Объединённый институт ядерных исследований
Лаборатория физики высоких энергий

Овчаренко
Егор Владимирович

Разработка методов моделирования физических установок и сбора данных и их применение для детектора черенковских колец эксперимента CBM

Специальность 01.04.01 -
Приборы и методы экспериментальной физики

Диссертация на соискание учёной степени
кандидата физико-математических наук

Научный руководитель: к.ф.-м.н. С.Г. Белогуров

Дубна - 2017

Содержание

Введение **4**

1 Эксперимент CBM на FAIR **5**
1.1 Физическая программа CBM 5
1.2 Экспериментальная установка CBM 5
1.2.1 Вершинный микродетектор MVD 5
1.2.2 Кремниевая трекинговая система STS 5
1.2.3 Детектор черенковских колец RICH 6
1.2.4 Мюонная система MUCH 6
1.2.5 Детектор переходного излучения TRD 7
1.2.6 Время-пролётный детектор TOF 7
1.2.7 Электромагнитный калориметр ECAL 7
1.2.8 Детектор PSD 7
1.3 CBM RICH поподробней 7

2 Геометрия **8**
2.1 Сравнение представления геометрии в GEANT/ROOT и САПР 11
2.1.1 Представление геометрии в САПР 11
2.1.2 Представление геометрии в GEANT/ROOT 12
2.2 “CATIA-GDML geometry builder” 13
2.2.1 Примитивы в “Builder” 16
2.2.2 Макропрограммы для CATIA v5 16
2.2.3 Избранные подробности реализации “CATIA-GDML geometry builder” 20
2.3 Применение “CATIA-GDML geometry builder” к CBM RICH . 21

3 Результаты **22**
3.1 Время-цифровой преобразователь 22
3.2 Обработка сигнала с детектора 24
3.2.1 Бестриггерная передняя электроника 25
3.2.2 26
3.2.3 FLES 27
3.2.4 Построение события 27
3.2.5 LeadingEdgeDiff 28
3.3 Исследование спектросместителя с помощью флюориметра . 30
3.4 Прямые измерения временного профиля спектросместителя . 31
3.4.1 Прямые фотоны 31
3.4.2 Фитирование WLS_diff 32

Заключение **33**

Список литературы **34**

Введение

Глава 1

Эксперимент CBM на FAIR

1.1 Физическая программа CBM

1.2 Экспериментальная установка CBM

1.2.1 Вершинный микродетектор MVD

1.2.2 Кремниевая трекинговая система STS

From CBM STS TDR:

The detector system's task is to measure the trajectories and momenta of charged particles originating from the interactions of heavy-ion beams with nuclear targets. Up to 1000 charged particles are produced per interaction, at rates up to 10 MHz to enable CBM physics with rare observables. The track reconstruction has to be achieved with 95% efficiency and a momentum resolution $\Delta p/p = 1\%$. These requirements can be fulfilled with a tracking system of 8 low-mass layers of silicon microstrip sensors located at distances between 30 cm and 100 cm downstream of the target inside the magnetic dipole field. The sensors are mounted onto lightweight mechanical support ladders and read out through multi-line micro-cables with fast self-triggering electronics at the periphery of the stations where cooling lines and other infrastructure can be placed. The micro-cables will be built from sandwiched polyimide-Aluminum layers of several $10\mu m$ thickness. The microstrip sensors will be double-sided with a stereo angle of 7.5°, a strip pitch of $58\mu m$, strip lengths between 20 and 60 mm, and a thickness of $300\mu m$ of silicon. According to the CBM running scenario the maximum non-ionizing dose for the sensors closest to the beam line does not exceed $10^{14}\ n_{eq}\ cm^{-2}$. The STS is operated in a thermal enclosure that keeps the sensors at a temperature of about $-5\,°C$. The heat dissipated in the read-out electronics is removed by a CO_2 cooling system. The mechanical structure of the detector system including the service and signal connections is designed such that single detector ladders can be exchanged without disconnecting and removing more than one detector station.

Задача кремниевой трековой системы — измерение траекторий и импульсов заряженных частиц, вылетающих из точки взаимодействия пучка тяжёлых ионов с мишенью. Для выполнения физической программы CBM — наблюдения редких **явлений** — необходима частота взаимодействий до

10 МГц, при том что в одном взаимодействии будет рождаться до 1000 заряженных частиц. Реконструкуия треков должна выполняться с эффективностью порядка 95% и разрешением по моменту порядка $\Delta p/p = 1\%$. Для удовлетворения перечисленных требований STS должна состоять из 8 слоёв кремниевых микростриповых сенсоров, расположенных внутри поля от дипольного магнита на растоянии от 30 см до 100 см от точки взаимодействия вниз по пучку с шагом 10 см. Сенсоры будут монтироваться на легкую механическую опору в виде карбоновый ферм. Считывание будет осуществляться по многоканальных микро-кабелям самотриггирующейся электроникой, расположенной по краям станций вместе с линиями охлаждения и другими инфраструктурными подсистемами. Микро-кабели будут выполнены из sandwiched polyimide-Aluminum layers толщиной несколько десятков мкм. Микростриповые сенсоры будут

1.2.3 Детектор черенковских колец RICH

1.2.4 Мюонная система MUCH

From CBM MUCH TDR:

The MuCh system is designed to identify muon pairs which are produced in high-energy heavy-ion collisions in the beam energy range from 4 to 40 AGeV. The measurement of lepton pairs is a central part of the CBM research program, as they are very sensitive diagnostic probes of the conditions inside the fireball. At low invariant masses, dileptons provide information on the in-medium modification of vector mesons which is a promising observable for the restoration of chiral symmetry. At intermediate invariant masses, the dilepton spectrum is dominated by thermal radiation from the fireball reflecting its temperature. At invariant masses around $3GeV/c^2$, dileptons are the appropriate tool to study the anomalous charmonium suppression in the deconfined phase. In the CBM experiment both electrons and muons will be measured in order to obtain a consistent and comprehensive picture of the dilepton physics.

The experimental challenge for muon measurements in heavy-ion collisions at FAIR energies is to identify low-momentum muons in an environment of high particle densities. The CBM strategy is to track the particles through a hadron absorber system, and to perform a momentum-dependent muon identification. This concept is realized by an instrumented hadron absorber, consisting of staggered absorber plates and tracking stations. The hadron absorbers vary in material and thickness, and the tracking stations consist of detector triplets based on different technologies. The MuCh system is placed downstream of the dipole magnet hosting the Silicon Tracking System (STS) which determines the particle momentum. In order to reduce the number of muons from pion and kaon weak decays, the absorber/detector system has to be as compact as possible.

The MuCh system will be built in stages which are adapted to the beam energies available. Within the FAIR modularized start version the SIS100 ring will provide heavy ion beams with energies up to 14 AGeV, and proton beams up to 29 GeV. The first two versions of MuCh (SIS100-A and SIS100-B) will comprise of 3 and 4 stations suitable for the measurement of low-mass vector mesons in $A + A$ collisions at 4-6 AGeV and 8-14 AGeV, respectively. The third version of the MuCh system (SIS100-C) will be equipped with an additional iron absorber of 1 m thickness in order to be able to identify charmonium at

the highest SIS100 energies. The absorber slices will be built only once so that they could be rearranged properly to obtain required absorber thicknesses. Once SIS300 is operational, we will upgrade the MuCh system further by inserting additional absorbers and detector stations for the measurement of low-mass vector mesons and charmonium at beam energies above 14 AGeV (MuCh versions SIS300-A and SIS300-B).

1.2.5 Детектор переходного излучения TRD

1.2.6 Время-пролётный детектор TOF

1.2.7 Электромагнитный калориметр ECAL

1.2.8 Детектор PSD

1.3 CBM RICH поподробней

Глава 2

Геометрия

Процесс проектирования современной экспериментальной установки подразумевает разнообразное компьютерное моделирование этой установки. В первую очередь выполняется компьютерное геометрическое моделирование в трёхмерном пространстве с целью получения конструкторской документации и анализа расположения элементов в пространстве. Геометрическая модель для этих целей обычно строится средствами систем автоматизированного проектирования (САПР), в которых стандартным способом представления геометрической информации является граничное представление (BREP).

Далее представление геометрической информации в САПР мы будем называть просто BREP, хотя строго говоря это не совсем верно.

Также, как и в любой другой прикладной области, необходимо выполнять многочисленные расчёты, которые нередко требуют геометрическую модель в качестве входных данных. Так, например, в инженерно-конструкторской среде широкое распространение получил метод конечных элементов (МКЭ) для решения задач прочности и устойчивости механических конструкций, динамики жидкостей и газов и т.д. МКЭ получил своё название от способа разбиения расчётной области на элементарные блоки — конечные элементы. В простом случае расчётной областью является пространство заполненное материалом, т.е. сама деталь, а конечным элементом — тетраэдр. Такую модель, в которой деталь представлена множеством конечных элементов, называют КЭ-моделью. Существуют алгоритмы, позволяющие эффективно получить разбиение исходной геометрической модели, например представленной с помощью BREP и построенной в САПР, на конечные элементы. Многие алгоритмы основаны на триангуляции Делоне, разработанной в начале 20 века.

Отличительной особенностью сферы физики частиц является то, что в процессе проектирования установки помимо типовых расчётов требуется выполнение моделирования прохождения частиц через материал, которое чаще всего выполняется физиками, формулирующими требования к конструкции установки, но в общей массе не владеющими САПР. Такое моделирование достаточно специфично, но оно также выполняется над геометрической моделью, в идеальной ситуации — максимально подробной, совпадающей с полной детальной моделью, полученной инженерами-

конструкторами с помощью САПР. Также стоит отметить, что процесс конструирования, в том числе получения инженерной геометрической модели, и процесс моделирования физики не имеют чётко определённого порядка и тесно между собой переплетены. В результате обоих процессов уточняются геометрические параметры деталей, компоновка узлов, применяемые материалы, и т.д. Это приводит к необходимости постоянного обмена геометрической информацией.

Как было сказано выше, инженеры для получения геометрической модели используют САПР. Во многих физических лабораториях, включая CERN, GSI и ОИЯИ, применяется САПР CATIA v5. Моделирование взаимодействия частиц с материалом широко применяет метод Монте-Карло (MC) и реализовано в соответствующих программных пакетах, многие из которых основаны на фреймворках GEANT4 или GEANT3, разработанных в CERN. Также часто применяют поход Virtual Monte-Carlo (VMC), в котором все процедуры, связанные с геометрией, поручены системе ROOT. Все перечисленные физические пакеты (GEANT3, GEANT4, ROOT, далее коротко GEANT/ROOT) используют представление геометрии, принципиально отличающееся от BREP. Модели для GEANT/ROOT часто называют MC-моделями. Это отличие состоит из двух пунктов, подробно описанных в 2.1, и приводит к невозможности прямого обмена геометрическими моделями между физиками и инженерами. Существует теоретическая возможность прямой конвертации из представления, принятого в GEANT/ROOT, в BREP, однако в процессе работы не было найдено существующей реализации подобного перехода. Конвертация в обратном направлении до настоящего времени не была математически описана, хотя теоретически также представляется возможной.

Алгоритмы проведения частиц, реализованные в GEANT/ROOT, оптимизированы для соответствующего описания геометрии, применяемого в этих пакетах. Подходы геометрического моделирования, принятые в САПР, обеспечивают максимально эффективную работу как ЭВМ, так и инженеров, в частности за счёт того, что эти подходы интуитивно понятны человеку. Главным фактором против прямой конвертации в том или ином направлении являтся то, что она имеет малую практическую пользу. Одна и та же геометрическая модель с точки зрения разных задач может быть одновременно оптимальна и, наоборот, избыточна или недостаточна. Это просто понять на следующем примере. С точки зрения инженерного проекта массив болтов, вкрученных в корпус, представляет собой важную информацию. В чертежах и другой конструкторской документации ошибка в точном положении отверстий, их диаметре, типе резьбы болтов и т.д. может привести к невозможности собрать продукт после изготовления отдельных компонентов. В то же время, в САПР принято не хранить, и следовательно не визуализировать, витки резьбы с целью снижения нагрузки на графический адаптер ЭВМ. Это значит, что резьба присутствует только формально, в документации, а геометрическая модель имеет лишь условное обозначение резьбы в соответствующем месте. С точки зрения моделирования прохождения частиц через материал в зависимости от расположения в общей установке подобные подробности могут оказаться как критическими, так и наоборот излишними и вызывающими значительное увеличение времени выполнения моделирования. Так, например, резьба болта, находящегося близко к области, где проходит пучок, может оказать влияние

на функционирование всей установки, а та же резьба где-то за пределами геометрического аксептанса не даст ровно никого эффекта может быть упрощена до цилиндра. Более того, без ущерба реалистичности моделирования упрощения могут носить неожиданно масштабный характер. Например, где-то рассматриваемый массив болтов может быть вообще проигнорирован, а пространство в отверстиях заполнено материалом корпуса.

В связи с этим в GEANT/ROOT принятно иметь несколько моделей одной и той же установки, имеющих разный уровень подробностей. Чем выше уровень подробностей — тем больше времени занимает выполнение моделирования. Для оценочных расчётов удобно применять грубые модели, для точного определения каких-либо характеристик — подробные модели. В САПР же подобная проблема решается другим образом. Так, например, в САПР CATIA v5 присутствует возможность автоматического огрубления геометрической модели для снижения нагрузки на графический адаптер и повышения частоты кадров при динамической визуализации трёхмерных объектов. Это становится актуально, когда количество треугольников, которые необходимо визуализировать, составляет десятки миллионов.

Принимая во внимание развитие вычислительной техники, в особенности резкое повышение производительности графических карт, их доступность широким массам, и вообще увеличение их значимости в вычислениях общего назначения, представляется возможным разработка новых алгоритмов проведения частиц через материал, учитывающих особенности геометрического представления в САПР. Более того, возможна также некоторая корректировка подходов САПР к геометрическому моделированию с целью повышения совместимости с пакетами проведения частиц. Однако следует учитывать следующие факты, мешающие движению в данном направлении. Во-первых, САПР — это в большинстве своём коммерческое программное обеспечение с закрытым исходным кодом, а геометрическое ядро САПР — базовая составляющая, которую отлаживают десятилетиями. Внесение изменений в столь важную компоненту коммерческого продукта, вероятно, будет проблемным даже при наличии интереса со стороны фирмы-разработчика. Во-вторых, в обоих сферах накоплен огромный массив моделей, применяемых для поддержки изделий на всех этапах жизненного цикла, даже после окончания процесса проектирования. GEANT/ROOT модели могут применяться для выполнения моделирования даже после того, как физическая экспериментальная установка уже собрана.

Таким образом с целью упрощения взаимодействия физиков и инженеров было принятно решение не пытаться разработать конвертеры или какие-либо новые универсальные способы представления геометрии, а сосредоточиться на облегчении существующей процедуры за счёт плавной корректировки привычных методов и предоставления новых инструментов как физикам, так и инженерам. “CATIA-GDML geometry builder” — это как раз набор таких инструментов. Он описан в 2.2 вместе с предлагаемой организацией рабочего процесса и реальным случаем использования для проектирования детектора RICH эксперимента CBM.

2.1 Сравнение представления геометрии в GEANT/ROOT и САПР

Разница между двумя способами описания геометрической информации в САПР и пакетах моделирования прохождения частиц через материал GEANT/ROOT заключается в двух пунктах. Во-первых, отличается способ задания геометрических форм. В САПР применяется граничное представление (BREP), для описания которого используются понятия типа «поверхность», «грань», «ребро», «кривая», и за которыми стоят соответствующие уравнения, описывающие эти объекты в пространстве. В GEANT/ROOT применяется конструктивная твердотельная геометрия (CSG), которая оперирует понятиями «примитив» и «Булева операция». Очевидно, что и за этими объектами также стоят конкретные уравнения, описывающие кривые и поверхности, однако есть существенное различие описанное ниже. Во-вторых, отличается способ задания взаимоотношения форм в пространстве. В САПР, по аналогии с тем, как человек воспринимает окружающий мир, присутствует некоторое бесконечное окружающее пространство без материала, а все предметы находятся в этом пространстве. Невозможна такая ситуация, чтобы один объект находился внутри другого — в таком случае подразумевается, что во втором есть соответствующая полость, освобождающая место под первый объект. В GEANT/ROOT для описания взаимоотношения форм используется иерархия объёмов. Это объясняется тем, что такой метод более удобен для описания геометрии, где главной задачей является однозначное задание материала в каждой точке пространства. Вводится понятие объёма — сущности, имеющей форму и материал. Из всех объёмов выбирается один, называемый объёмом верхнего уровня, а остальные помещены либо в него, либо в какой-то другой, формируя таким образом дерево объёмов.

2.1.1 Представление геометрии в САПР

В BREP есть два типа понятий — геометрические («точка», «кривая», «поверхность») и топологические («вершина», «ребро», «грань»). «Точка» — это тройка координат в некоторой системе координат. «Кривая» — это уравнение, задающее множество точек, принадлежащих данной кривой. Кривую удобно описать с помощью параметрического уравнения от одной переменной. «Поверхность» — это уравнение, задающее множество точек, принадлежащих данной поверхности. Соответственно, поверхность удобно описать с помощью параметрического уравнения от двух переменных. Топологические сущности задаются на базе геометрических. «Вершина» лежит в некоторой геометрической точке. «Ребро» лежит на некоторой геометрической кривой и ограничено двумя вершинами. Очевидно, что эти вершины должны принадлежать кривой, то есть и соответствующие геометрические точки должны принадлежать кривой. «Грань» лежит на некоторой поверхности и ограничена замкнутым циклом из рёбер. Также очевидно, что эти рёбра должны принадлежать поверхности, как и кривые, на которых они лежат, как и вершины и точки, ограничивающие эти рёбра. Замкнутая оболочка из граней с указанием внешних сторон этих граней ограничивает некоторую область пространства, называемую «телом».

В соответствии с BREP параллелипипед (которому эквивалентен примитив box в CSG) задаётся следующим образом.

Картинка и описание кратко

Стоит однако отметить, что человек, создающий геометрическую модель в САПР, хотя и может выполнять построения в соответствии с базовыми принципами BREP, чаще всего применяет интуитивно понятные формообразования, из которых система точно формирует BREP модель в памяти ЭВМ, которая также необходима для получения триангулированной геометрии для визуализации на дисплее ЭВМ. Есть 4 базовых формообразования и 4 им обратных (с вычитанием) — «выдавливание», «вращение», «протягивание» и «тело по сечениям». Многие другие формообразования, такие как фаски и скругления, разрезы, отверстия, внутри на самом деле являются лишь вариациями перечисленных. Последовательность формообразований, выполненных пользователем для получения итоговой формы, сохраняется в виде дерева построения модели, напоминающего историю построения, но позволяющего навигацию и редактирование. Дерево часто доступно пользователю в основном рабочем окне интерфейса САПР. Однако бывают случаи, когда история построения теряется, например при передаче модели из одной САПР в другую. Таким образом, в результате работы инженера получается модель, описанная с помощью BREP, и во многих случаях имеющая также и дерево построения.

В инженерной практике принято проектировать и соответственно строить 3d-модели, объединяя в сборки детали и другие сборки. Отсюда вытекает, что во многих САПР, в том числе в CATIA v5, существуют стандартные объекты, обозначающие детали и сборки. Например в САПР CATIA v5 существует отдельный тип документа CATPart для детали и отдельный тип документа CATProduct для сборки. Внутри документа типа CATPart есть минимальный набор обязательных элементов — 3 стандартные взаимноперпендикулярные плоскости в начале системы координат детали и главное тело детали, по умолчанию называемое PartBody. В документе типа CATProduct присутствует возможность добавлять в качестве дочерних компонентов либо документы CATPart либо другие документы CATProduct. В 2.2 описывается, как соотносятся перечисленные сущности CATIA v5 с понятиями геометрической подсистемы GEANT/ROOT.

Во многих САПР, в том числе и CATIA v5 присутствует возможность так называемого контекстного редактирования компонентов. Это означает, что пользователь во время работы над сборкой в документе типа CATProduct, имеющей в качестве дочерних компонентов детали в файлах типа CATPart, может также редактировать детали, не переключая активный документ. Эта возможность широко используется в “CATIA-GDML geometry builder” — большая часть работы выполняется в контексте единственного продукта, что с точки зрения пользователя аналогично работе над всей экспериментальной установкой.

2.1.2 Представление геометрии в GEANT/ROOT

Для описания геометрических форм в пакетах GEANT/ROOT применяется CSG. В качестве строительных блоков в CSG используются примитивы из списка реализованных в системе. Список примитивов включает в себя как относительно простые примитивы типа параллелипипед (box), сегмент

цилиндра (tubs), сегмент конуса (cons), так и достаточно сложные, типа эллипсоида, параболоида, скрученных (twisted) примитивов. Принимая во внимание тот факт, что геометрия в GEANT/ROOT нужна для выполнения моделирования взаимодействия частиц с материалом, можно сказать, что примитив — это объект, имеющий геометрическое представление и для которого реализовано решение геометрических задач, возникающих при моделировании. Среди таких геометрических задач можно отметить задачу нахождения расстояния до ближайшей границы примитива от некоторой точки внутри объёма, в одном заданном направлении или в любом возможном направлении. Эту задачу необходимо решать многократно в процессе проведения частицы для того, чтобы определить так называемый максимальный допустимый геометрический шаг. В результате моделирования физических процессов получается максимальный допустимый шаг из соображений физики. Для того чтобы собственно изменить координату частицы из этих двух шагов выбирается минимальный.

Форма может быть описана как результат Булевой операции над примитивами или другими Булевыми операциями. Есть три Булевы операции — объединение (union), вычитание (subtraction) и пересечение (intersection). Булевы операции позволяют задать практически любую геометрическую форму, имеющую границы из тех, что применяются в примитивах. При этом не требуется дополнительной реализации решения геометрических задач, т.к. удаётся комбинировать то, что реализовано в примитивах.

Таким образом наблюдается некоторая аналогия между BREP и CSG, заключающаяся в том, что в любом случае сложное тело или базовый примитив имеет некоторые границы, заданные аналитическими выражениями. Корни этой аналогии лежат в фундаментальной математике. Однако решающая разница заключается в том, что для примитива эти границы чётко определены и имеется лишь ограниченное число параметров, позволяющих изменять форму примитива.

Вторая составляющая геометрического представления в GEANT/ROOT это иерархия объёмов. Введём понятия логического и физического объёмов, формы и материала. Логический объём, или просто объём это базовый элемент для построения иерархии объёмов. Объём описывает непозиционированный объект и всё, что находится внутри него. Объём характеризуется формой и материалом. Форма — это заданные с помощью CSG границы пространства, по методу, описанному выше. Материал включает в себя описание химического состава, плотности, и т.д. При помещение одного логического объёма в другой, например объёма A в объём B, образуется так называемый физический объём, или узел, B_1, который обозначается взаимоотношение A и B как материнский-дочерний и характеризуется неоторой матрицей позиционирования B внутри A.

2.2 “CATIA-GDML geometry builder”

“CATIA-GDML geometry builder” (далее просто “Builder”) представляет собой набор документов-шаблонов и макропрограмм для САПР CATIA v5 вместе с настройками окружения и инструкциями к применению стандартных средств CATIA v5. “Builder” ставит своей задачей упростить процесс создания CSG моделей с иерархией объёмов, напрямую совместимых с GEANT/ROOT.

Центральная идея “Builder” заключается в правилах соответствия сущностей CATIA v5 и сущностей геометрии в GEANT/ROOT. Это соответствие делает возможным конвертацию МС-модели в CATIA v5 в любой внешний файл с целью дальнейшего импорта в ROOT/GEANT. В качестве формата для обмена был выбран XML-подобный формат GDML (geometry description markup language), разработанный в CERN, для которого в GEANT4 и ROOT реализованы методы импорта и экспорта.

Вся геометрия установки создаётся в одном документа типа CATProduct. Объёму соответствует деталь, хранящаяся в файле типа CATPart. Форме соответствует главное тело детали, по умолчанию называемое PartBody. В CATIA не записывается описание материала так, как это принято в GEANT/ROOT, а сохраняется только имя материала в пользовательском параметре Material. Это возможно по той причине, что существует практика хранить описание материалов во внешнем файле или базе данных, считывать его перед выполнением моделирования и приписывать объёмам в соответствии с именами. Для обозначения физического объёма B внутри A в структуре документа, описывающего объём A, создаются тела Body.A.*, где * по умолчанию обозначает номер вхождения, но допускается запись любой идентифицирующей строки.

Может быть частично перенести в описание методов описания геометрии в GEANT/ROOT. Также в “Builder” предусмотрена возможность задания геометрии некоторыми продвинутыми методами, специфичными для GEANT/ROOT. В GEANT/ROOT существует тип объёvов, называемый Asssembly, который характеризуется тем, что он не имеет формы и материала. Практически объём типа Assembly является контейнером без границ, который объединяет свои дочерние объёмы, что особенно удобно как минимум в двух случаях. Во-первых, если необходимо многократно позиционировать группу объёмов, которую невозможно охватить простой формой. Во-вторых, если преобразование координат при позиционировании одного или группы объёмов имеет сложную структуру и удобно представить его как суперпозицию двух преобразований. Как частный случай можно упомянуть ситауцию, когда какой-либо параметр преобразования является параметром модели (см. секцию 2.2.2)

Один из плюсов “Builder” заключается в том, что пользователю предоставляется возможность работать с полноценной инженерной моделью и МС-моделью в одной и той же среде, имеющей широкие возможности для анализа и редактирования геометрии. “Builder” не ставит своей задачей перевод модели из одного геометрического представления в другой, но значительно ускоряет процесс создания одной геометрии, на основе другой. На нашей практике это означает, что ??? забыл мысль

“Builder” включает в себя файлы, в которых специальным образом построены примитивы GEANT/ROOT, позволяющие пользователю при построении МС-модели не вникать в подробности реализации, а использовать их практически как и в процессе создания геометрии средствами ROOT или GEANT. “Builder” также включает в себя макропрограммы для CATIA v5, которые также ставят своей задачей сделать процесс построения геометрии в “Builder” максимально похожим на процесс построения геометрии в GEANT или ROOT. Основной макрос — это конвертер “CATIA2GDML”, который проецирует дерево построения модели в CATIA в GDML файл. Также разработан обратный конвертер “GDML2CATIA” для импорта GDML

файлов.

Целевая аудитория “CATIA-GDML geometry builder” — физики, владеющие CATIA v5 на базовом уровне, и инженеры, продвинутые пользователи САПР, изучившие способ представления геометии в GEANT/ROOT хотя бы на теоретическом уровне.

Есть опыт, который показывает, что для достижения такого уровня как физикам, так и инженерам, достаточно прохождения двухнедельного курса.

Предлагается новый алгоритм работы, в котором “Builder” используется как многофункциональный инструмент. Описанный ниже алгоритм сформулирован на успешном опыте разработки CBM RICH на протяжении 3 (4) лет.

Задача создания и поддержания актуальной МС-модели поручается ответственному человеку, владеющему CATIA, GEANT/ROOT и “CATIA-GDML geometry builder”.

В зависимости от того, какая информация и в каких файлах имеется к началу работы, алгоритм немного различается. Если разработка ведётся в нуля и нет никаких данных в ЭВМ, что возможно, например, когда проект находится на таком этапе, когда нужно выполнить грубое моделирование, показывающее принципиальную возможность реализации, то наиболее оптимальный способ — сразу строить МС-модель в CATIA средствами “Builder”. Если, скажем, проект находится на раннем этапе разработки и уже имеется какая-то приблизительная САПР модель, то рекомендуется импортировать её стандартными средствами CATIA, чтобы затем на её основе построить МС-модель в CATIA в автоматизированном режиме с помощью средств “Builder”. Инженерную геометрию можно импортировать практически из любой САПР, например с помощью широко распространённого формата STEP. Третий распространённый случай это когда уже имеется некоторая МС-модель в конечной системе моделирования. Как в GEANT4, так и в ROOT имеется стандартная возможность экспортировать геометрию в GDML файл без потери информации. Эту возможность могут наследовать все дочерние пакеты (как FairRoot и далее CbmRoot), но для этого необходимо явно активировать функциональность GDML. В этом случае можно импортировать модель в CATIA в МС-формате, однако иногда требуются некоторые дополнительные ручные операции после импорта. Они выполняются однократно и лишь делают структуру документа более оптимальной, но не изменяют геометрию.

Во всех этих алгоритмах, независимо от типа и количества исходных данных, получаются файлы CATIA в формате “Builder”, которые в дальнейшем будут являться основными (первичными) файлами для получения рабочей МС-модели в экспериментальном пакете, которым в случае CBM RICH является CbmRoot. Модель из CATIA экспортируется в GDML файл, который не требует каких-либо последующих изменений в структуре. Для достижения этого условия была проведена огромная работа по мере разработки МС-модели CBM RICH. Допускается и даже рекомендуется текстовое редактирование GDML файла, но только для изменения значений параметров в define секции у параметризованных моделей. Затем, по желанию коллаборации, GDML файл может быть конвертирован в бинарный ROOT-файл, который содержит геометрию, которую невозможно редактировать. Это защищает модель от случайных изменений, что особенно актуально

в случае параметризованных моделей. Соответственно, если требуется изменить значения параметров, пользователь может отредактировать GDML файл и экспортировать в новый ROOT файл. Практика показывает, что в случае, если требуется множество файлов с МС-геометрией, то обязательно нужно писать комментарии — либо в самом GDML файле, либо в текстовом файле рядом с GDML/ROOT файлом. Обычно в коллаборации вводят правила именования файлов.

2.2.1 Примитивы в “Builder”

2.2.2 Макропрограммы для CATIA v5

Макропрограммы для CATIA v5 написаны на VBA с применением CATIA API. Все макропрограммы, кроме «AddShape» и «Poly», доступны пользователю в режиме работы над сборкой. В CATIA различают открытый документ (верхний в дереве в текущем окне), активный документ (синий), выделенный объект (оранжевый) и рабочий объект (подчёркнутый). Пользователь может выполнить все операции, необходимые для получения МС-модели, самостоятельно без применения макропрограмм, но в этом случае велика вероятность упустить какой-либо шаг, что приведёт к ошибке, которую сложно диагностировать.

В МС-модели в CATIA есть строгие правила именования. Применение макросов избавляет пользователя от необходимости контролировать имена объектов в документах. Все имена, сгенерированные при использовании “Buider” не конфликтуют между собой и позволяют получить корректный GDML файл на выходе. Практически везде пользователь имеет право изменять суффиксы, не изменяя основного названия, несущего информацию о типе объекта — формообразования, тела, и т.д. Однако в редких случаях суффикс имеет решающее значение, как например в именах поворотах (напр, “Rotate.X”) суффикс несёт информацию о оси поворота.

В процессе разработки “Builder” был выработан стандартный алгоритм создания геометрии. Первый шаг — создание нового документа типа CATProduct, который в дальнейшем будет единственным продуктом, и его сохранение на диск. Этот продукт будет представлять модель всей экспериментальной установки. Второй этап — наполнение продукта описанием объёмов без описания взаимосвязей между ними. Для этого используется макрос «AddNewPart», который автоматически открывает в отдельном окне новый документ типа CATPart, сформированный из специального шаблона и соответствующий создаваемому объёму. Система переходит в режим редактирования детали, где доступны только два макроса «AddShape» и «Poly» для создания формы объёма. Здесь же можно и задать имя материала объёма. По окончании редактирования нового объёма в отдельном окне пользователь должен сохранить активный документ и закрыть это окно. CATIA при этом возвращается к редактированию продукта. После того, как созданы объёмы, заданы формы и, возможно, имена материалов, алгоритм подразумевает задание иерархии объёмов, то есть позиционирование одних объёмов в других. Для этого в “Builder” существует целый ряд макропрограмм для создания различных типов взаимосвязей — «Inserter», «ArrayMaker», «Replica». После того, как выполнено размещение дочернего объёма A в материнском объёме B, пользователь может указать пово-

рот и сдвиг, задающие матрицу позиционирования A в B. Для упрощения расчётов в некоторых случаях очень удобно применять макропрограммы «PointToPointAligner» («Pt2PtAligner»), «Mover» и «Measure». Для удобного редактирования материалов всех объёмов был разработан менеджер материалов «MaterialsManager», который обычно имеет смысл вызывать перед экспортом для проверки ранее заданных имён материалов, либо назначения новых. Также перед экспортом рекомендуется проверить модель на наличие ошибок с помощью макроса «Checker». В конце выполняется экспорт макросом «CATIA2GDML». Отдельно стоят макропрограммы «Duplicator» для создания множественных идентичных, но не связанных, параметризованных подсборок и обратный конвертер «GDML2CATIA» для импорта GDML файла.

Для комфортной работы с “Builder” в поставке также имеются файлы для настройки окружения CATIA. Использования окружения в принципе не обязательно, но часть функционала зависит от путей к файлам, которые прописаны в переменных окружения, поэтому настоятельно рекомендуется перед использованием “Builder” выполнить настройку, следуя инструкции, поставляемой в пакете.

AddNewPart

Данная макропрограмма автоматизирует создание нового документа типа CATPart на основе шаблона, содержащего необходимые элементы — публикация главного тела детали, называемая PartBody, пользовательский параметр под названием Material со значением по умолчанию ???. Также для удобства погашены стандартные плоскости.

AddShape

«AddShape» используется для создания примитивов, в случае необходимости вместе с поворотами и сдвигом. Макропрограмма играет роль интерфейса между пользователем и файлами примитивов. При запуске макроса выводится окно со списком доступных примитивов, по нажатии на кнопку “создать” в рабочее тело детали вставляется выбранный примитив со значениями параметров по умолчанию. Если на форме графического интерфейса выбраны флаги создания поворотов и сдвига, то создаются соответствующие формообразования.

Poly

В силу ограничений CATIA нет возможности представить полипримитивы (polycone и polyhedra) с помощью тех же средств, что и остальные примитивы, поэтому для них была разработана специальная структура дерева и правила именования. Для автоматизации построения полипримитивов в соответствии с этой структурой предоставляется макрос «Poly». Секции поликонуса представлены стандартными конусами. В случае polyhedra для представления секции используется hedra — специальный примитив, не поддерживаемый GEANT/ROOT.

Inserter

Макрос «Inserter» — это инструмент для помещения одного выбранного объёма в другой. Также можно сказать, что «Inserter» создаёт физический объём, задающий связь материнский-дочерний между двумя существующими логическими объёмами. «Inserter» — возможно, самый используемый макрос, в результате работы которого в документе типа CATPart, представляющем материнский объём, создётся тело с именем "Body.B.*", где B — имя дочеhнего объёма. Внутри этого тела имеется ссылка на публикацию PartBody документа типа CATPart, представляющего объём B, и элементы преобразования типа Rotate и Translate — три поворота и сдвиг, задающие матрицу позиционирования B внутри A.

ArrayMaker

Replica

PointToPointAligner

Mover

Measure

MaterialsManager

Приложение «MaterialsManager» предоставляет пользователю возможность изменять материалы отдельных объёмов, находясь на уровне виртуальной сборки. Это избавляет от необходимости часто переключаться между документами либо режимами работы CATIA при контекстном редактировании. Также заметным преимуществом использования «MaterialsManager» является наглядность — информация о материалах всех объемов представляется в компактном списке, присутствует возможность быстро изменять значения в нескольких элементах списка. Помимо этого, наличие «MaterialsManager» позволяет отложить работу с материалами на последний этап. Использование шаблона файла детали предотвращает от того, что пользователь вообще не укажет материал объема — по умолчанию указан вакуум.

Checker

Существует необходимость проверять правильность построенной пользователем МС-модели в CATIA перед тем как выполнять экспорт в GDML. Она возникает в силу того, что в разработанной структуре документов CATIA для МС-модели введено множество правил и ограничений, нетипичных для conventional использования системы. «Checker» выполняет 2 типа проверок. Первый — корректность с точки зрения конвертера, т.е. соблюдение структуры документов, правильность именования, второй — корректность с точки зрения правил построения геометрии в GEANT/ROOT.

Использование разработанных интерактивных приложений ограждает пользователя от ошибок именования в итоговой МС-модели. Есть только одно место, где необходимо вручную указывать имя — формообразование-вращение при позиционировании операнда на уровне формы (?уточнить?).

Корректность геометрии определяется по двум критериям:

1. любые два объёма, находящиеся на одном уровне, не должны пересекаться;

2. любой дочерний объём не должен выходить за пределы материнского объёма.

Чтобы организовать проверку указанных условий, разработан специальный алгоритм и реализован в виде отдельной макропрограммы «Checker» САПР CATIA. В цикле перебираются все пары объёмов, лежащих на одном уровне, и проверяется, не пусто ли множество пересечения текущей пары. В том случае, если не пусто, то возникает событие, оповещающее, что текущая пара объёмов расположена недопустимым образом.

Более детально описанный процесс выглядит следующим образом. В силу того, что физические объёмы описываются телами детали, перебор объёмов, лежащих на одном уровне, сводится к перебору всех тел детали, кроме PartBody. Чтобы исследовать множество пересечения пары тел, создаётся новое пустое тело и копии исходных. Затем применяется булева операция Intersect над копиями, и результат заносится в ранее созданное пустое тело. На этапе выполнения формообразования булевой операции выдаётся возможность отследить корректность результата. С точки зрения CATIA пустое пересечение является ошибочным результатом и возникает внутренняя ошибка. Именно программный отлов и обработка этой ошибки говорит о корректности расположения объёмов. После выполнения булевой операции результирующее тело удаляется, не оставляя таким образом никаких следов промежуточных преобразований.

Для того чтобы отследить, не выходит ли объём за пределы материнского объёма, применяется схожий подход. Отличие заключается в последовательности булевых операций — вместо пересечения $A * B$ двух объёмов A и B проверяется объём, полученный последовательностью двух операций $(A + B) - A$, где A — материнский объём, а B — дочерний. Присутствие результата операции $(A + B) - A$ говорит о том, что какая-либо часть дочернего объёма расположена за пределами материнского.

CATIA2GDML

Duplicator

GDML2CATIA

«GDML2CATIA» выполняет процедуру, обратную «CATIA2GDML» — проецирует GDML файл на дерево модели CATIA v5. Отличительной особенностью является то, что в “Builder” есть возможность задавать линейные и круговые массивы — многократные вхождения дочернего объёма в материнский, позиционированные с некоторым шагом вдоль линейной или круговой оси соответственно. В CAITA для массивов применяется соответствующее стандартное формообразование pattern. Такая возможность отсутствует в GDML, поэтому при экспорте из CATIA в GDML выполняется расчёт поворотов и сдвигов для каждого элемента массива и они представляются как отдельные, независимые дочерние объёмы. Таким образом при конвертации в обратном направлении, из GDML в CATIA, невозможно восстановить массив. Следовательно, одна из немногих (единственная) операций, которые необходимо совершать после импорта геометрии из GDML

в CATIA — ручной перевод множества дочерних объёмов в массив. Обычно это очень простая процедура,

Параметризация

Одна из наиболее важных возможностей “CATIA-GDML geometry builder” — это возможность создания параметризованных геометрических МС-моделей. У параметризованной модели имеются входные параметры и формулы, задающие зависимость между этими входными параметрами и внутренними переменными, такими как параметры примитивов, значения поворотов и смещений. Данная концепция хорошо ложится на методы работы с геометрией в САПР, особенно CATIA v5. Также параметризация поддерживается форматом GDML и импортёрами GEANT(?) и ROOT.

В модели CATIA v5 можно вводить пользовательские параметры как в документах типа CATProduct, так и в документах типа CATPart. Причём сборка в CATProduct файле может иметь свои пользовательские параметры и формулы, а дочерние компоненты в CATPart файлах — свои. Обязательное требование “Builder” таково, что все параметры и формулы должны находиться в верхнем продукте. CATIA v5 позволяет задавать зависимости между любыми параметрами, в том числе внутренними, не являющимися пользовательскими, однако для успешного экспорта в GDML файл формула должна в левой части иметь параметр примитива или угол поворота или значение сдвига, а в парвой части — формулу только над пользовательскими параметрами. Пользовательский параметр CATIA v5, экспортируемый в переменную в GDML должен обязательно иметь безразмерный тип Real. В связи с этим имеются правила оформления формул и приведения единиц измерения. Также имеется стандартная переменная DEGtoRAD для перевода значения улгов из градусов в радианы.

На выходе получается GDML файл, у которого в define секции есть тэги variable, обозначающие входные параметры модели со значениями. При импорте параметризованной геометрии из GDML в ROOT все значения внутренних переменных рассчитываются в соответствии с формулами по значениям входных параметров и параметризация теряется. Следовательно значения входных параметров должны задаваться пользователем непосредственно в GDML файле перед импортом в конечную систему.

2.2.3 Избранные подробности реализации “CATIA-GDML geometry builder”

Каждый макрос “Builder” — это VBA проект, который хранится в отдельном catvba файле. Проект состоит из трёх разделов — элементы графического интерфейса (формы), модули и модули классов. Большинство макросов “Builder” написано в соответствии с идеологией структурного программирования, без применения классов, и разделение на модули выполнено из соображений читаемости кода. Обычно в отдельный модуль выносился функционал, объединённый некоторой задачей. Так, например, во многих макросах имеется модуль ??? (имя) для продвинутой работы со строками, модуль ??? (имя) для ??? (задача). В некоторых случаях естественным образом требовалось использовать классы. Так, например, был реализован

класс матрицы с методами нахождения углов поворота, который использовался в ??? и более подробно описан в 2.2.3.

Работа с матрицами позиционирования в "CATIA-GDML geometry builder"

2.3 Применение "CATIA-GDML geometry builder" к CBM RICH

Значительная часть работы над "Builder" выполнялась при поддержке группы CBM RICH, поэтому самая сложная MC-модель построенная с помощью "Builder" это CBM RICH. Построенная за несколько итераций модель имеет достаточно сложную иерархию и характеризуется высокой степенью подробностей.

Для того, чтобы GDML без проблем импортировался в CbmRoot было написано дополнение.

Глава 3

Результаты

3.1 Время-цифровой преобразователь

На рисунке 3.1 приведена условная схема функционирования одного канала ВЦП, дающая представление о причинах сдвига калибровочной таблицы и объясняющая минус в формуле расчёта полного времени.

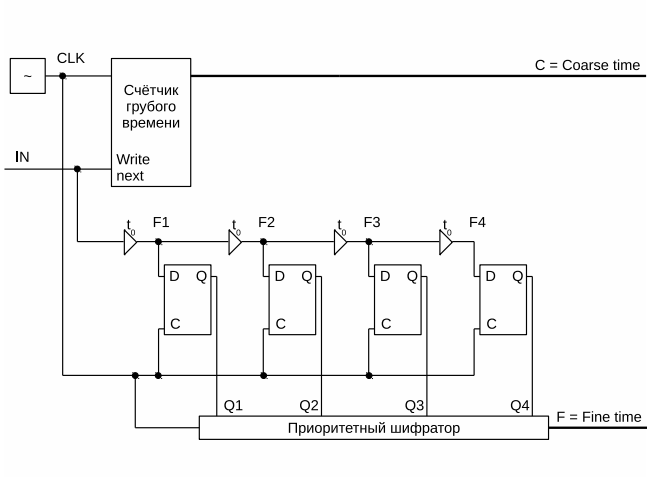

Рис. 3.1: Структурная схема одного канала ВЦП.

Имеется тактовый генератор частотой 200 МГц. Период такого генератора — 5 нс. Он управляет счётчиком грубого времени. Каждые 5 нс значение грубого времени увеличивается, но не выдаётся на выход. Счётчик точного времени выполнен по технологии Tapped delay line (TDL) — цифровая линия задержки (DDL) с промежуточными выходами. Используются элементы задержки t_0, имеющие одинаковые характеристики в пределах некоторой точности. Количество элементов должно быть таким, чтобы полностью заполнить период между двумя отсчётами грубого времени. Регистрируемый фронт, поступающий на вход IN, проходит линию задержки, состоящую из нескольких элементов задержки. По мере прохождения ли-

нии фронтом триггеры переключаются из 0 в 1, каждый следующий через промежуток времени, равный t_0. При поступлении следующего фронта от тактового генератора происходит считывание грубого времени и перенос значений выходов триггеров в приоритетный шифратор.

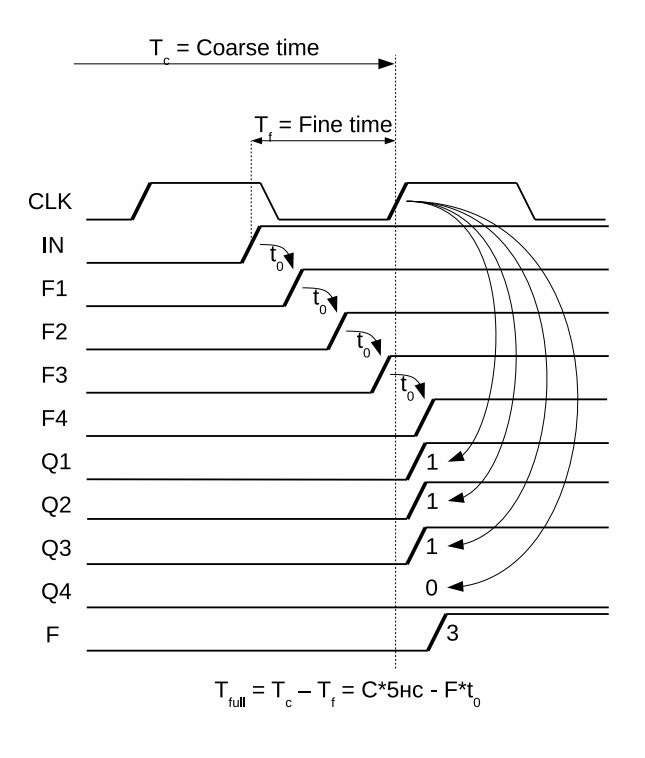

Рис. 3.2: Пример временных диаграмм при регистрации входного фронта.

Отличительная особенность такого шифратора — индиффирентность к значению на входах $j < i$ при наличии логической единицы на входе i. Иными словами имеет значение только старший бит, а младшие игнорируются. (У обычного шифратора только один вход должен иметь единицу на входе). Шифратор преобразует номер последнего переключившегося триггера в число, обозначающее значение точного времени. Таким образом точное время должно вычитаться из грубого времени потому что линия задержки измерила время между моментом прихода входного сигнала (start) и моментом прихода следующего отсчёта грубого времени (stop).

Приведена наиболее понятная схема, фактическая же реализация отличается. Например, в качестве элемента задержки может выступать сам триггер. Тогда выход i-го триггера напрямую соединяется со входом (i+1)-го триггера. Используемые нами ВЦП в ППВМ имеют в качестве элемента задержки ячейку матрицы, запрограммированную как полный сумматор.

ВЦП разрабатывался так, чтобы интервал 5 нс между двумя отсчётами грубого счётчка разбивался на 512 элементов. Тогда было бы достаточно 9-битного шифратора для формирования значения точного времени. Из-за того, что существует также и задержка сигналов в проводниках ненулевой

длины, реально таблица может быть сдвинута. Например значение точного времени 0 должно означать, что входной фронт пришёл одновременно с фронтом от тактового генератора (точнее совсем чуть-чуть раньше). Если учитывать задержку в проводниках на приведённой условной схеме, то получится, что команда считывания шифратору идёт дольше, чем до первого триггера. В таком случае никогда не будет принято нулевого значения и 9 бит для хранения точного времени будет недостаточно. По данной причине сообщение, несущее точное время, имеет длину 10 бит, а все калибровочные таблицы имеют правую границу на значении 1024.

Пример таблицы калибровки точного времени представлен в виде графика на рисунке 3.3. По оси абсцисс откладывается значение счётчика точного времени, а по оси ординат — значение точного времени в наносекундах.

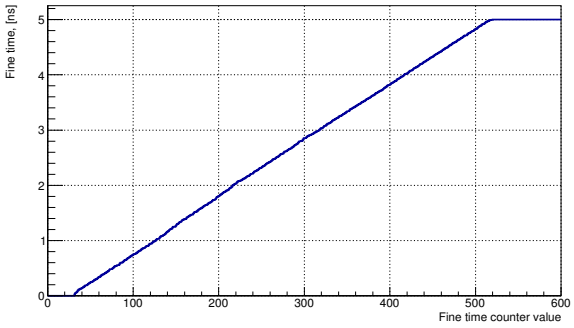

Рис. 3.3: Пример таблицы калибровки точного времени.

3.2 Обработка сигнала с детектора

Задача: дать общую картину — детектор, передняя электроника, передающая электроника, ЭВМ. Затем указать, что вот типа электроника с внешним триггером, а вот самотриггирующаяся электроника. Далее переходим к free-running DAQ.

Рассмотрим общую последовательноть обработки сигнала детектора частиц. Источником является некоторое детектирующее устройство (или просто детектор). В зависимости от типа детектируемых частиц и внешних условий — радиационная среда, температура, частота регистрации — это может быть фотодиод, фотоэлектронный умножитель (ФЭУ), полупроводниковый детектор, микроканальная пластина, газовый электронный умножитель, и т.д. Все эти детекторы обладают таким общим свойством, что выходной сигнал, несущий информацию о зарегистрированной частице, это относительно слабый токовый импульс. Для того, чтобы выполнять дальнейшую обработку этого сигнала в ЭВМ его необходимо оцифровать. В связи с этим обрабатывающую электронику условно можно разделить на переднюю и остальную. Основная задача передней электроники — оцифровать выходной аналоговый сигнал детектирующего устройства, при необходимости предварительно его усилив и отфильтровав. Передача аналогового сигнала затруднена, поэтому его обработка и оцифровка обычно выполняется как можно ближе к месту, где этот сигнал выбрабатывается, т.е.

непосредственно на детекторе. В простом случае последующие слои электроники концентрируют и передают оцифрованные сигналы со многих каналов передней электроники. В более сложном случае возможна какая-то аппаратная обработка оцифрованных сигналов, например архивация или фильтрация с целью уменьшения потока данных.

Разработан целый ряд подходов к оцифровке сигналов с детекторов. Каждый из них лучше всего подходит к определённому детектору, однако возможно применение методов, разработанных для одного детектора, для оцифровки сигналов с другого детектора. Перечислим лишь некоторые из них. Если не вдаваться в подробности реализации, можно выделить следующие способы: регистрация момента времени прихода фронта, регистрация амплитуды сигнала, непрерывное семплирование (ещё?). Практически всегда требуется регистрация момента времени прихода сигнала, поэтому широко применяются комбинации перечисленных способов — регистрация амплитуды сигнала вместе с моментом времени прихода переднего фронта, регистрация моментов времени переднего и заднего фронтов без захвата амплитуды, регистрация момента времени переднего фронта и формирование ограниченного числа семплов после него.

Если необходимо предварительное усиление, то применяют один из двух типов усилителей — зарядочувствительный усилитель либо ???. В силу устройства зарядочувствительный усилитель является также формирователем (шейпером). Любая электроника подвержена шумам, поэтому для подавления шумов сигнал обычно предварительно проходит фильтр нижних частот, эффективно отрезающий частоты выше некоторого значения. (Что-то сказать про шейпирование и его неизбежность и влияние на временные характеристики) Ещё один способ борьбы с шумами — фильтрация по амплитуде, т.е. установление некоторого порога по напряжению, ниже которого сигнал игнорируется.

3.2.1 Бестриггерная передняя электроника

Рассмотрим схему, когда аналоговый импульс с детектора обрабатывается электроникой, регистрирущей момент времени прихода переднего фронта. Любой сигнал переключается за некоторое ненулевое время, поэтому необходимо определить точку, обозначающую фронт. Для этого применяют дискриминатор — прибор, вырабатывающий логический “0”, когда входной сигнал ниже установленного порога, и логическую “1”, когда входной сигнал выше установленного порога. Таким образом точка пересечения сигнала и порога это точка, условно обозначающая момент времени прихода сигнала. На выходе дискриминатора получается логический сигнал, который необходимо преобразовать в цифровой с помощью время-цифрового преобразователя. На выходе ВЦП уже будет цифровой сигнал, а не логический. Далее могут работать концентраторы данных и какие-то ещё платы, обеспечивающие приём данных в ЭВМ. Таким принимающим устройством может быть обычный сетевой интерфейс, работающий с Ethernet. В СВМ это не так, поэтому нужен FLIB.

3.2.2

Рассмотрим систему считывания и сбора данных “традиционного” эксперимента.

(Тут нужно обсудить и почитать ещё. Возможен ли такой сценарий, в наше время или в прошлом, когда триггер чисто аналоговый. То есть передняя электроника игнорирует сигнал до тех пор, пока не сработает схема совпадения с триггером. В той модели традиционного триггера, которую я себе сейчас представляю, входной сигнал обрабатывается непрерывно, но выпускается на выход из буфера только при наличии триггера. Вероятно, возможно и то и то, но вопрос в том, что реально используется, что более распространено.)

Каждый канал передней электроники имеет выходной буфер, куда по принципу FIFO складываются оцифрованные входные сигналы. В экспериментальной установке присутствуют дететоры, вырабатывающие триггер — сигнал, который заводится (условно) на каждый канал считывания, и говорит о том, что произошло интересное событие, которое необходимо сохранить для последующей обработки. Данный подход имеет свои причины. Во-первых, до недавнего времени физические эксперименты не требовали высоких частот регистрации — выполнение физической программы при относительно низких частотах первичного взаимодействия было осуществимо в разумные сроки. Во-вторых, многие регистрирующие приборы имеют заметное “мёртвое время” — время, в течение которого прибор не может обрабатывать входные сигналы. Следовательно, если канал регистрирует ложный входной сигнал, велика вероятности того, что будет пропущен полезный сигнал.

С развитием электроники “мёртвое время” уменьшалось. Более того, возможность применения принципиально другой считывающей электроники, как например чисто временной канал, реализованный в ППВМ, для обработки сигналов с МА ФЭУ в CBM RICH, исследуемая в данной работе, позволяет на порядки снизить “мёртвое время” и повысить точность регистрации временной отметки в ущерб полноты информации.

(Сюда подмешивается секция 2.2)

В CBM планируется использование программного триггера. Это означает, что для того, чтобы принять решение, сохранять принятые данные или нет, необходимо выполнить полную реконструкцию события, включая реконструкцию треков, которая является высоко-затратной задачей. Рассматривается также возможность на определённых этапах работы установки использовать для выработки триггера частичную реконструкцию. Например исследуется возможность триггирования по результатам реконструкций треков только в MUCH, когда стоит задача поиска (такой-то частицы).

В “традиционном” эксперименте триггер может формроваться в результате логических операций над сигналами с нескольких детекторов, реализованных аппаратно. Такая логика работает за (масштаб времени). Реконструкция треков выполняется за гораздо большее время (на столько-то порядков выше). По этой причине необходимо иметь не только буфер в электронике — его будет недостаточно.

(Сказать о том, что нужно хорошо настраивать пороги.)

3.2.3 FLES

Всё вместе привело к разработке аппаратно-программного комплекса, называемого системой отбора первого уровня — First Level Event Selector (FLES). Электроника бестриггерная, несколько уровней концентрации данных, многочисленные буферы, формирование срезов времени, построение интервалов и только после этого мы переходим к построению событий.

Для эксперимента CBM был выполнен оценочный расчёт. Отправная точка — возможно сохранение 1 Гбайт/сек данных. Считается, что одно событие CBM в среднем имеет объём 40 Кбайт. Отсюда следует, что максимальная частота первичного взаимодействия может быть 25 кГц. В стартовой конфигурации CBM частота первичного взаимодействия равна 10 МГц, следовательно необходимо уменьшить поток данных в 400 раз. В полноценном режиме работы CBM ожидается 25 МГц, т.е. $25 \cdot 10^6 \cdot 40$ Кбайт = 1 Тбайт/сек. Планируется разбить этот поток в 1 Тбайт/сек на 1000 входных каналов FLES, каждый по 1 Гбайт/сек, передающихся по 10-Гбитным оптическим каналам связи. Один входной канал FLES соответствует одному "входному узлу" (input node, IN) — ЭВМ с установленной платой FLIB. Все вычисления, необходимые для отбора данных будут осуществляться на так называемых "вычислительных узлах" (computing node, CN). Входные и вычислительные узлы объединены в компьютерную сеть посредством InfiniBand QDR, образуя уникальную распределённую вычислительную систему, называемую FLES. Вычислительная подсеть будет иметь приблизительно 60000 ядер.

Планируется также, что FLES сможет функционировать в особом режиме, когда для выполнения реконструкции с целью отбора данных для сохранения будет использоваться только часть входного потока. Это представляется возможным при работе эксперимента над некоторыми пунктами физической программы. Например, можно восстанавливать такую-то частицу только по трекам в MUCH. При этом система приёма работает в полную силу — идёт приём со всех детекторов и никакие данные не выбрасываются до тех пор, пока не будет выполнена реконструкция по данным с MUCH. Если в результате реконструкции выясняется, что принятая порция данных потенциально интересна, то она извлекается из буферов и записывается. Это позволяется снизить поток сохраняемых данных в ??? раз, что особенно актуально при экстремально высоких частотах взаимодействия.

(Картинка, где показана схема для частичного триггера, вроде бы от Вальтера)

3.2.4 Построение события

Тот факт, что физическая программа эксперимента CBM подразумевает исследование очень редких явлений, для которых практически невозможно вырабатывать аппаратный триггер, привёл к решению разработать и использовать бестриггерную систему считывания. В бестриггерной системе считывания каждый канал передней электроники вырабатывает сообщение при преодолении входным аналоговым сигналом установленного порога. Получается, что электроника выдаёт для программного обеспечения непрерывный поток никак не сгруппированных сообщений, содержащих временную отметку. Для того, чтобы выполнять физический анализ, необходимо

в этом непрерывном потоке выделять осмысленные группы, которые мы называем событиями. Строго говоря, задача построения событий — это одномерная задача кластеризации на оси времени с последующим отбором кластеров по некоторым критериям.

Задача также усложнена тем фактом, что электроника не может обеспечить непрерывный поток сообщений, упорядоченных по времени регистрации. Происходит группировка сообщений в так называемые DAQ-события, которые необходимы для обеспечения передачи информации, а сообщения внутри DAQ-событий могут быть упорядочны произвольно. Соответственно первый этап построения события — упорядочивание сообщений.

В данных с пучковых тестов 2014 г., для того, чтобы определить, является ли распознанная группа событием, можно использовать сигналы с детекторов пучка — пороговых черенковских счётчиков, годоскопов и др. В лабораторных данных, где выполнялись измерения с лазером, в качестве триггера можно использовать сигнал от генератора, управляющего лазером. В ситуации, когда нет дополнительной информации, как в случае полного детектора RICH в итоговом эксперименте, необходимо принимать решение о том, является распознанная группа событием, или нет, на основе исключительно информации, полученной из этой группы. Распознанный кластер может являться событием, но чаще всего будет состоять из одного сообщения — шумового хита. Следовательно можно использовать кол-во хитов в событии для подавления шумов, что особенно актуально для дететора CBM RICH, где выполняется реконструкция черенковских колец, требующая некоторого минимального числа хитов в плоскости реконструкции.

(Сначала общая идея, что в любом случае будет некоторое временное окно и все сообщения попадающие в окно формируют событие. Размеры окна — один из параметров построителя событий, которым можно играть с целью повышения эффективности.)

(А здесь можно описать предлагаемый алгоритм)

3.2.5 LeadingEdgeDiff

Один из этапов обработки данных — построение событий. В данной работе рассматривается два типа событий — сигналы от лазера и сигналы от черенковского кольца. В любом случае, событие — это структура данных, содержащая информацию о хитах, сгруппированных по времени. Каждый хит содержит, как минимум, временную отметку момента прихода переднего фронта сигнала и номер канала, который в случае CBM RICH указывает номер пикселя фоточувствительной камеры, т.е. говорит о геометрическом положений зарегистрированного фотона.

Данное исследование посвящено, в первую очередь, временным характеристикам системы считывания, поэтому в основном речь пойдёт о временных отметках.

Очевидно, что для каждого события можно построить несколько распределений, которые на большом массиве данных, т.е. на многих событиях, характеризуют систему считывания и могут быть использованы для калибровки электроники с целью повышения временного разрешения системы. Т.к. событие имеет максимальную ширину, определяемую размерами окна

в алгоритме построения событий, распределения могут иметь “обрезанные хвосты”, которые, однако, невозможно избежать.

Пусть событие содержит N хитов. Введём внутри события нумерацию хитов от 0 до N. Пусть внутри события хиты упорядочены по времени, т.е. хит с временной отметкой t_0 был зарегистрирован раньше остальных, а хит с временной отметкой t_N — позже всех. Такой порядок может, например, обеспечиваться естественным образом алгоритмом построения событий. Внутри события все временные отметки зарегистрированы в разных каналах — множественные хиты в одном канале в одном событий являются признаком того, что порог дискриминатора установлен слишком низко и регистрируются шумы. Введём в рассмотрение распределение ω разностей временных отметок всех хитов, кроме первого, относительно первого, т.е. распределение

$$t_j - t_0, \text{ где } j \in [1..N].$$

Также введём распределение σ_1 всех пар временных отметок одного события, т.е.

$$t_j - t_i, \text{ где } i \in [0..N], j \in [0..N], i \neq j.$$

Очевидно, что в такой формулировке одна и та же пара временных отметок войдёт в распределение дважды с разными знаками — например, $t_1 - t_2$ и $t_2 - t_1 = -(t_1 - t_2)$. Это делает распределение симметричным, среднее значение строго равно 0, а ширина распределения чуть больше, чем в случае, когда нет дублирования информации. Введём непрерывную нумерацию каналов и примем, что в разности $t_j - t_i$ первая временная отметка была зарегистрирована каналом a, а вторая — каналом b. Введём распределение σ_2, по сути очень похожее на σ_1, но без дублирования информации, в котором будем учитывать только пары, у которых $b > a$.

В идеальной ситуации, если событие соответствует одной вспышке лазера или одному черенковскому кольцу, и отсутствуют факторы, размывающие время регистрации, все разницы были бы равны нулю. В качестве таких размывающих факторов можно привести, например, следующие: временные характеристики лазера, разброс геометрических путей черенковских фотонов, разброс времени прохождения электронной лавины в динодной системе ФЭУ, дребезг сигналов в передней электронике. Из-за перечисленных явлений распределение ω имеет следующую форму — (описание). Распределение σ_2 — (описание).

Среднее значение либо положение масимума распределения σ_2 можно использовать для того, чтобы определить значение поправки для данной пары каналов. Если выполнить анализ с применением коррекций, то вид всех распределений изменится. ω сгруппируется ближе к нулю, σ_2 передвинется к нулю, а σ_1 сузится к нулю.

Представляется возможность анализировать различные области фоточувствительной камеры. Интересно группировать хиты в соответствии с тем, какой электроникой они обрабатываются. В данном анализе было введено 4 подмножества: 1 пара каналов, 16 каналов одной платы передней электроники, 64 канала одного МА ФЭУ, 256 каналов 4 МА ФЭУ, образующих площадку 2x2 МА ФЭУ в одном углу камеры. При том, что вся фоточувствительная камера на пучковых тестах имела размер 4x4 МА ФЭУ, рассматривать более 4 МА ФЭУ одновременно не имеет смысла, т.к. в прототипе были установлены различные модели МА ФЭУ, некоторые покрытые сместителем спектра, а некоторые нет.

3.3 Исследование спектросместителя с помощью флюориметра

Были проведены независимые флюорометрические измерения пара-терфениловой плёнки, нанесённой по той же технологиий (dip-coating), что применяется для напыления сместителя спектра на поверхность МА ФЭУ в CBM RICH. Измеренный временной профиль приведён на рисунке 3.4, а результаты фитирования — в таблице 3.1.

Таблица 3.1: Результаты фитирования флюорометрических измерений сместителя спектра.

τ, нс	амплитуда	амплитуда, нормированная по 2-й компоненте
1.4	1387	5.15613
3.8	269	1.00000
45.0	19	0.07063

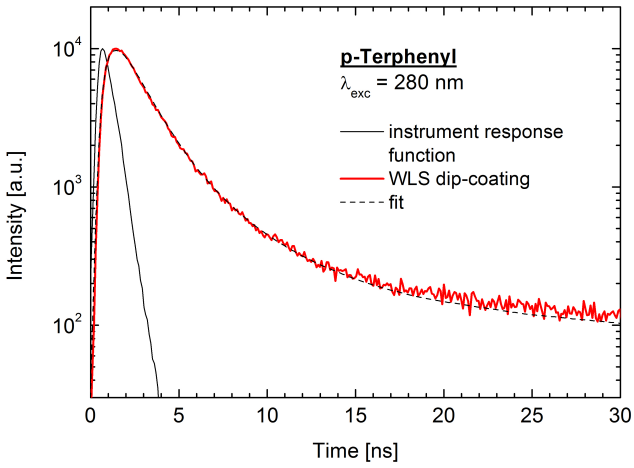

Рис. 3.4:

Данный фит неплохо воспроизводится функцией

$$f(t) = 10 \cdot (A_1 \cdot e^{-(t-1)/\tau_1} + A_2 \cdot e^{-(t-1)/\tau_2} + A_3 \cdot e^{-(t-1)/\tau_3})$$

со значениями амплитуд и времён, приведёнными в таблице 3.1. Здесь единица в скобке возле переменной t означает сдвиг графика по горизонтали на 1 нс и обусловлена разрешением измерительного прибора.

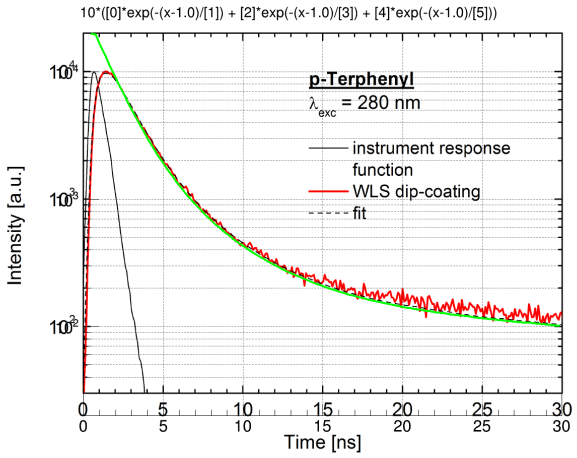

Рис. 3.5:

3.4 Прямые измерения временного профиля спектросместителя

Можно фитировать распределение WLS_on или WLS_off 4-мя компонентами, а можно фитировать их разницу WLS_diff трёмя компонентами. Можно фитировать функцией с зафиксированными временами, чтобы определить амплитуды, а можно фитировать функцией, где параметрами являются и времена и амплитуды.

3.4.1 Прямые фотоны

В результате анализа экспериментальных данных были получены две гистограммы — с и без сместителя спектра. Первый этап — фитирование профиля без сместителя спектра. В результате фитирования одной экспонентой в различных диапазонах и с различными начальными условиями было получено значение временной постоянной 0.65 нс.

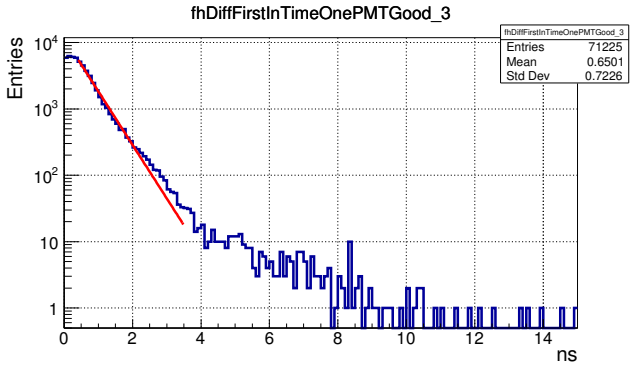

Рис. 3.6:

3.4.2 Фитирование WLS_diff

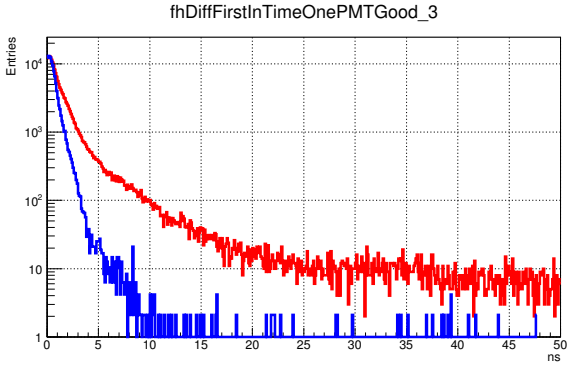

Рис. 3.7:

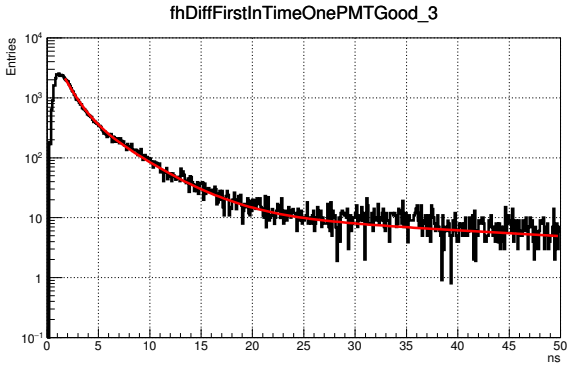

Рис. 3.8:

Заключение

Основные результаты диссертационной работы

Науная новизна результатов, полученных автором

Представление основных положений и результатов

Публикации

Список литературы

[1] bibitem1