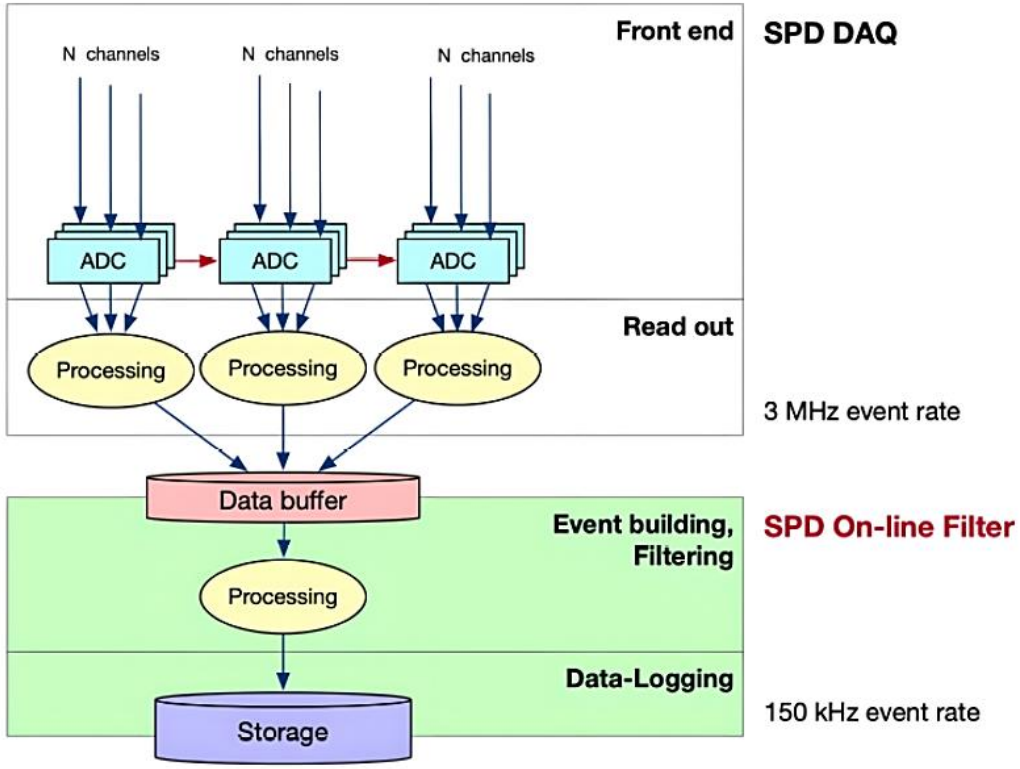
Следствие такого решения

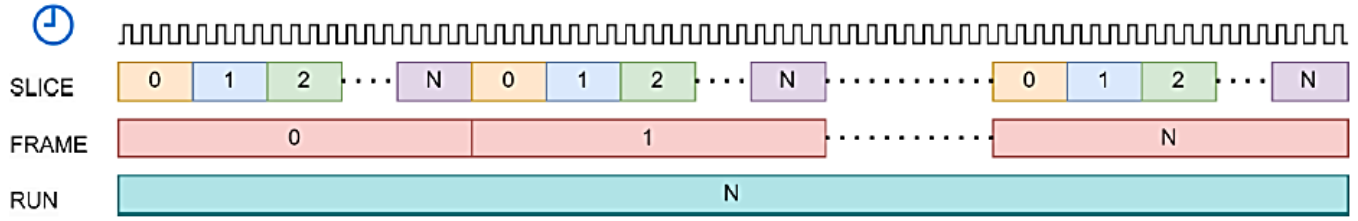
- Большой объём данных: в год объём предварительно обработанных данных достигнет 10 Пб.
- Для последующей оффлайн обработки и хранения будет использоваться распределённая вычислительная система.
- Комплексная сшивка и обработка событий: от DAQ приходит не массив «сырых» данных, а набор блоков данных от периферийной электроники детекторов. Сразу после DAQ – расшифровка событий путём частичной реконструкции, программный триггер, который по сути будет являться фильтром событий (высокопроизводительная система для обработки больших объёмов данных).



Реализация Triggerless DAQ

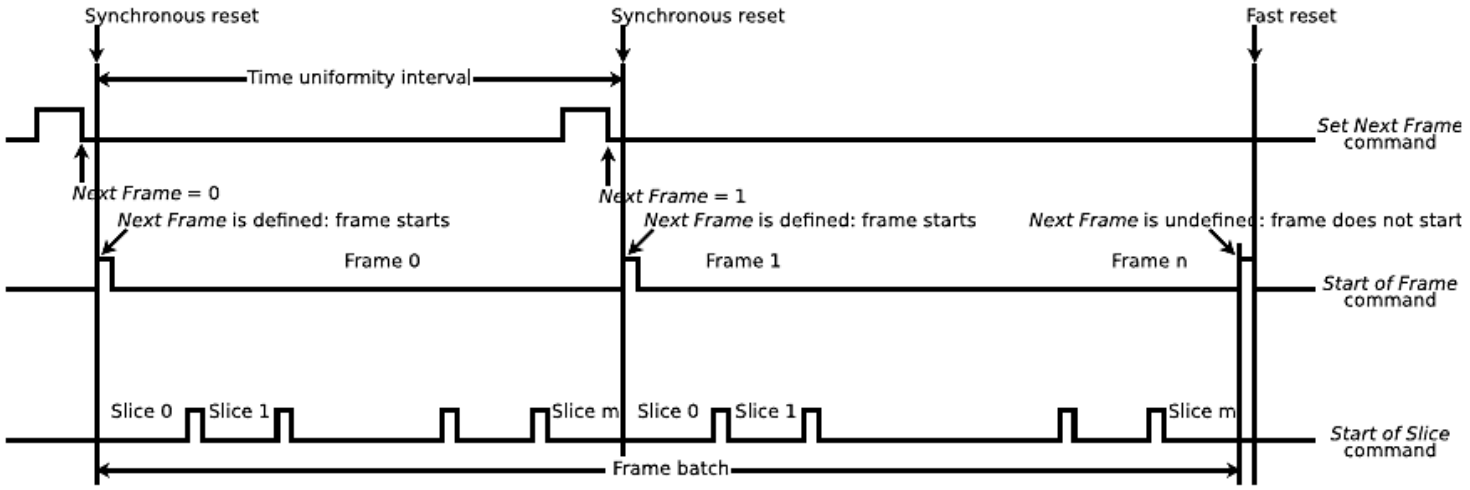
- Такая концепция предполагает минимальную задержку в передачи данных до считывающих компьютеров.
- Части слайсов с каждой подсистемы объединяются в один блок данных. Онлайн фильтр занимается отбором событий для сохранения в постоянное хранилище.
- Slice: 10 ÷ 100 мксек
- Frame: 0.1 ÷ 10 с
- Run: 8 часов





Команды ССД

- Асинхронные команды:
  - Disarm: активация линии сигнала сброса от L1 к FEE
  - Arm: деактивация линии сигнала сброса, позволяя FEE обрабатывать синхронные команды



- Синхронные команды:
  - Set next frame – подготовка системы к следующему кадру.
  - Start of frame – команда завершает текущий кард и инициирует новый.
  - Start of slice – команда завершает текущий фрагмент и запускает новый в том же кадре.

Чтение

L1 концентратор

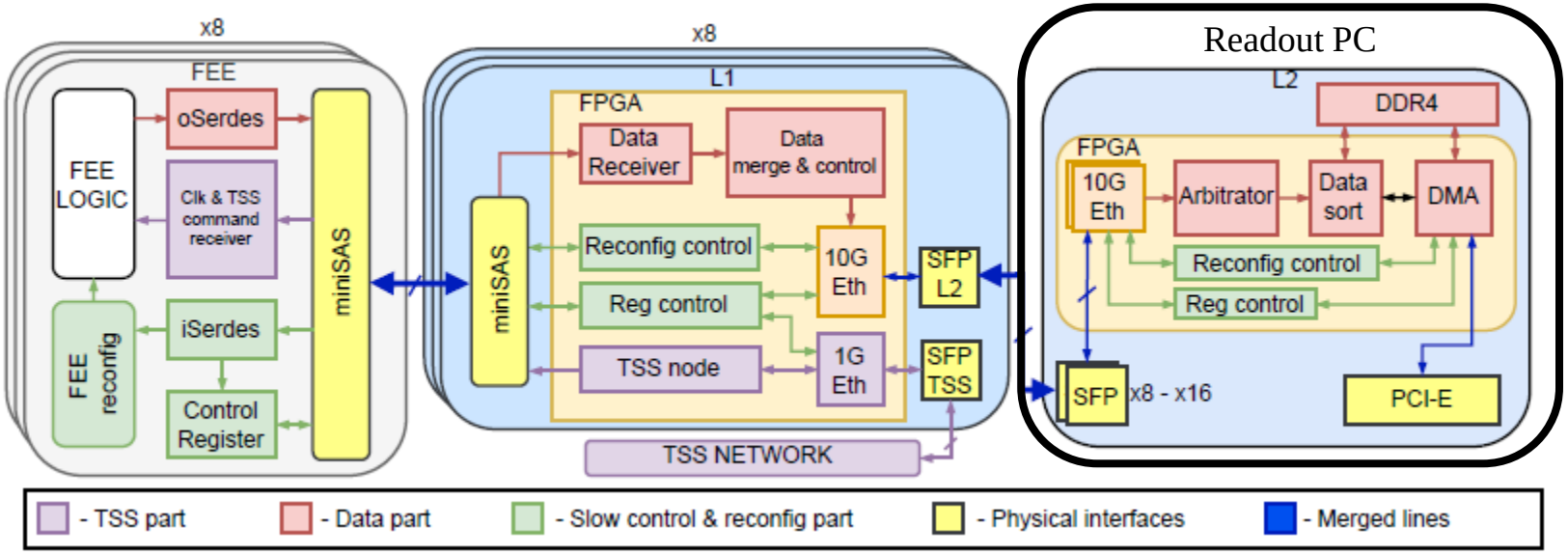
Аппаратная платформа:

- Целевая FPGA Cyclone 10GX (105YF780E6G)
- Опциональная FPGA Pangomicro Titan 2
- 8x разъемов MiniSAS (8 дифференциальных пар и 8 однополярных линий на 1 разъем)
- SFP+ 10Gb приемопередатчик для подключения к L2 концентратору
- SFP+ Приемопередатчик для организации работы TSS
- Пропускная способность 1 Gb/s между FEB и L1

L2 концентратор

Аппаратная платформа:

- Отладочная плата ALINX Z19-P
- FPGA: AMD Zynq UltraScale+ MPSoC XCZU19EG
- ARM Cortex-A53 x4, Cortex-R5 x2
- До 16 SFP приемопередатчиков для подключения к L1
- PCIe 3.0 x8 с пропускной способностью до 15 GB/s (6.5 GB/s достигнуто на данный момент)
- Внешняя DDR4 память 4+4 GB



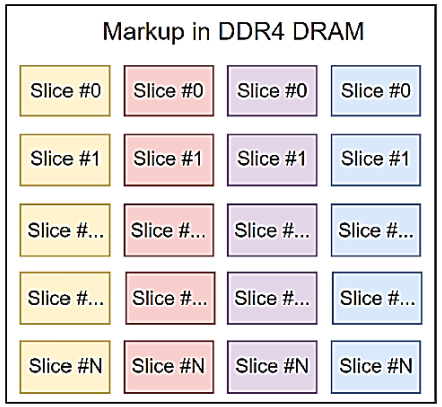
Задача концентратора L1:

- Сбор данных с FEE, контроль целостности данных, контроль потока данных
- Приём и распределение глобального тактового сигнала и команд TSS контроллера
- Переконфигурирование FEE и реконфигурация FEE

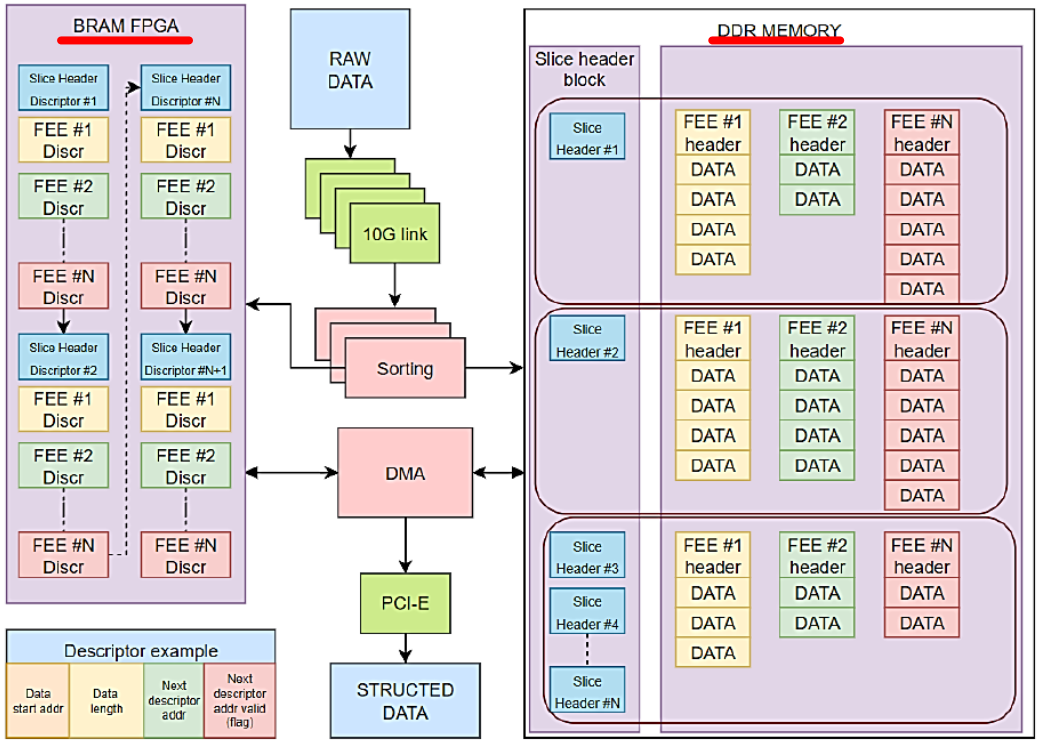
Задача концентратора L2:

- Приём данных с концентраторов уровня L1
- Сортировка данных по слайсам и буферизация во внешней DDR памяти
- Передача сортированных данных в Readout PC
- Организация управления внутренними регистрами L1 и регистрами FEE через L1
- Переконфигурирование FEE через L1

Работа с данными (кластеризация)

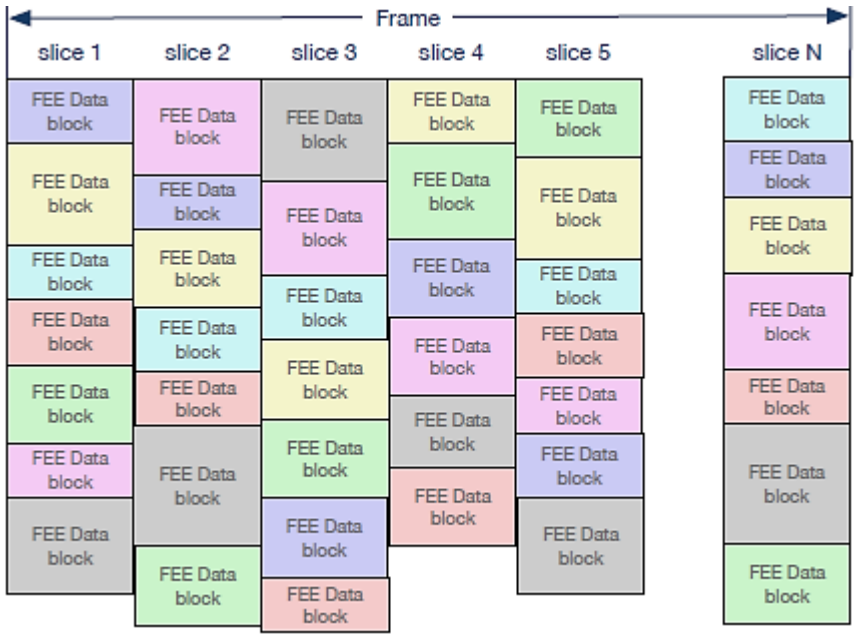


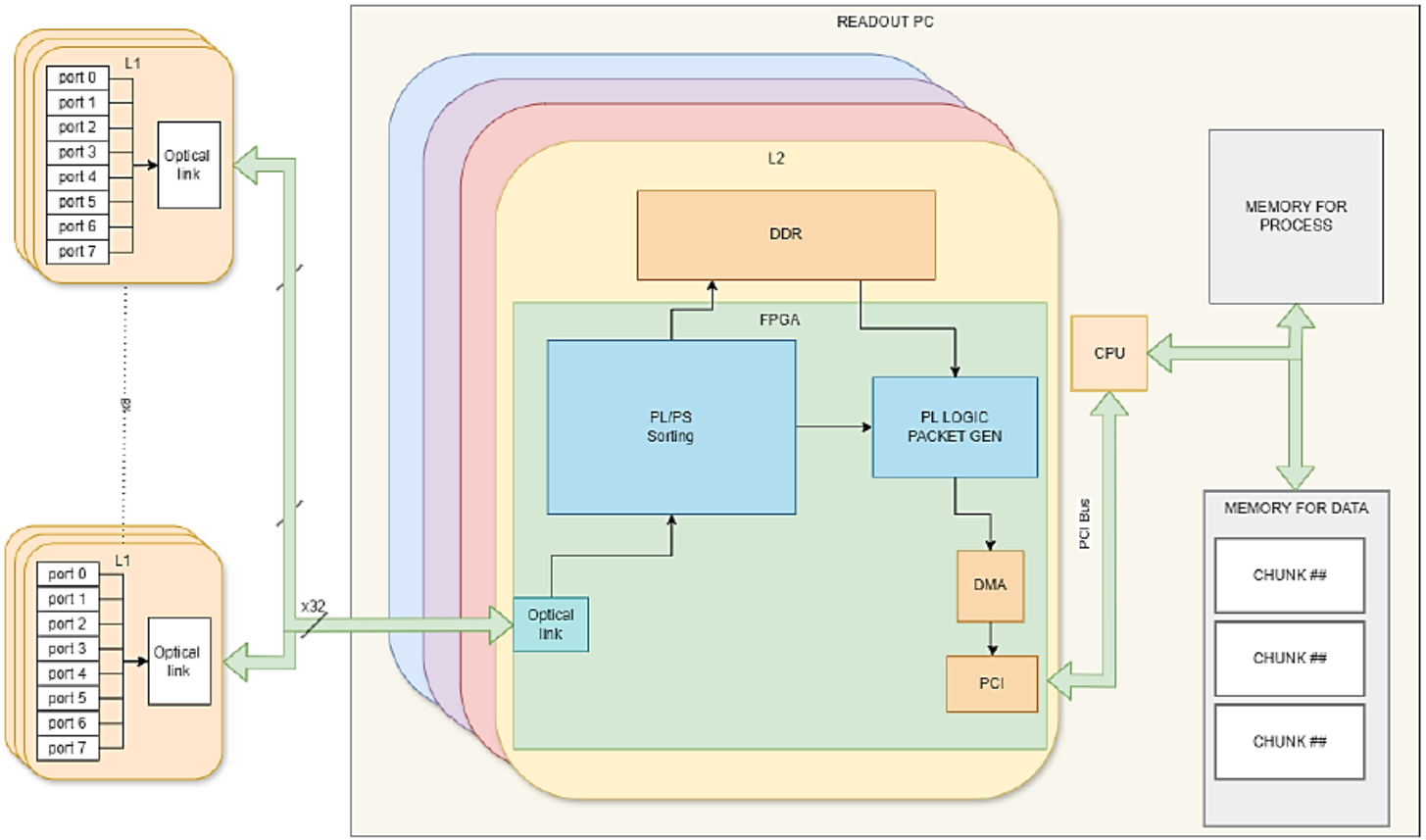
- Данные с заголовком о FEE и номер slice хранятся в промежуточном буфере в произвольном порядке.
- Кластеризация включает в себя сортировку данных по номеру FEE и slice.
- Отсортированные данные передаются в память DDR4.



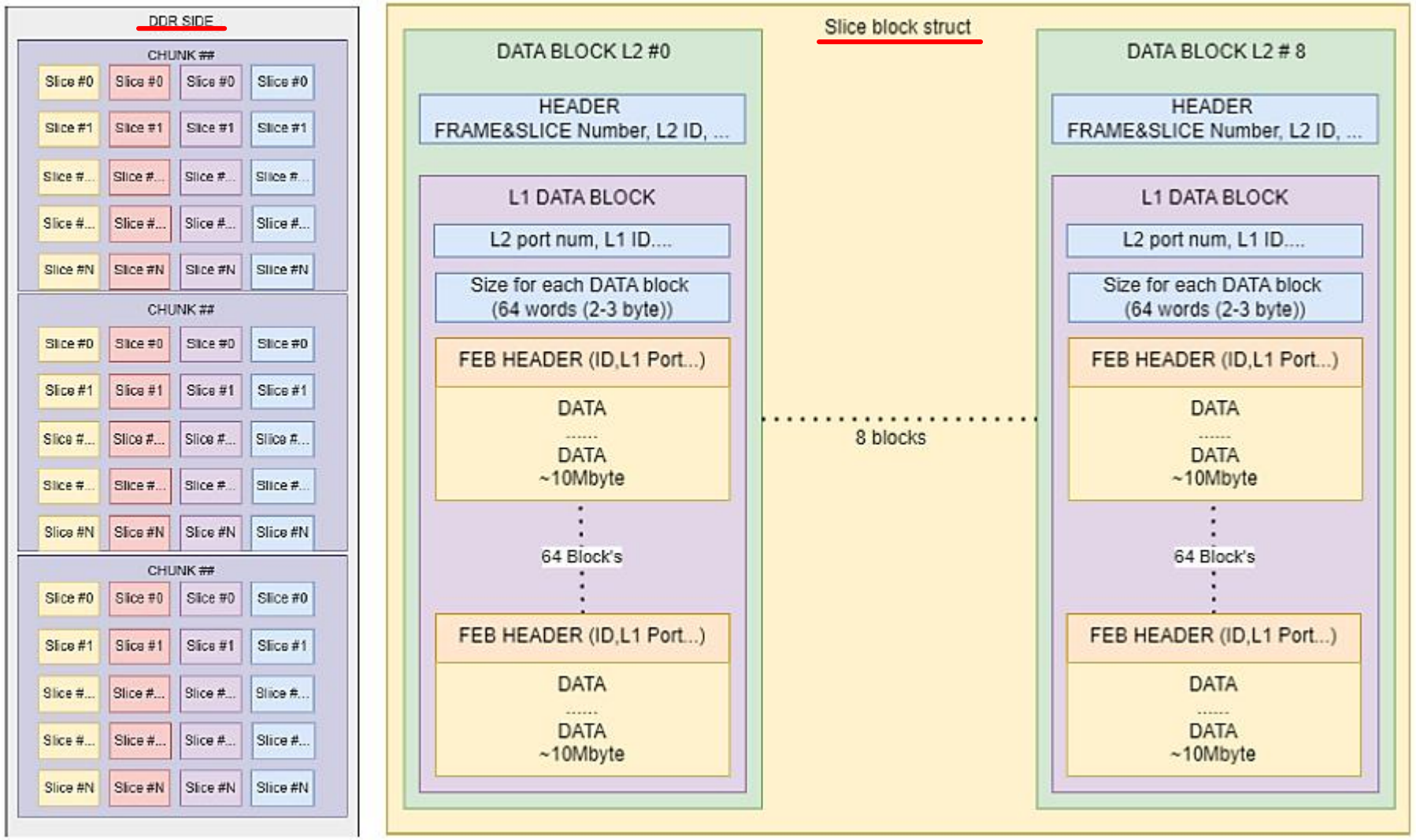
Работа с данными

- FPGA и CPU
- Необходимость реализации кольцевого буфера посредством DMA – dynamic memory access.

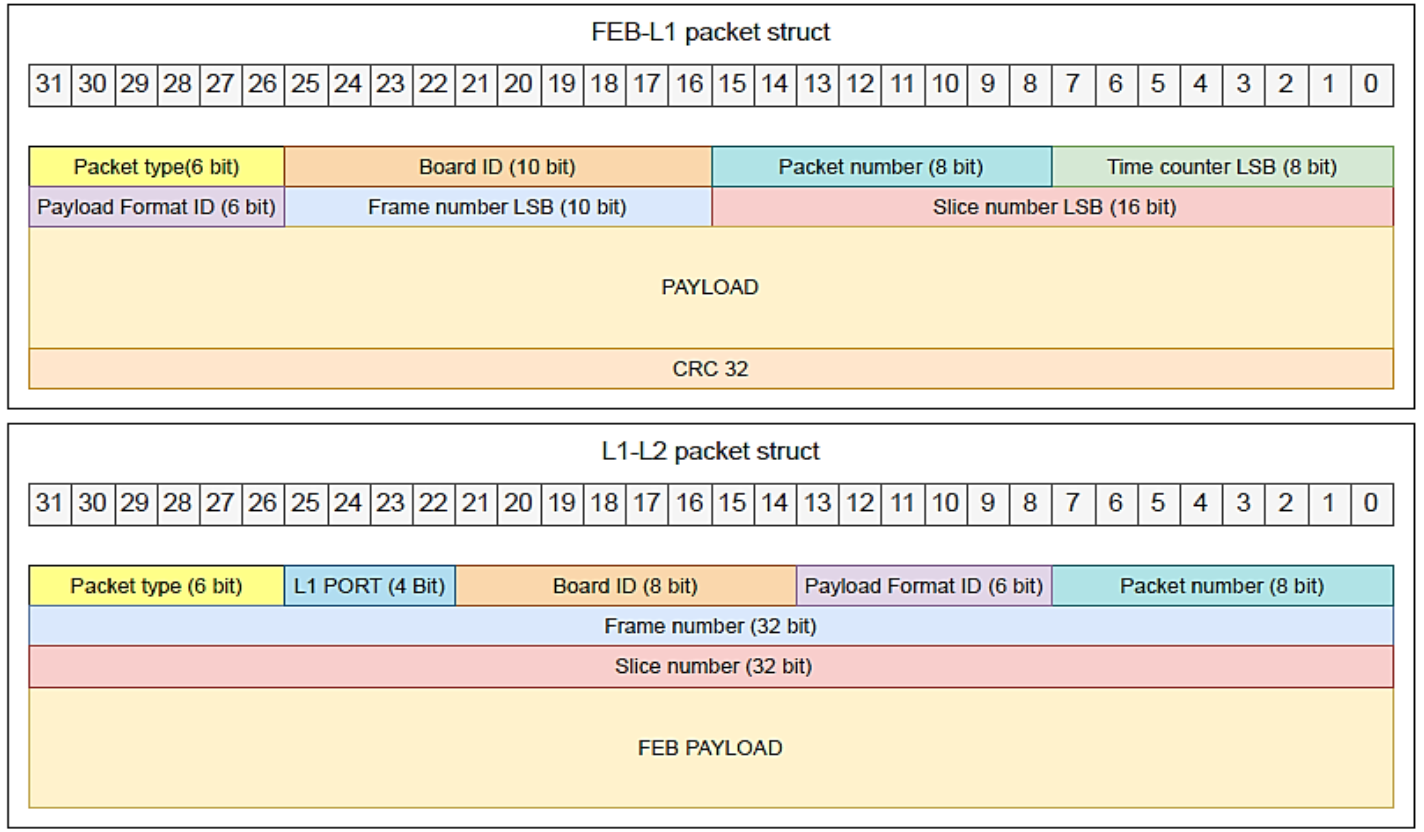




Работа с данными. Формирование пакета

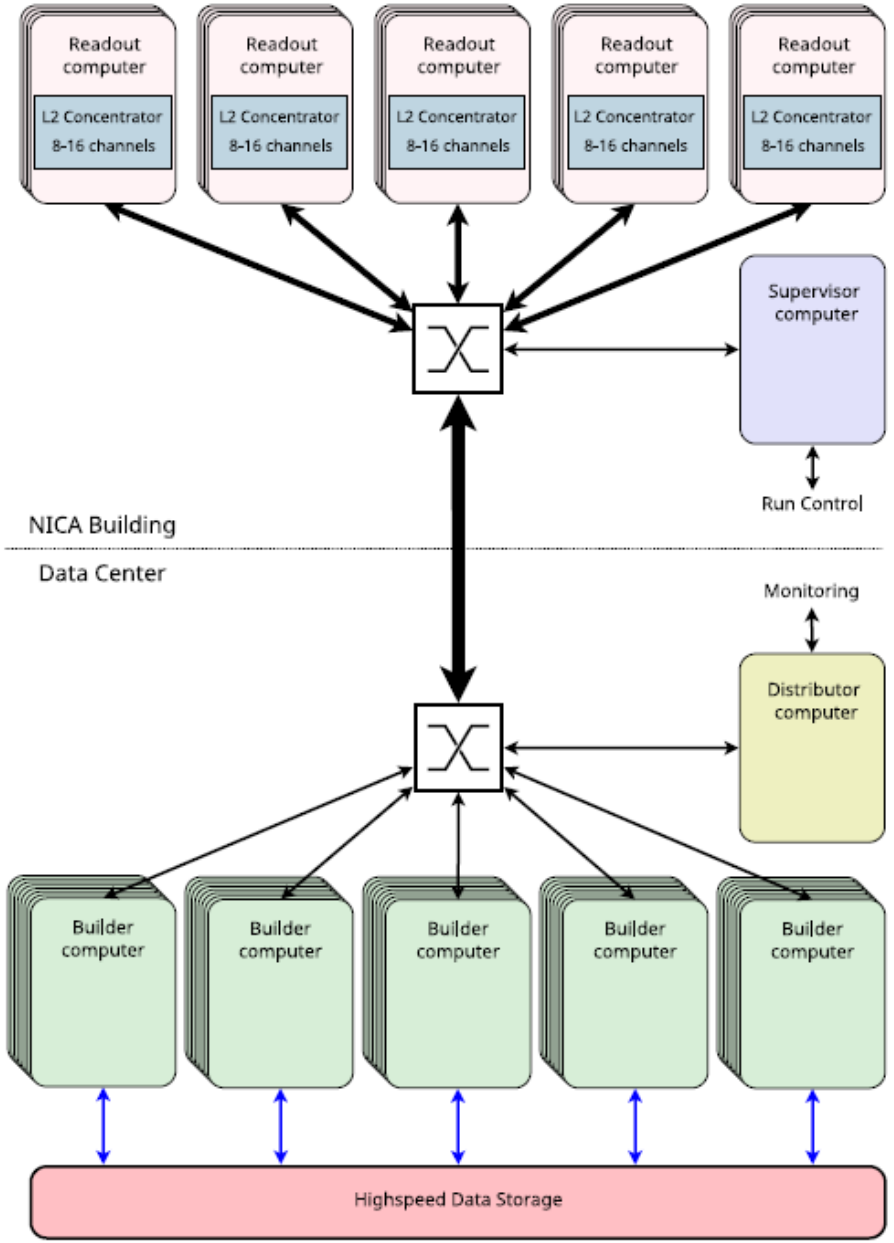


Выходной пакет L1, L2



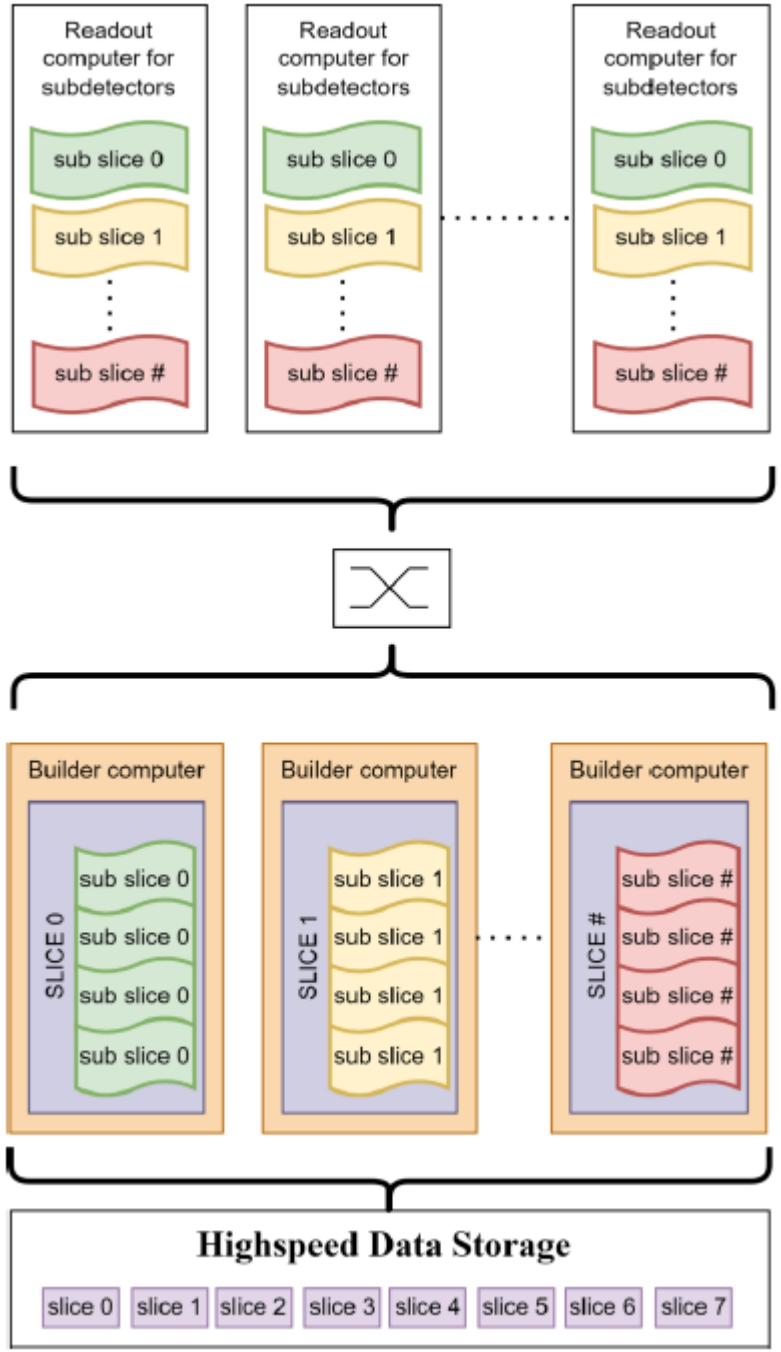
Система построения срезов

- Readout computer – формирование и буферизация части slice, относящихся к конкретной группе детекторов.
- Builder computer – формирование цельного slice из данных каждого readout computer и сохранение в промежуточное хранилище.
- Distributor computer –организация мониторинга работы системы.
- Supervisor computer – управление набором данных и цепочками чтения.
- Highspeed Data Storage – промежуточное хранилище объёмом 2-4 Пб для хранения данных, подготовленных для онлайн фильтра.

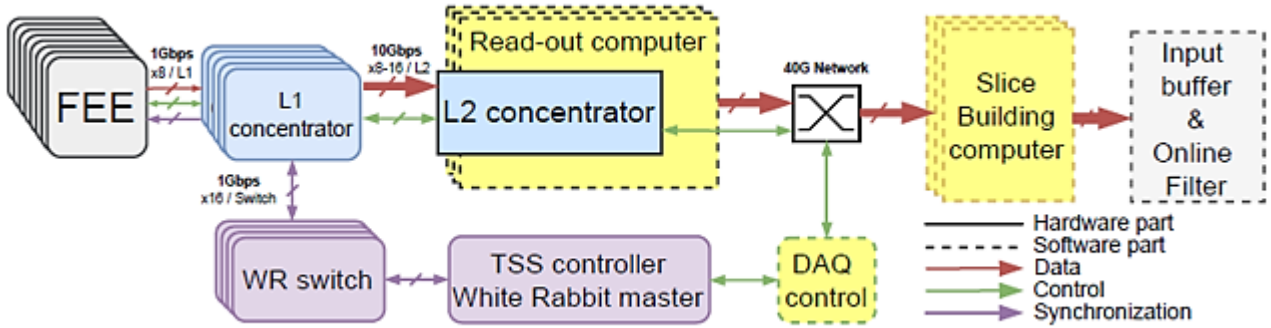


Система построения срезов

- В системе построения срезов данные обрабатываются по частям, каждая часть охватывает определённый период времени (кадр).
- В некоторых случаях объем данных в кадре может быть слишком большим для удобной обработки. Поэтому кадр программно делится на части, называемые блоками, кусками (chunks). Разумная длина куска составляет около 1 с и может быть выбрана с учётом фактического потока данных.

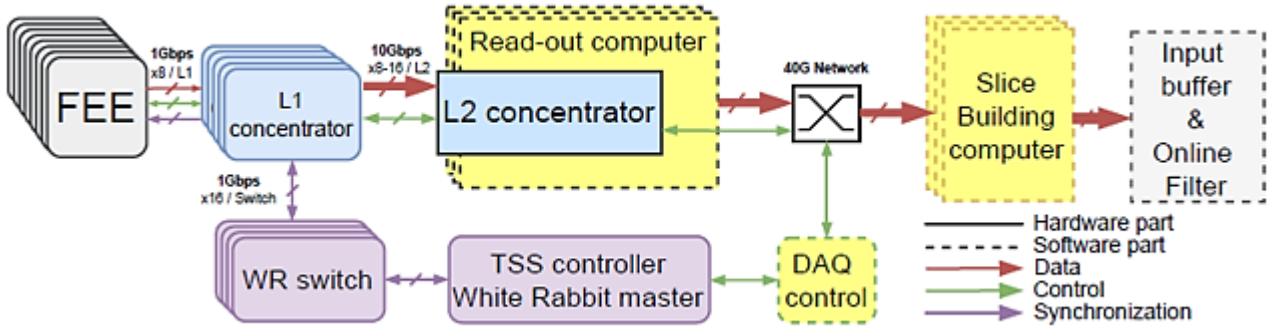


Синхронизация



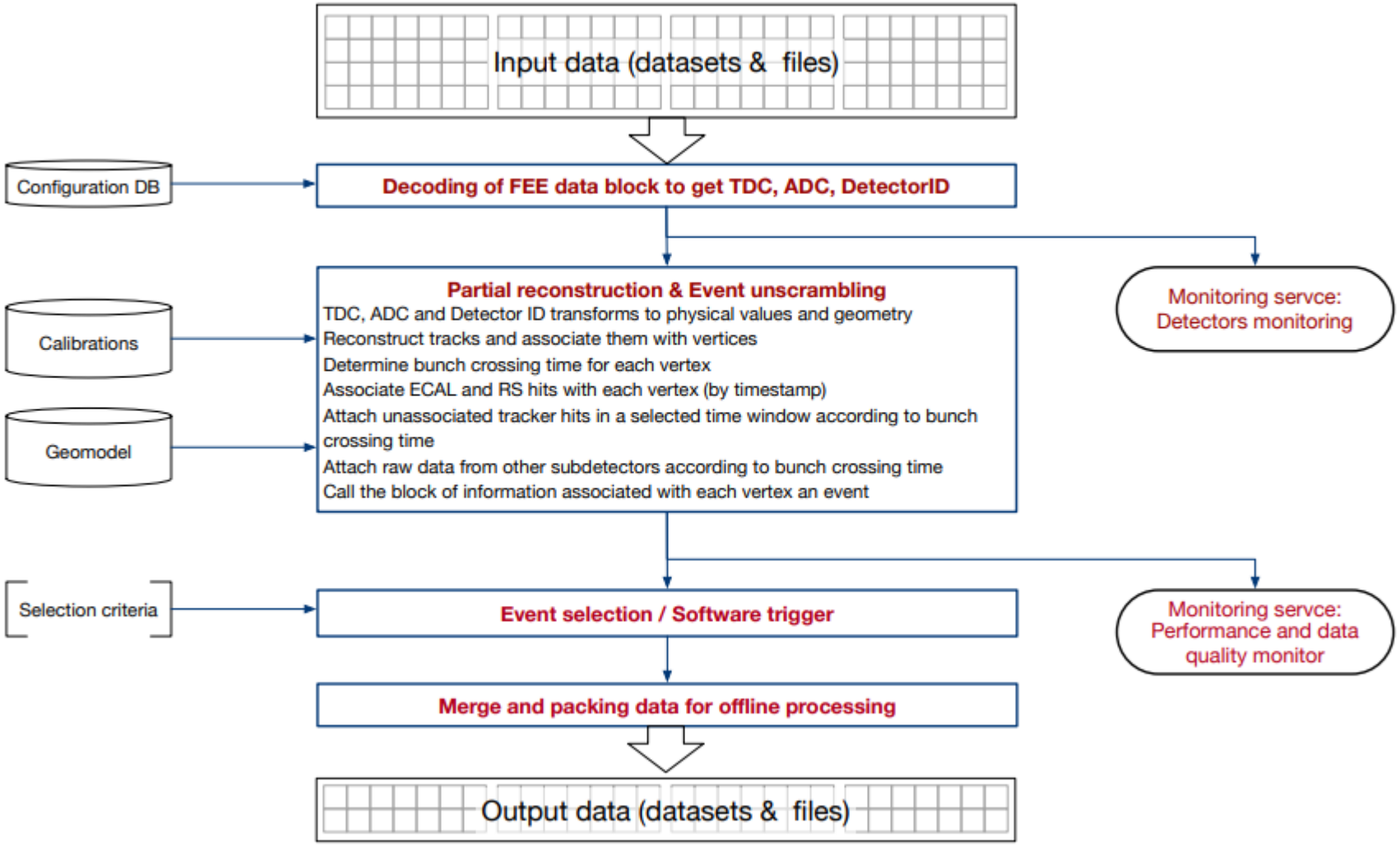
- Задача системы синхронизации
  - генерация и распределение тактовой частоты 125 МГц с точностью лучше 1 нс и джиттером лучше 50 пс;
  - Генерация и распределение синхронных команд по детектору
  - Распределение информации о текущем slice и frame
  - Стабильная поддержка сетей до 10 км
- Масштабируемая система. Позволит адаптироваться к увеличению числа каналов детекторов по мере перехода эксперимента на новые этапы реализации (сначала обработка со 180 000 каналов, с расширением до 600 000 в полной конфигурации).

Синхронизация



- Три основные подсистемы DAQ:
  - Тракт считывания
  - Система временной синхронизации
  - Система формирования сегментов данных
- Система временной синхронизации TSS основана на технологии White Rabbit.
- Обеспечивает точность синхронизации всех субдетекторов на субнаносекундном уровне, что позволяет точно и согласованно присваивать временные метки событиям.
- Тракт считывания гарантирует эффективный сбор и передачу данных
- Система формирования сегментов организует полученные данные во временные фрагменты для последующего анализа и хранения.
- Масштабируемая система. Позволит адаптироваться к увеличению числа каналов детекторов по мере перехода эксперимента на новые этапы реализации (сначала обработка со 180 000 каналов, с расширением до 600 000 в полной конфигурации).

SPD Online Filter



RegMap (SystemRDL)

accellera SYSTEMS INITIATIVE

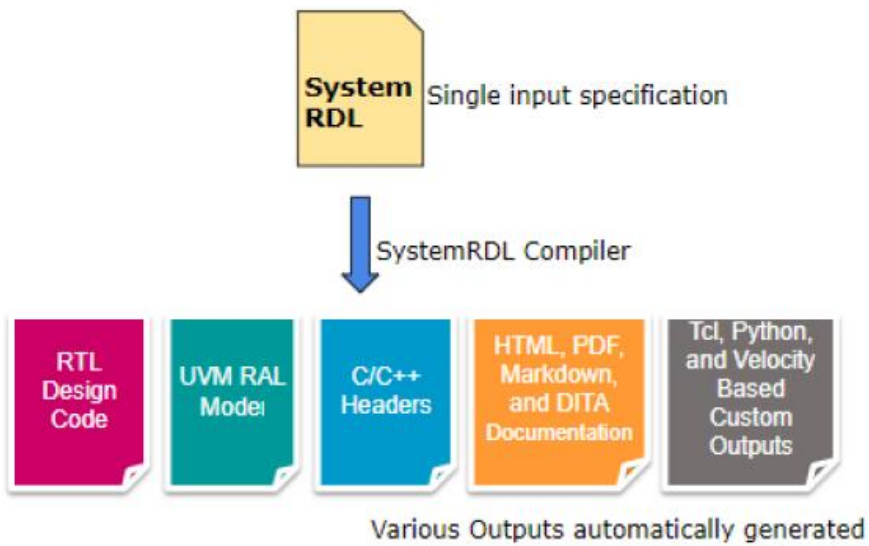
SystemRDL 2.0
Register Description Language

PeakRDL

This tool can:

+

- Process SystemRDL 2.0 register descriptions.
- Generate synthesizable SystemVerilog RTL register blocks.
- Generate a C register abstraction header for software.
- Import & export IP-XACT XML.
- Create rich and dynamic HTML documentation.
- Build a UVM register model abstraction layer.



System RDL language

```
`ifdef OFFSET
    `define OFFSET_CONDITION OFFSET_GLOBAL::teng_map_addr
`else
    `define OFFSET_CONDITION 0
`endif
addrmap teng_map {
    name = "LINK CONFIG REG";
    desc = "Register description of LINK MAIN MODULE. Includes setti

    default regwidth = 32;
    default addressing = fullalign;
    default accesswidth = 32;
    default sw = rw;
    default hw = r;

    reg {
        name = "source_mac_part1";
        regwidth = 32;
        field {
            sw = w;
            hw = r;
            desc = "Source mac addr1";
        } source_mac_part1 [32] = 32'h00_1b_21_ed;
    } source_mac_part1 @ `OFFSET_CONDITION;

    reg {
        name = "source_mac_part2";
        regwidth = 32;
        field {
           sw = w;
           hw = r;
           desc = "Source mac addr2";
        } source_mac_part2 [16] = 16'h00_FE;
        field {
        } reserved [16];
    } source_mac_part2;

    reg {
        name = "source_ip";
        regwidth = 32;
        field {
            desc = "Source ip addr";
        } source_ip [31:0] = 32'hC0_A8_00_FE;
    } source_ip;
```

System Verilog module

```
module teng_map (
        input wire clk,
        input wire rst_n,

        output logic s_axil_awready,
        input wire s_axil_awvalid,
        input wire [31:0] s_axil_awaddr,
        input wire [2:0] s_axil_awprot,
        output logic s_axil_wready,
        input wire s_axil_wvalid,
        input wire [31:0] s_axil_wdata,
        input wire [3:0]s_axil_wstrb,
        input wire s_axil_bready,
        output logic s_axil_bvalid,
        output logic [1:0] s_axil_bresp,
        output logic s_axil_arready,
        input wire s_axil_arvalid,
        input wire [31:0] s_axil_araddr,
        input wire [2:0] s_axil_arprot,
        input wire s_axil_rready,
        output logic s_axil_rvalid,
        output logic [31:0] s_axil_rdata,
        output logic [1:0] s_axil_rresp,

        output teng_map_pkg::teng_map__out_t hwif_out
    );
```

Docs (markdown)

teng_map address map

- Absolute Address: 0x0
- Base Offset: 0x0
- Size: 0x134

Register description of LINK MAIN MODULE. Includes settings

Offset	Identifier	Name
0x100	source_mac_part1	source_mac_part1
0x104	source_mac_part2	source_mac_part2
0x108	source_ip	source_ip
0x10C	dest_mac_part1	dest_mac_part1
0x110	dest_mac_part2	dest_mac_part2
0x114	dest_ip	dest_ip
0x118	config_ports	Config port
0x11C	jitter_ports	Jitter port
0x120	data_ports	Data port
0x124	gateway_ip	gateway_ip
0x128	subnet_mask	subnet_mask
0x12C	padding	padding
0x130	configs	config

C-h file for software

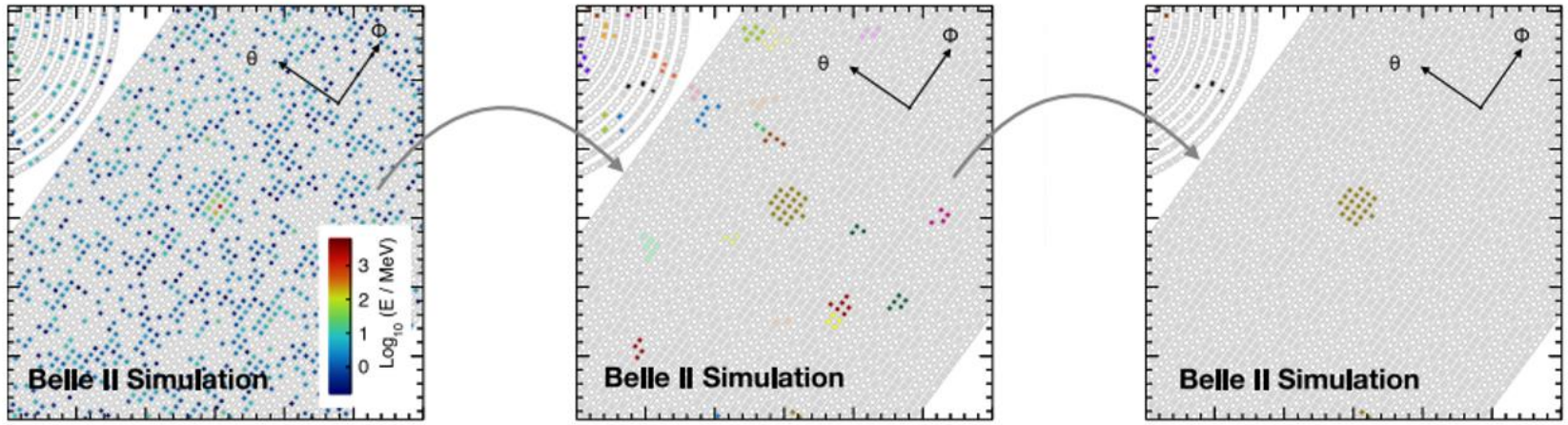
Belle II как пример гибридной ССД

- Первичный аппаратный триггер с задержкой в 5 мкс и частотой 30 кГц
- Высокоуровневый триггер – суперкомпьютерный кластер
- Замена старых процессорных плат COPPER на современные PCIe40 на базе ПЛИС Intel Arria 10.
- Пропускная способность – 100 Гбит/с на плату

Belle II как пример гибридной ССД

ECL Trigger: новый алгоритм кластеризации

- Первый в мире GNN-алгоритм реконструкции работающий в режиме реального времени
- Динамическое построение графа: Система не просто смотрит на энергию. Она представляет триггерные ячейки как узел графа. Связи (ребра) между соседними узлами показывают, как они влияют друг на друга.
- Нейросеть обучена «стягивать» узлы графа в центры кластеров. Она сама решает, сколько кластеров в событии и где их центр, без жестких порогов.
- GNN прошита прямо в FPGA платы UT4, работает синхронно с частотой 8 МГц и укладывается в 3.168 мкс задержки.
- Точность позиции кластера увеличилась на 20%.



Belle II как пример гибридной ССД

Graph Neural Network (GNN) Hit Clean-up

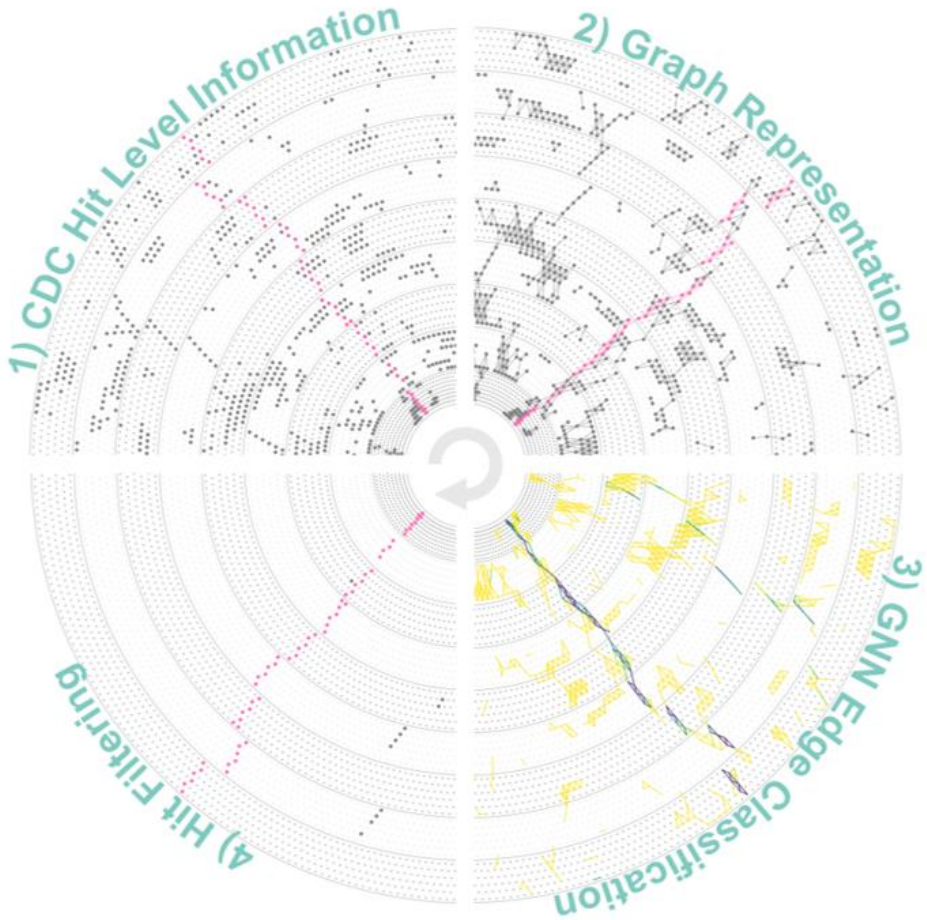
Масштабное моделирование — ключевой источник данных для обучения нейронных сетей триггера, где размечены как сигнальные, так и фоновые события.

Этапы фильтрации:

1. Извлечение и предварительная фильтрация
2. Представление с помощью графов
3. Классификация границ с помощью GNN
4. Фильтрация хитов

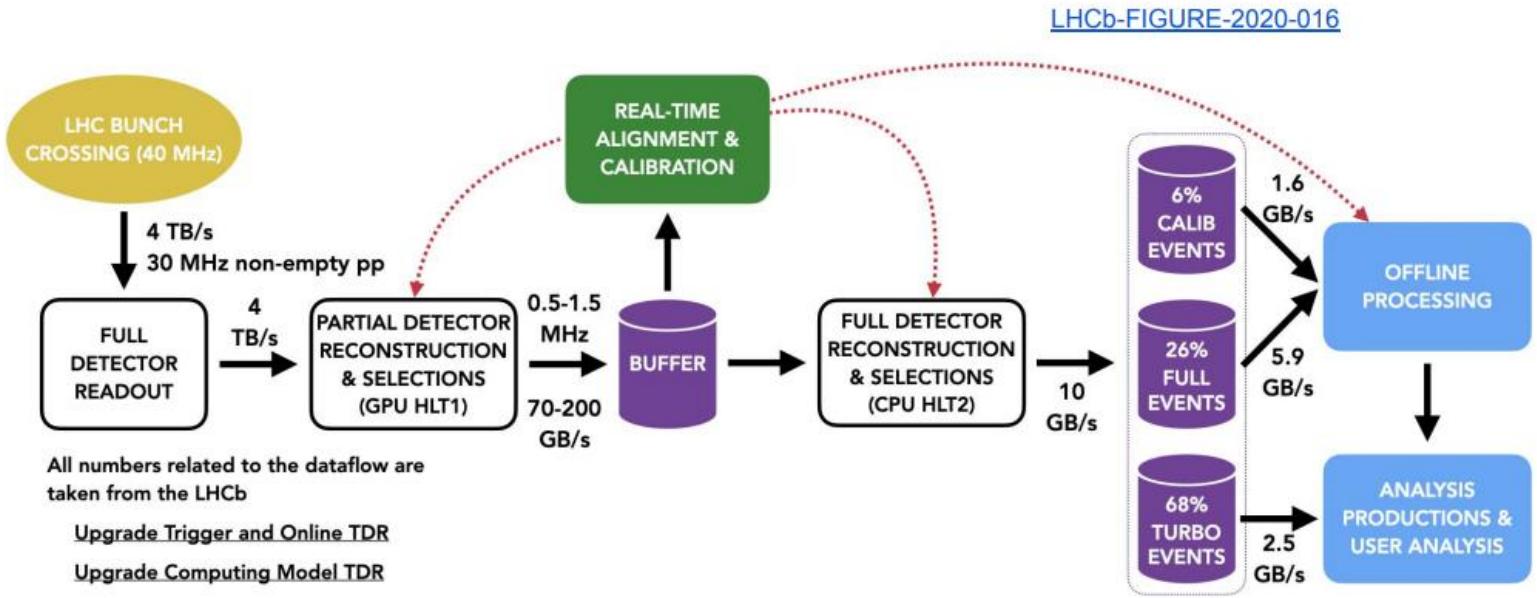
GNN-фильтрация хитов :

- сохраняет 95% истинных хитов, удаляет 83% фоновых хитов
- Эффективность реконструкции треков увеличилась 8% (при той же частоте ложных срабатываний)
- Задержка: работа нейросети < 500 нс



LHCb. Функционирующая новая ССД

The Run 3 data flow



- Detector data @30 MHz received by O(500) FPGAs
- 2-stage software trigger, HLT1 & HLT2
- Real-time alignment & calibration
- After HLT2, 10 GB/s of data for offline processing

LHCb. Функционирующая новая ССД

HLT1 trigger

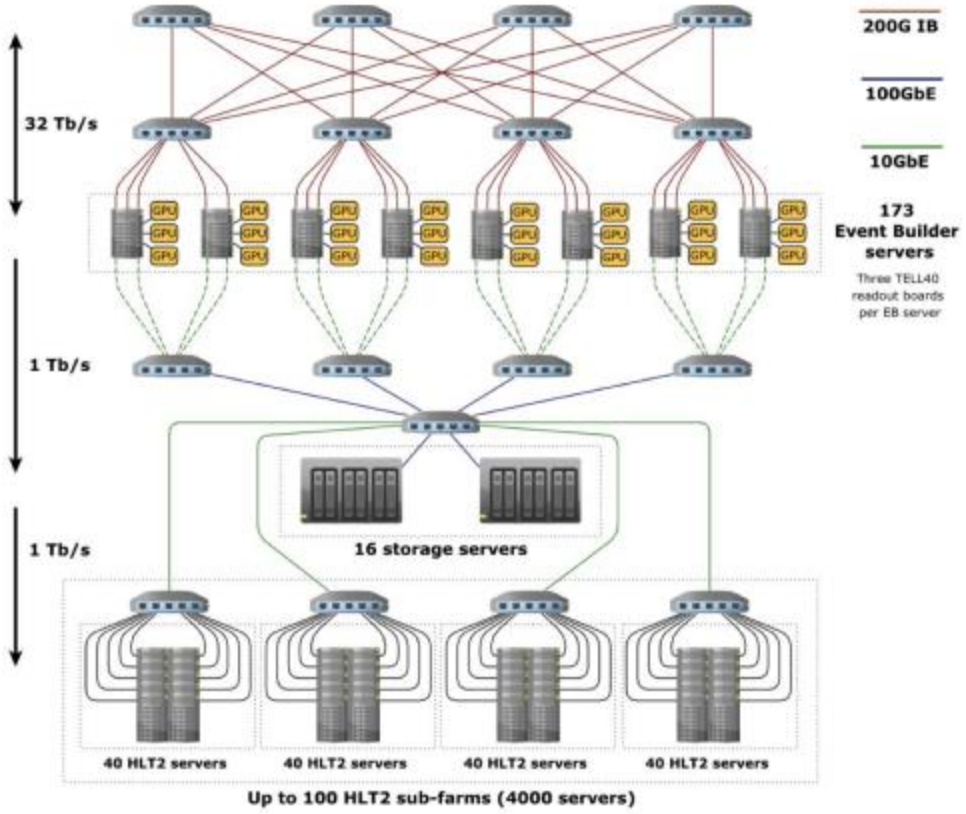
- Take as input LHCb raw data (**4 TB/s**) at 30 MHz
- Perform partial event reconstruction & coarse selection to cover the full breadth of LHCb physics
- Reduce the input rate by a factor of 30 (~1 MHz)
- **~ 500 GPUs NVIDIA RTX A5000 GPUs** installed
  - The baseline TDR design could be achieved with 300 GPUs
  - Extra GPU power used to extend the improvements beyond-TDR

LHCb HLT2 trigger

- HLT2 runs a full reconstruction and all the necessary selections (inclusive but mostly exclusive) for the wide LHCb physics programme (~3000 lines)
- Given the hard limit on bandwidth (10 GB/s to tape and 3.5 GB/s on disk) and expected signal rate, event size is the only free parameter
- Need to "persist" all the reconstructed objects for offline analysis
- The successful strategy of the Turbo paradigm used at full speed also in Run 3

Полностью программный отбор данных в реальном времени. GPU фермы + «точечный» анализ на CPU

LHCb. Функционирующая новая ССД



Полностью программный отбор данных в реальном времени. GPU фермы + «точечный» анализ на CPU

Основные выводы

- Начиная с эксперимента PANDA, в детекторах на ускорительных установках стали использовать FPGA не для триггера, а вместо триггера, с простановкой временных меток для хитов.
- В одних экспериментах FPGA не чувствительны к физике, не участвуют в отборе данных, являются концентраторами-упаковщиками для передачи данных на следующий уровень ССД.
- В другой реализации, FPGA производят фильтрацию сигналов, формируют набор триггерной информации (триггерные примитивы).
- Использование связки FPGA+DDR позволяет сортировать данные – считывается временная метка каждого хита, вычисляется для него уникальный адрес ячейки в памяти DDR и раскладывается в памяти. В ПК отправляется отсортированный хронологический блок.
- Реализация потоковой или бестриггерной ССД индивидуальна для каждого эксперимента.

Основные выводы

- Эксперименты нового поколения строятся изначально с потоковым чтением.
- Современные работающие установки как Belle II на ходу перестраивают свои системы сбора данных подсистем детектора.
- Модернизируют высокоуровневый триггер с использованием нейросетей GNN для поисков кластеров и генерации триггерного сигнала (ECL, CDC), а также для высокоуровневого триггера.
- На SPD используются решения, разрабатываемые для экспериментов на FAIR, а также LHC.
- Успешное внедрение такой системы произошло на детекторах ALICE и LHCb на LHC.
- Реализация первичной обработки данных определяет реализацию конечной реконструкции событий – CPU ферма либо GPU ферма.

Заключение

Зачем нужна триггерная система?

Классическая триггерная система

Данные непрерывно буферизируются и обрабатываются триггером. В случае положительного решения триггера данные записываются.

Плюсы:

- Низкий поток записываемых данных
- Фиксированное время задержки
- Меньшие требования к вычислительным ресурсам для постобработки
- Хорошо подходит для высокой частоты столкновений (LHC, Belle II)

Минусы:

- Риск потерять редкое или нестандартное событие, не попавшее под критерии триггера
- Жёсткая логика (сложно менять условия отбора без перепрошивки FPGA)
- Требует глубокого понимания физических сигналов на этапе проектирования

Бестриггерная система

Данные непрерывно читаются и отправляются на запись с временными метками. Событие собирается программно — по совпадению временных меток от разных подсистем.

Плюсы:

- Не теряются редкие / экзотические события
- Гибкость — условия отбора можно менять постфактум, переобрабатывая данные
- Идеальна для экспериментов с низкой частотой событий или сложной топологией
- Позволяет искать новую физику

Минусы:

- Огромные объёмы записываемых данных (терабайты/петабайты)
- Высокие требования к вычислительным ресурсам для обработки
- Сложность синхронизации временных меток от разных подсистем
- Не подходит для экспериментов с высокой частотой столкновений (>МГц) без агрессивного сжатия

Back-up

Работа с данными (кластеризация)

Clusterization

- The new version of the clustering architecture uses **Scatter Gather** technology
- The **arbiter** sorts and buffers data from each L2 hub channel
- To use the Scatter Gather DMA technology, it was necessary to develop a **ring buffer** with addresses and column lengths in DDR4. This memory is used to **generate descriptors** and transfer them to the Scatter Gather DMA.

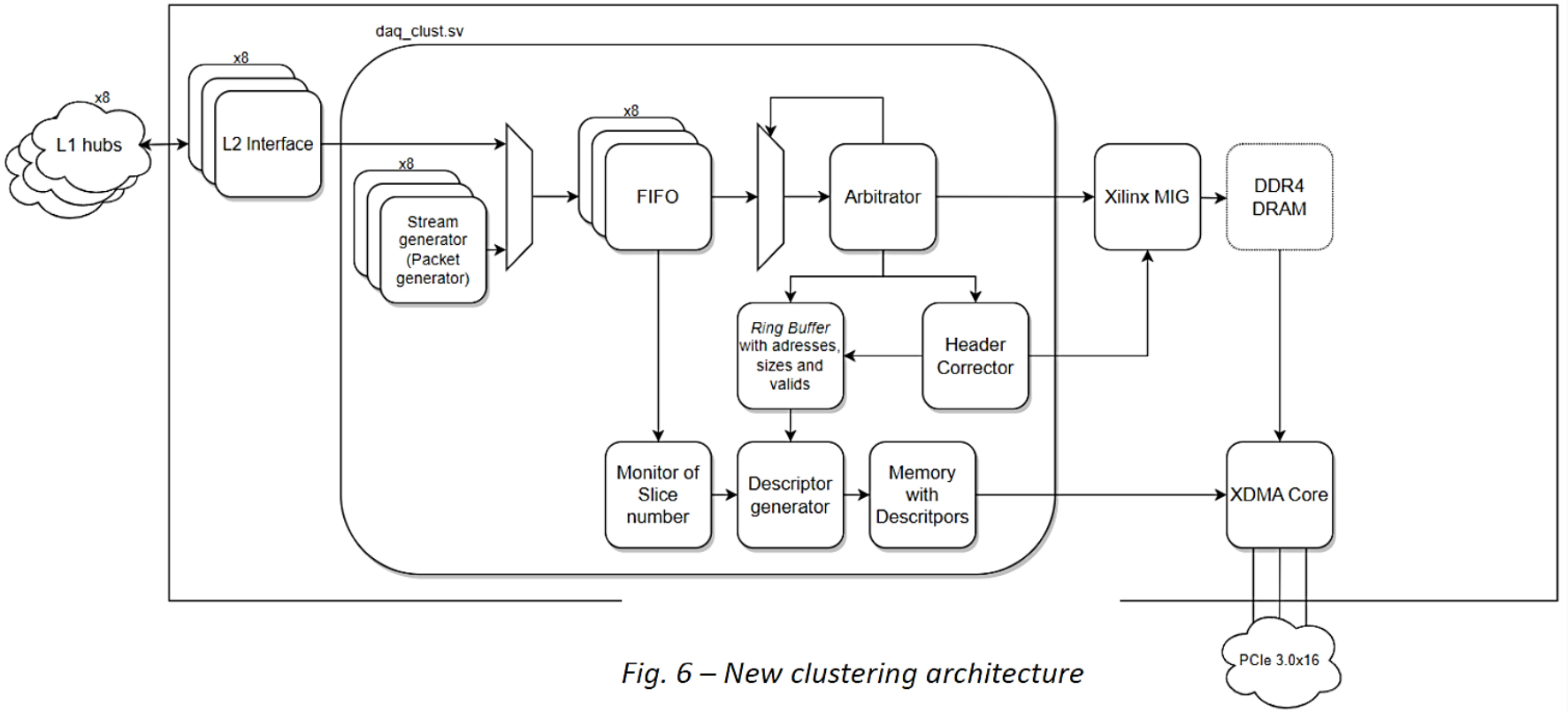
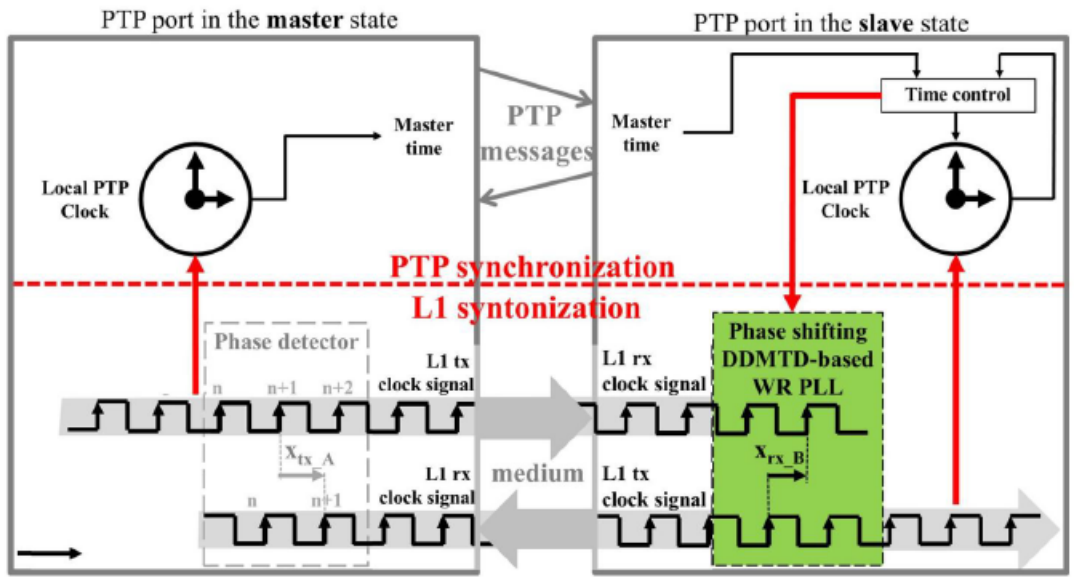
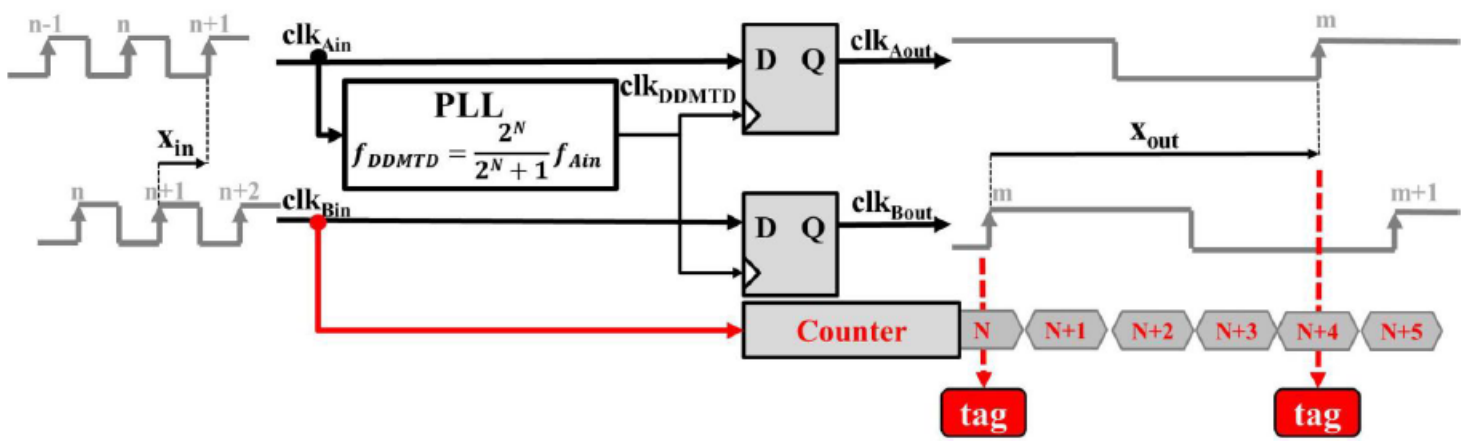
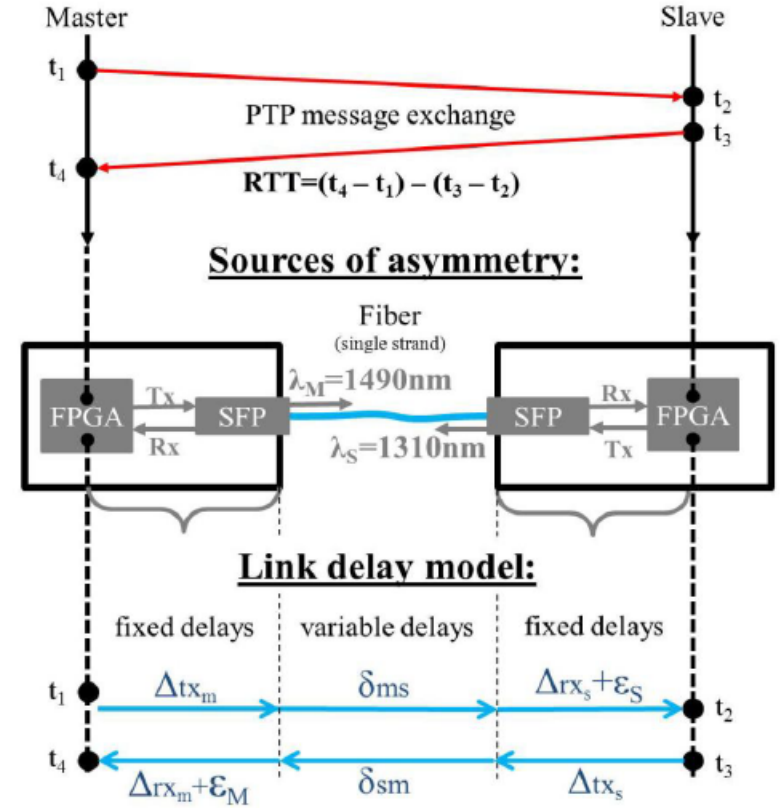


Fig. 6 – New clustering architecture


PTP port in the **master** state
PTP port in the **slave** state
Master time
Local PTP Clock
PTP messages
Master time
Time control
Local PTP Clock
PTP synchronization
L1 syntonization
Phase detector
L1 tx clock signal
L1 rx clock signal
Phase shifting DDMTD-based WR PLL
x_{tx_A}
x_{rx_B}
L1 rx clock signal
medium
L1 tx clock signal


n-1
n
n+1
clk_{Ain}
PLL
$f_{DDMTD} = \frac{2^N}{2^N+1} f_{Ain}$
clk_{DDMTD}
D Q
clk_{Aout}
m
x_{in}
x_{out}
n
n+1
n+2
clk_{Bin}
D Q
clk_{Bout}
m
m+1
Counter
N
N+1
N+2
N+3
N+4
N+5
tag
tag


Master
Slave
t_1
t_2
PTP message exchange
t_3
t_4
$RTT=(t_4 - t_1) - (t_3 - t_2)$
Sources of asymmetry:
Fiber
(single strand)
FPGA
Tx
Rx
SFP
λ_M=1490nm
λ_S=1310nm
SFP
Rx
Tx
FPGA
Link delay model:
fixed delays
variable delays
fixed delays
t_1
Δtx_m
δms
$\Delta rx_s+\varepsilon_S$
t_2
t_4
$\Delta rx_m+\varepsilon_M$
δsm
Δtx_s
t_3