分类号_____ 密级_____

UDC _____ 编号_____

中国科学院研究生院

博士学位论文

大亚湾中微子实验数据获取系统的研究与实现

季筱璐

指导教师 王贻芳 研究员 中国科学院高能物理研究所

申请学位级别 博士学位 学科专业名称 粒子物理与原子核物理

论文提交日期 **2010. 4** 论文答辩日期 **2010. 5. 25**

培养单位 中国科学院高能物理研究所

学位授予单位 中国科学院研究生院

答辩委员会主席 赵京伟

Research and Implementation of the DAQ System for Daya Bay Neutrino Experiment

A Dissertation for the Doctoral Degree of Philosophy

in the Graduate School of Chinese Academy of Sciences

By

Ji Xiaolu

Directed By

Prof. Wang Yifang

Chinese Academy of Sciences

April，2010

声明

本人郑重声明：所呈交的学位论文，是本人在导师指导下独立进行研究工作所取得的成果，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签名：_____日期：_____

关于学位论文使用授权的说明

本人完全了解中国科学院高能物理研究所“关于中国科学院高能物理所研究生论文及研究成果使用权的规定”（2001）高发研生字第 315 号文件，即：高能物理研究所拥有在著作权法规定范围内学位论文的使用权，其中包括：（1）已获学位的研究生必须按规定提交学位论文，高能物理研究所可以采用影印、缩印或其他复制手段保存研究生上交的学位论文；（2）为教学和科研目的，高能物理研究所可以将公开的学位论文作为资料在图书馆、资料室等场所供科研人员阅读，或在所内网站供科研人员浏览部分内容；（3）根据《中华人民共和国学位条例暂行实施办法》，向国家图书馆等相关部门报送可以公开的学位论文。

签名：_____日期：_____

摘要

数据获取系统是高能物理实验的重要课题，一个成熟可靠、性能优良的数据获取系统对于实验的成功起着关键性的作用。

本篇论文论述了大亚湾中微子实验的数据获取系统的研究与实现。以满足大亚湾中微子实验的具体需求为目标，在借鉴 ATLAS 与 BESIII DAQ 软件的基础上开发了大亚湾实验的数据获取系统。论文中详细介绍了数据获取系统的设计方案，包括硬件框架与软件部署，并且通过实例检验了方案的可行性。

作为大亚湾数据获取系统的重要组成，本文重点研究了数据获取的读出系统的设计与实现。读出系统创新性的在高能物理实验中采用了嵌入式 Linux 操作系统，利用与 VME 总线的交互技术进行中断响应与数据传输，通过多线程编程技术实现任务的并发以提高读出效率。一系列测试中对各项指标的考察表明基于嵌入式 Linux 的数据获取读出系统功能完善、性能优良，可以满足系统需求。

性能是数据获取系统的重要指标，因此对数据读出以及传输性能的关注也是本文重点探讨的内容。文中对于数据流传输的整个过程中各环节的性能情况都做出了详细的分析，在保证数据质量的前提下，通过对软件机制的研究以尽可能做到性能的最优化。

对数据获取系统的存盘数据文件进行读取分析，是检验系统工作正确性与稳定性的重要手段。文中介绍了通过回读数据文件对电子学各通道性能的检验，以及利用 miniDryRun 的实验数据对触发效率进行的初步分析。

目前大亚湾中微子实验的数据获取系统已经完成了基本的框架设计，基于反中微子探测器读出机箱的数据获取系统的最终设计方案在 2009 年 12 月通过了专家评审，并且成功应用在随后开始的大亚湾 miniDryRun 实验中。实验结果表明，数据获取系统的性能和可靠性满足大亚湾实验的设计要求。

关键词：大亚湾中微子实验，数据获取系统，嵌入式 Linux，读出系统

Abstract

Data Acquisition system is an important research project in high energy physics experiments. A reliable and capable DAQ system plays a key role in the experiments.

This dissertation presents the research and implementation of DAQ system for Daya Bay neutrino experiment. The DAQ system for Daya Bay experiment had been developed based on ATLAS and BESIII DAQ. Its design scheme for hardware framework and software arrangement was newly developed. And its feasibility is confirmed by system tests on an AD VME crate.

This dissertation focused on the read-out system which played an important role in Daya Bay DAQ system. Embedded Linux operating system was creatively introduced as a readout software platform in high energy physics experiment. DAQ readout system uses the VME bus transport technology and threading tools. According to a series of tests, the performance of readout system based on Linux could meet the system requirements.

Performance behavior, an essential feature of DAQ system, was also fully studied. Each segment during dataflow transmission was meticulously investigated. On the premise of the high quality data, the software mechanism was optimized to improve the system efficiency.

Reading and analyzing data files of DAQ system is an important method to test validity and stability of the system. In this dissertation, the test result of electronics performance by reading data files is introduced, and the analysis of trigger efficiency with miniDryRun data is showed as well.

The basic framework design of DAQ system for Daya Bay neutrino experiment has been completed. In December 2009, the final design of DAQ system based on PMT readout electronics has passed the experts' review. At the same time, DAQ system run successfully during miniDryRun experiment. Data collected during miniDryRun showed that the capability and reliability of the DAQ system could fulfill the demand of Daya Bay experiment.

KEY WORDS： Daya Bay Neutrino Experiment, DAQ, Embedded Linux, Read-out System

目录

摘要…… I
Abstract……III
目录……V
图目录……XI
表目录…… XVII
1 绪论……1
1.1 引言 ……1
1.2 大亚湾中微子实验简介 ……2
1.2.1 实验的科学意义 ……2
1.2.2 实验方案设计 ……4
1.3 高能物理实验中的数据获取系统 ……9
1.3.1 数据获取系统在高能物理实验中的地位 ……9
1.3.2 数据获取系统的任务 ……9
1.3.3 数据获取系统随高能物理实验的发展 ……10
1.3.4 国内外高能物理实验中的数据获取系统 ……11
1.4 论文工作 ……14
1.5 论文组织 ……16
2 大亚湾中微子实验的数据获取系统 ……17
2.1 大亚湾数据获取系统的任务 ……18
2.2 系统需求 ……18
2.2.1 功能需求 ……18
2.2.2 性能需求 ……21
2.3 系统硬件架构 ……23
2.3.1 VME 前端 ……24
2.3.2 DAQ 控制 ……24
2.3.3 数据流传输与数据存储 ……25
2.4 系统软件框架 ……25
2.4.1 数据流部分 ……26
2.4.2 在线软件部分 ……27
2.4.3 大亚湾 DAQ 主要组件及其功能 ……28

2.5 DAQ 数据文件格式 32
2.6 系统部署 33
2.6.1 系统状态转换 33
2.6.2 Partition 36
2.6.3 Run 36
2.7 DAQ 与外界的交互 37
2.8 本章小结 37
3 大亚湾实验数据获取读出系统的设计 39
3.1 读出系统任务 39
3.2 读出系统性能需求 39
3.3 读出系统硬件环境 41
3.3.1 VME 总线介绍 42
3.3.2 MVME5500 简介 43
3.3.3 UNIVERSEII 芯片简介 44
3.4 读出系统软件框架 45
3.4.1 嵌入式实时操作系统 Linux 46
3.4.2 VME 总线访问 48
3.4.3 多任务并行的读出系统 53
3.4.4 与其他系统的接口定义 54
3.5 国内外高能物理实验中的读出系统 55
3.5.1 ATLAS 实验的读出系统 55
3.5.2 BESIII 实验的读出系统 57
3.5.3 BELLE 实验的读出系统 58
3.5.4 小结 60
3.6 本章小结 61
4 基于嵌入式 Linux 的读出性能研究 63
4.1 研究背景 63
4.2 单项性能测试 63
4.2.1 测试环境介绍 64
4.2.2 单周期读写性能测试 64
4.2.3 DMA 传输性能测试 66
4.2.4 中断开销测试 69
4.2.5 上下文切换 71
4.2.6 网络性能测试 72

4.2.7 单项测试小结 73
4.3 综合性能测试 74
4.3.1 测试平台介绍 74
4.3.2 综合测试性能结果 76
4.4 VME Universe 驱动的研究与优化 77
4.4.1 中断的处理 77
4.4.2 内存泄漏的解决办法 82
4.4.3 cache 相关的处理 84
4.5 其他提高系统性能的措施 88
4.5.1 多事例读出 88
4.5.2 并行的多任务系统与任务优先级的设置 89
4.6 本章小结 90
5 大亚湾实验数据获取读出系统的实现 93
5.1 读出系统的状态转换与实现 93
5.2 电子学配置数据库 95
5.2.1 DAQ 配置数据库 95
5.2.2 电子学配置数据库的设计 96
5.2.3 电子学配置数据库的实现 98
5.2.4 电子学配置的实现 99
5.3 多任务并行系统的实现机制 100
5.3.1 缓冲区保护 100
5.3.2 线程间通信 101
5.3.3 读出系统中各任务的具体实现 103
5.4 数据检查与组装 107
5.4.1 DAQ 事例格式 107
5.4.2 电子学数据格式 109
5.4.3 数据检查 110
5.4.4 数据组装 111
5.5 数据流监测 113
5.5.1 is_monitor 113
5.5.2 读出系统数据流监测 115
5.6 出错处理 116
5.6.1 消息报告显示程序 116
5.6.2 读出系统的错误处理 116

5.7 启动与退出的时序考虑 118
5.8 多种运行模式的实现 119
5.8.1 物理取数模式 120
5.8.2 FEE 电子学诊断模式 121
5.8.3 台阶模式 122
5.8.4 AD 刻度模式 123
5.8.5 小结 124
5.9 本章小结 124
6 单 AD 机箱的数据获取系统 125
6.1 硬件组成 125
6.1.1 PMT 读出电子学简介 125
6.1.2 DAQ 系统硬件配置 127
6.2 软件环境 128
6.3 DAQ 与电子学系统的联调 130
6.3.1 测试软件包的开发与维护 130
6.3.2 电子学配置流程 130
6.3.3 电子学系统功能的完善 132
6.4 DAQ 系统工作情况 132
6.5 结合 AD 机箱的数据获取性能分析 134
6.5.1 读出系统的数据通过能力 135
6.5.2 DAQ 后端组件的数据通过能力 138
6.6 系统稳定性分析 140
6.7 数据文件的回读分析 141
6.7.1 检验 DAQ 系统的工作情况 141
6.7.2 PMT 电子学通道性能检验 142
6.8 本章小结 147
7 多机箱数据获取系统的设计 149
7.1 RPC 电子学的 DAQ 系统 149
7.2 多探测器取数的同步控制 151
7.3 多探测器取数模式下的读出系统 151
7.4 数据流合并 152
7.4.1 数据流合并的内容 152
7.4.2 数据流合并的原则 152
7.4.3 数据流合并的位置 152

7.5 本章小结 153
8 大亚湾 miniDryRun 实验 155
8.1 miniDryRun 实验介绍 155
8.2 基于 miniDryRun 实验的触发效率的研究 157
8.2.1 研究意义 157
8.2.2 实验方法 157
8.2.3 计算触发效率的第一种方法 160
8.2.4 计算触发效率的第二种方法 165
8.2.5 两种方法的比较 168
8.2.6 影响触发效率的因素 169
8.2.7 关于触发效率的研究小结 175
8.3 miniDryRun 实验中的 DAQ 系统 175
8.4 本章小结 176
9 总结…… 177
参考文献… 179
附录 A 论文中常见词汇解释 185
附录 B 大亚湾 DAQ 数据格式说明 190
附录 C 大亚湾数据文件格式说明 194
附录 D 电子学配置数据库 197
发表文章目录 203
致谢…… 205

图目录

图 1-1 目前国际上对 $\sin^2 2\theta_{13}$ 的实验测量，左图) 直接测量；右图) 目前全部中微子实验数据的拟合结果 3
图 1-2 理论上对 $\sin^2 2\theta_{13}$ 数值大小预言的分布情况 3
图 1-3 大亚湾实验的总体方案图 5
图 1-4 大亚湾实验中心探测器示意图 6
图 1-5 大亚湾实验反符合探测器示意图 7
图 1-6 电子学系统与触发系统逻辑关系图 8
图 1-7 DAQ 在高能物理实验中的位置 9
图 1-8 数据获取系统的三个发展阶段 11
图 1-9 CMS 实验的数据获取系统示意 12
图 1-10 ATLAS 实验的数据获取系统架构示意图 12
图 1-11 BESIII 数据获取系统示意图 14
图 2-1 大亚湾数据获取系统总体框图 23
图 2-2 大亚湾实验数据获取系统框架 24
图 2-3 数据获取系统软件框架 25
图 2-4 数据流结构图 26
图 2-5 数据获取系统软件部署框图 27
图 2-6 在线软件之间的交互 27
图 2-7 读出系统 ROS 功能示意图 28
图 2-8 EFD 与 ROS、SFO 之间的消息传递时序图 28
图 2-9 运行控制树实例 29
图 2-10 进程管理（PMG）体系结构 30
图 2-11 信息服务组件接口 30
图 2-12 消息报告组件接口 31
图 2-13 DAQ 用户界面的主要功能组成 31
图 2-14 数据文件的三级结构示意图 32
图 2-15 大亚湾 DAQ 数据文件格式 33
图 2-16 数据获取系统基本状态图 34
图 3-1 读出系统硬件结构示意图 41
图 3-2 数据获取读出系统结构框图 42

图 3-3 MVME5500 外观图 43
图 3-4 MVME5500 内部结构图 44
图 3-5 UniverseII 芯片外观图 45
图 3-6 Universe II 芯片结构图 45
图 3-7 Timesys Linuxlink 47
图 3-8 VME 总线读周期[] 48
图 3-9 PCI 总线与 VME 总线的映射关系示意 49
图 3-10 Direct Mode DMA 传输流程示意图 50
图 3-11 Linked List Mode DMA 传输流程示意图 50
图 3-12 Command Packet 结构示意图 51
图 3-13 IACKIN*/IACKOUT*菊花链 53
图 3-14 数据获取读出系统各任务间交互关系图 54
图 3-15 读出系统 ROS 和其他组件的接口 55
图 3-16 RCD Monitor 功能示意 56
图 3-17 BESIII 读出系统整体组成结构示意 58
图 3-18 TDC readout system 结构示意图 59
图 3-19 SVD readout system 结构示意图 60
图 4-1 单项测试硬件环境 64
图 4-2 单周期总线读写时间测试程序流程图 65
图 4-3 DMA 传输时间测试程序流程图 66
图 4-4 32 位 DMA 传输速率与传输每字节的平均时间随传输数据块的变化情况 68
图 4-5 MVME2431 产生中断流程图 69
图 4-6 MVME5500 接收中断示意图 70
图 4-7 线程切换开销测试流程图 71
图 4-8 网络速率测试流程图（client 端） 73
图 4-9 综合测试硬件环境 75
图 4-10 综合测试性能图 76
图 4-11 两种中断释放方式示意图 78
图 4-12 VME Universe 驱动的中断处理流程 78
图 4-13 修改后的中断处理流程 80
图 4-14 修改驱动前数据传输性能情况 81
图 4-15 驱动优化后数据传输性能情况 82
图 4-16 应用程序、内核与硬件的关系 83

图 4-17 中断函数调用示意图.....83
图 4-18 修改后的中断函数调用示意图.....84
图 4-19 优化前后事例率对比.....87
图 4-20 采用多事例读出模式对 CPU 占用率的影响.....88
图 5-1 简化后的读出系统状态转换图.....94
图 5-2 配置服务的用户接口.....95
图 5-3 电子学配置数据库结构.....97
图 5-4 电子学配置数据库示例.....98
图 5-5 在线数据库中的电子学配置信息.....99
图 5-6 电子学配置的实现流程.....99
图 5-7 生产者-消费者模型.....100
图 5-8 环形缓冲区示意图.....101
图 5-9 信号量使用示例.....102
图 5-10 读出任务间数据通信示意图.....103
图 5-11 读出系统任务时序关系流程图.....103
图 5-12 CBLT 读出任务流程图.....104
图 5-13 组装任务流程图.....105
图 5-14 网络发送任务流程图.....106
图 5-15 数据流监测任务流程图.....107
图 5-16 大亚湾实验 DAQ 事例格式.....108
图 5-17 事例头部格式.....109
图 5-18 电子学读出插件数据头部格式.....109
图 5-19 读出系统事例组装示意图.....112
图 5-20 读出系统多事例读出组装示意图.....113
图 5-21 is_monitor 主界面.....114
图 5-22 is_monitor 子界面.....114
图 5-23 读出系统数据流监测界面.....115
图 5-24 消息报告显示程序界面.....116
图 5-25 读出系统的数据检查结果处理示意图.....117
图 5-26 异常数据组装示意图.....118
图 5-27 读出系统退出时序.....119
图 5-28 物理取数模式下读出系统工作流程图.....120
图 5-29 FEE 电子学诊断模式下读出系统工作流程图.....121
图 5-30 台阶取数模式下读出系统工作流程图.....122

图 5-31 AD 刻度模式下读出系统工作流程图 123
图 6-1 AD 电子学机箱硬件图 125
图 6-2 AD 机箱内部 FEE 与 LTB 之间的信号传递 127
图 6-3 单 AD 机箱的 DAQ 系统示意图 128
图 6-4 大亚湾 DAQ 软件版本升级情况 129
图 6-5 PMT 读出电子学插件配置流程 131
图 6-6 简化的电子学配置流程 132
图 6-7 DAQ 长期稳定性实验 134
图 6-8 数据库查询功能示意 134
图 6-9 DAQ 数据流简化示意图 135
图 6-10 单机箱 CBLT 读出性能情况 136
图 6-11 使用 dummy ROS 的 DAQ 读出性能 139
图 6-12 LTB 逻辑框图 140
图 6-13 FEE 单通道输入 DAC 与输出 ADC 线性关系示意 143
图 6-14 FEE 电子学诊断结果图 144
图 6-15 FEE 电子学诊断结果（有死道的情况） 145
图 6-16 FEE 单通道噪声谱 146
图 6-17 台阶模式得到电子学噪声水平 146
图 7-1 RPC 电子学读出逻辑图 149
图 7-2 数据流合并的两种方案 153
图 8-1 miniDryRun 实验中的 AD 俯瞰图 155
图 8-2 miniDryRun 实验中的电子学系统 156
图 8-3 trigger time interval 158
图 8-4 不同触发阈值条件下的 trigger interval 159
图 8-5 触发时间间隔分布情况 160
图 8-6 挑选 triggerInterval 方法得到的效率曲线 161
图 8-7 QSum 阈值的设置 161
图 8-8 设置 QSumCut 对触发效率结果的影响 162
图 8-9 对 QSum 与 peSum 设置阈值条件进行 Cut 164
图 8-10 采用 QSumCut 与 peSumCut 得到的触发效率曲线对比 164
图 8-11 寻找有效 LED 事例示例 166
图 8-12 周期 T=2.5ms 时改变ΔT得到的触发效率情况 167
图 8-13 对应单倍与双倍 LED 周期的触发时间间隔分布 167
图 8-14 修正 LED 发光周期后得到的触发效率曲线 168

图 8-15 两种分析方法得到的效率曲线的比较.....168
图 8-16 ESum 触发模式下阈值对触发效率的影响.....170
图 8-17 Nhit 触发模式下阈值对触发效率的影响.....170
图 8-18 ESum 触发模式下 LED 光强对触发效率的影响.....171
图 8-19 Nhit 触发模式下 LED 光强对触发效率的影响.....172
图 8-20 PMT 增益对触发效率的影响.....173
图 8-21 FEE 击中率 hit_rate 分布.....174

表目录

表 2-1 实验中各种取数模式需求统计.....20
表 2-2 大亚湾实验触发率和数据量需求估算.....21
表 2-3 数据获取系统状态表.....35
表 2-4 数据获取系统链命令.....36
表 3-1 VME 总线中断的分类.....52
表 4-1 32 位 DMA 传输测试结果.....67
表 4-2 百兆网口传输性能测试结果.....73
表 4-3 双网卡网络传输性能测试结果.....73
表 4-4 基于 Linux 的 MVME5500 单项测试结果.....74
表 4-5 优化前后事例率对比.....86
表 4-6 串行读出性能.....89
表 4-7 线程优先级对读出性能的影响.....90
表 5-1 电子学读出数据格式.....110
表 6-1 大亚湾实验单 AD 机箱整体取数性能.....133
表 6-2 单机箱 CBLT 读出性能.....136
表 6-3 事例组装任务对读出性能的影响.....138
表 8-1 miniDryRun 实验中 PMT 与 FEE 连接关系表.....156
表 8-2 实验中 24 个 PMT 的刻度参数与对应 FEE 的台阶值.....163

1 绪论

1.1 引言

高能物理，又称粒子物理，是研究物质的最小组元的特性及其相互作用规律的一门前沿科学[1]。高能物理探索的是比原子核更小的微观粒子的性质，通过了解和掌握微观粒子的基本特性和运动规律，从而进一步地研究宏观世界的自然规律。

粒子物理实验可概括为粒子源、探测器和数据分析三个组成部分。粒子源可以是加速器或对撞机，也可以是不用加速器的宇宙线[2]。当前，新一代粒子物理实验的三个研究前沿是：一：加速器物理实验的高能量研究，以欧洲核子中心(CERN, European Organization for Nuclear Research)目前刚投入运行的质子-质子(P-P)对撞机 LHC(Large Hadron Collider)为代表；二：高亮度研究，以粒子工厂为代表，例如 B 粒子工厂，目前世界上正在运行的B粒子工厂有 SLAC(Stanford Linear Accelerator Center)的 BaBar 实验和日本 KEK(High Energy Accelerator Research Organization)的 Belle 实验；三：非加速器物理实验研究，这类实验具有探测器体积大、灵敏度高、低本底低噪声的特点，用来探测低事例率的物理现象，如宇宙线或天体过程中的中微子、宇宙暗物质粒子等。如正在南极进行的 IceCube 中微子观测实验，用来观测中微子与超新星事例的日本 Super-Kamiokande 实验。正在中国建造的大亚湾反应堆中微子实验，也是非加速器物理实验的代表之一。

本文重点论述的就是大亚湾反应堆中微子实验（Daya Bay Reactor Neutrino Experiment）的数据获取系统（DAQ）的研究与实现。

1.2 大亚湾中微子实验简介

1.2.1 实验的科学意义

中微子物理是当今粒子物理，天体物理与宇宙学的交叉与热点。在 Super-K、 SNO 与 KamLAND 实验发现中微子振荡后，人们认识到这是发现粒子物理标准模型之外新物理的突破口与关键。

目前国际中微子物理实验的前沿是精确测量中微子混合角 θ_{13}。无论从理论还是实验上来说，精确测量 $\sin^2 2\theta_{13}$ 都具有重大的科学意义。首先，它是中微子物理中六个最基本的参数之一，其数值的大小决定了未来中微子物理的发展方向。它的大小直接决定了中微子振荡中的 CP 破坏强度。后者是目前最有希望解释宇宙中物质-反物质不对称现象的（leptogenesis 机制）。如果它是如人们所预计的在 0.01－0.03 左右的话，则中微子的 CP 相角可以测量，宇宙中物质与反物质的不对称可能得以解释。如果它太小，则中微子的 CP 相角无法测量， 目前用中微子来解释宇宙中物质与反物质不对称的理论便难以证实。其次，$\sin^2 2\theta_{13}$ 接近于零也预示着新物理或一种新的对称性的存在。因此不论是测得 $\sin^2 2\theta_{13}$ 或只给出其上限值，均有极为重要的意义。第三，它与基本粒子物理的大统一理论有关。它是轻子味混合的六个基本物理参数中的两个未知参数之一，测量它的大小将有助于我们了解轻子味混合与夸克味混合之间的关系。如果 θ_{13} 不是太小，那么有可能在 10^{16}GeV 能量标度下实现大统一。第四，中微子振荡是第一个，也是唯一一个超出目前粒子物理标准模型的物理现象。影响标准中微子混合与振荡的新物理效应很有可能比 θ_{13} 所引起的次级效应更小。因此精确测定它的数值将使我们能够寻找与鉴别新物理。

目前国际上对 $\sin^2 2\theta_{13}$ 的测量均未给出肯定的答案，如图 1-1 所示[3]。Chooz, Palo Verde 与 Super-K 等实验在直接搜寻中未发现θ_{13}振荡的实验证据，而利用目前全部中微子实验数据的拟合结果得出的 $\sin^2\theta_{13}$ 的最可几值约为 0.01（相应于 $\sin^2 2\theta_{13}$=0.04）。

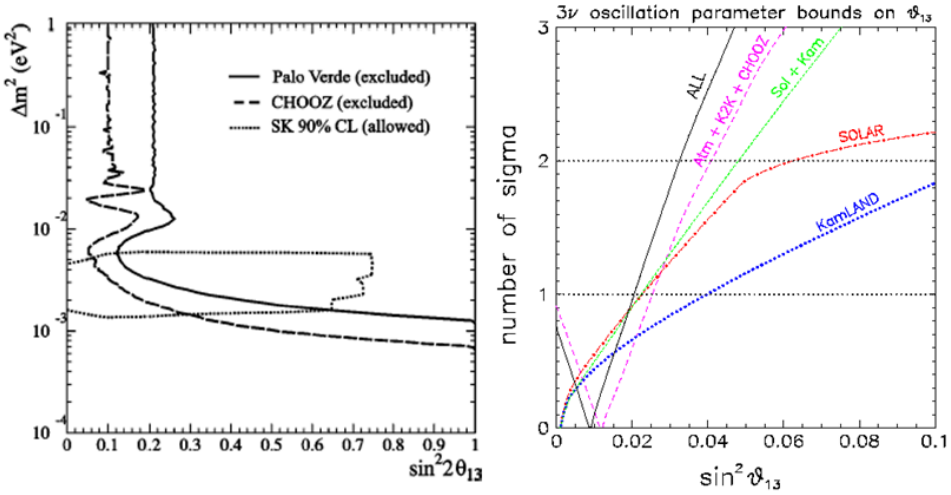


图 1-1　目前国际上对 $\sin^2 2\theta_{13}$ 的实验测量，左图) 直接测量；右图) 目前全部中微子实验数据的拟合结果

而理论上目前对 $\sin^2 2\theta_{13}$ 的预言也相差很大，不同的模型给出完全不同的数值，但基本上集中在 0.001-0.1 之间，如图 1-2 所示[3]。因此实验上希望测量 $\sin^2 2\theta_{13}$ 的灵敏度应达到 0.01 以下。

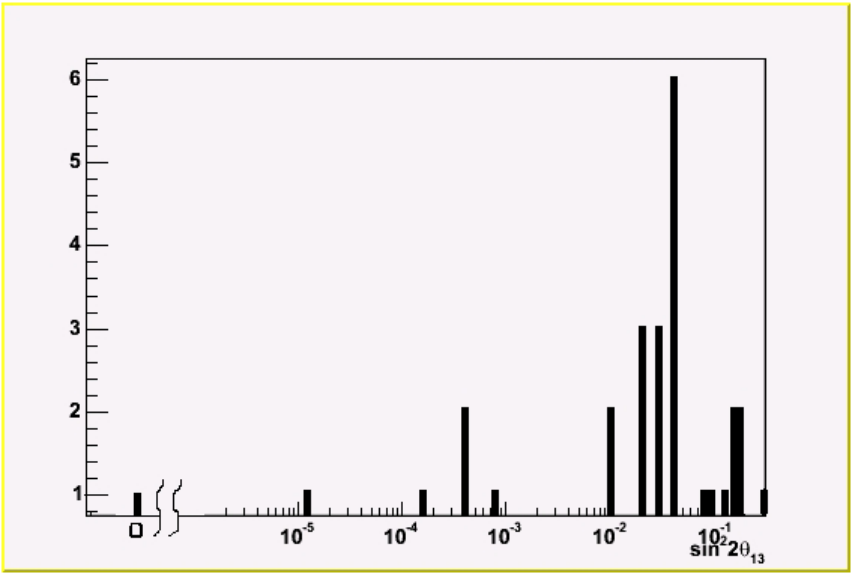


图 1-2 理论上对 $\sin^2 2\theta_{13}$ 数值大小预言的分布情况

用反应堆中微子测量 θ_{13} 具有造价低，速度快，物理结果干净，无歧义的特点，而大亚湾核电站则是全世界最佳场所，一是其功率高，二是反应堆都靠近山，可以屏蔽宇宙线本底。大亚湾中微子实验的设计是当前国际上精度最高的，可以达到好于 1%的目标。目前由反应堆中微子实验给出

的 $\sin^2 2\theta_{13}$ 的上限为 0.17，对应的实验误差为 3.9%。将 $\sin^2 2\theta_{13}$ 的测量灵敏度下推到 0.01 左右在理论与实验方面都具有极为重要的意义，这要求实验误差达到小于 1%的水平，对实验工作提出了极大的挑战。

1.2.2 实验方案设计

1.2.2.1 总体方案

反应堆中发生的核裂变反应在释放核能的同时，会产生大量的电子反中微子。根据目前的中微子混合理论，这些中微子会发生振荡，变成其它类型的中微子。这样在一定距离观测电子型反中微子时会发现有一部分中微子消失了。在距离小于几公里时，振荡的振幅取决于 $\sin^2 2\theta_{13}$，频率取决于中微子的能量和参数Δm_{32}^2。根据其它类型的中微子振荡实验可以确定振荡最大值会发生在 1800-2400 米。大亚湾反应堆中微子实验将通过测量反应堆释放的中微子在不同距离处的数量与能谱来确定是否存在振荡，并确定振荡的参数，为测量 CP 破坏、研究超出标准模型的新物理奠定基础。

测量 $\sin^2 2\theta_{13}$ 是一个非常精密的实验，实验的质量直接取决于对系统误差的控制。为了在 90%以上的置信水平下确认θ_{13} 振荡的存在，并达到 $\sin^2 2\theta_{13}<0.01$ 的精度，要求实验的总误差小于 0.5%。这对实验方案的设计，特别是探测器的设计和本底的控制，是一个挑战。

实验的要点是：1）充分利用大亚湾核电站功率大的优点。大亚湾反应堆群将成为世界上功率第二大的近距离的反应堆群，因此实验的事例率高；2）建造近端和远端的中微子探测器，利用中微子近端和远端距离的变化进行中微子振荡的相对测量，从而抵消实验的大部分误差，大大提高反应堆中微子实验的精度；3）利用大亚湾核电站的地理条件，近端和远端的探测器都将有较厚的岩石覆盖层以减小本底的影响。足够的岩石覆盖层是减小本底最有效的手段。由于大亚湾的独特的地理条件，在国际上提出的同类方案中，大亚湾实验的精度最高，而且最有可能成为第一个测到θ_{13}不为零的实验[4]。

1.2.2.2 实验布局

大亚湾反应堆中微子实验将使用三个实验厅对中微子进行远近相对测量，包括两个近厅和一个远厅。每个实验厅内主要有 3 种探测器：中心探测器（反中微子探测器，Anti-neutrino Detector，AD），水切伦科夫探测器（水池探测器，Water Pool Detector，WP）和阻性板探测器（RPC）。

实验的布局如图 1-3。两个近点实验厅分别对大亚湾和岭澳的反应堆进行测量，离反应堆的距离分别为 364 米和 504 米，测量结果结合各反应堆的热功率将精确给出反应堆释放的中微子总数和能谱。远点实验厅距岭澳核电站为 1615 米，距大亚湾核电站为 1985 米。远点探测器与近点探测器的测量结果相比较，就可以知道是否存在中微子振荡。各探测器位置的选取经过了详细的最优化计算，主要影响因素包括根据山的具体形状计算的宇宙线本底、反应堆误差的抵消、估计的探测器误差、以及统计误差等。

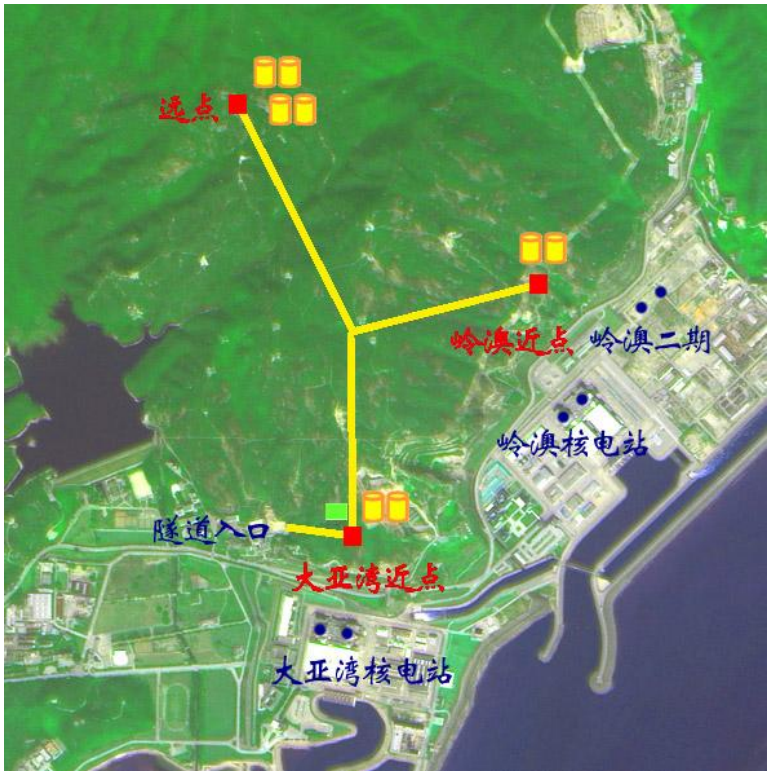


图 1-3 大亚湾实验的总体方案图

1.2.2.3 各子探测器简介

（1）反中微子探测器

反中微子探测器观测中微子反β 衰变反应产生的正电子与中子信号，只对电子反中微子敏感。为减少系统误差，反中微子探测器设计成三层同心圆柱结构，如图 1-4。最里层为掺钆的液体闪烁体（简称液闪）作为中微子的靶；中间层为普通的液体闪烁体，用来提高能量收集效率，减少能量判选误差；最外层为矿物油，用来屏蔽天然放射性。各层之间用高透光度的有机玻璃隔开。正电子与中子在液体闪烁体中产生闪烁光，用来探测闪烁光的光电倍增管（PMT）放置在最外层的矿物油中。

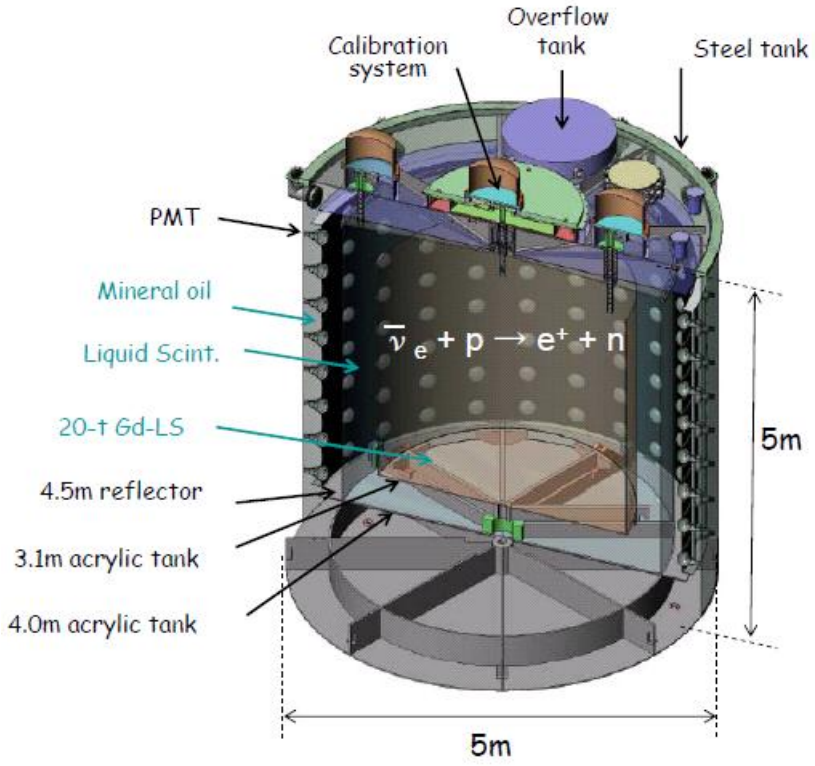


图 1-4 大亚湾实验中心探测器示意图

（2）反符合探测器

反应堆中微子实验的能量较低，宇宙线和天射放射性都能在这一能区带来大量本底。为了减小宇宙线本底，中微子探测器必须放在较深的地下，同时必须有足够的主动和被动的屏蔽来去除本底。大亚湾实验近点和远点地下实验厅处的宇宙线流强比地表分别小 100 倍和 2000 倍。剩余的高能宇宙线在中微子探测器周围的岩石中也能产生大量的中子，这些中子能漂移到中心探测器中形成本底。为了屏蔽这些中子以及岩石中的天然放射性，大亚湾实验的中微子探测器周围将用 2.5 米以上的水屏蔽。水是屏蔽中子最

有效的材料。同时山体岩石产生的天然放射性也非常高，会形成大量本底，2.5 米的水屏蔽层可以将它压低 1 百万倍以上。

除了水构成的被动屏蔽外，大亚湾实验将采用两种宇宙线探测器作为主动屏蔽，如图 1-5 所示。一种是水切伦科夫探测器（WP），由在水池中放置光电倍增管形成。宇宙线穿过水时会产生切伦科夫光，通过光电倍增管探测切伦科夫光，可以探测宇宙线。为了提高探测器的可靠程度，水池被分成内外两层，由不透光的薄膜隔开。水池壁与隔膜均粘贴高反光度的 Tyvek 膜，以提高宇宙线的探测器效率。内外两层均有光电倍增管，将独立地作为宇宙线探测器。模拟表明水切伦科夫探测器的探测效率为 98%以上。另外一种为阻性板探测器（RPC），放置在水池上方。大亚湾将采用 4 层 RPC，通过触发选择来降低 RPC 的噪声和本底，同时保持较高的探测效率（>90%）。

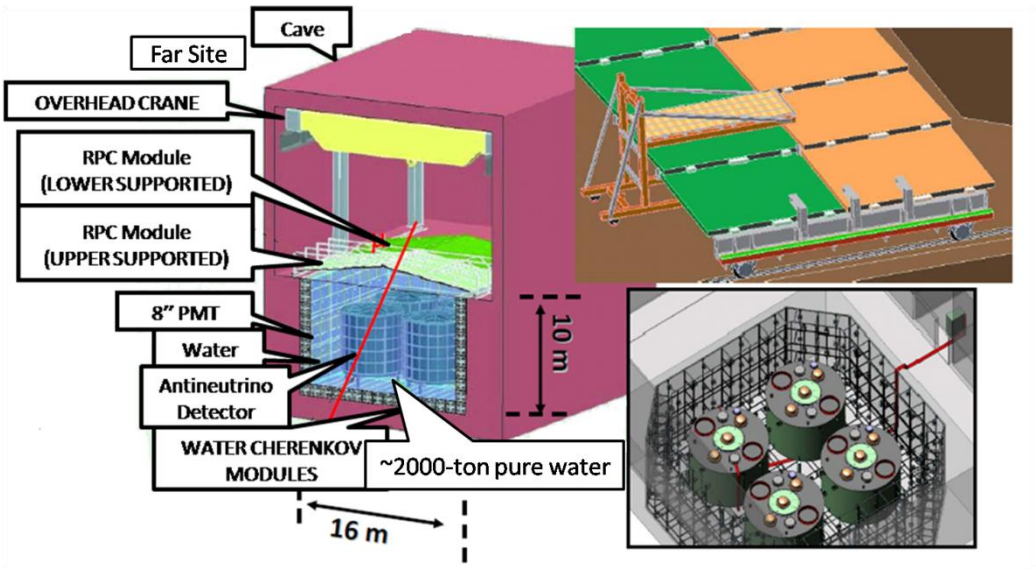


图 1-5 大亚湾实验反符合探测器示意图

两种宇宙线探测器联合起来，总效率可达 99.5%。经模拟计算，中子和天然放射性造成的本底可以控制在信号的 0.2%左右，满足实验的精度要求。

（3）电子学系统

电子学系统包括前端电子学子系统、触发和时钟子系统、数据获取子系统和探测器监控子系统。整个电子学系统设计的基本思路是各个探测器

（反中微子探测器、内外层水切伦科夫探测器、RPC 探测器）采用各自独立的前端电子学子系统、触发和数据读出子系统，并且每个站点有一套完整的时钟系统。

大亚湾中微子实验前端读出电子学（简称读出电子学）包括 PMT 读出电子学和 RPC 读出电子学。PMT 读出电子学用于反中微子探测器和水池切伦科夫探测器的信号读出，采用相同的硬件设计，在此基础上通过配合使用不同的 FPGA 逻辑功能来满足各自的要求。RPC 读出电子学用于阻性板探测器 RPC 的读出。

触发系统根据预先设定的物理事例的判选条件，综合从电子学系统送来的各个通道的击中信息和总能量信息完成事例判选。数据获取系统根据触发系统的判选结果，从电子学系统中读取相应的事例数据，形成文件并最终保存下来。为确保在整个实验过程中，探测器和各电子学系统能够安全、稳定地运行，还需要一个完整的探测器监控系统。

图 1-6 显示了电子学系统与触发系统之间的逻辑关系。

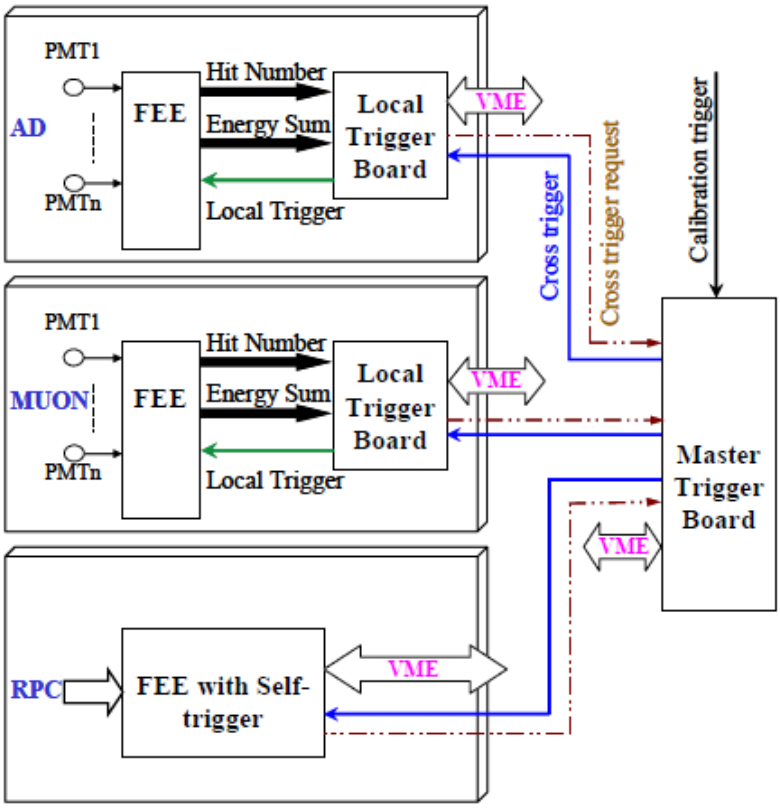


图 1-6 电子学系统与触发系统逻辑关系图

1.3 高能物理实验中的数据获取系统

1.3.1 数据获取系统在高能物理实验中的地位

如图 1-7 所示为一个典型的高能物理实验的结构框图。在现代高能物理实验中，前端电子学、触发判选和数据获取（DAQ）系统互相穿插，很难截然分开。电子学将探测器输出的信号经过放大、成形等处理后数字化，将信息暂存并作数据预处理；触发判选系统选择满足物理条件的好事例，压缩本底事例；离线数据分析将在线机器记录下的数据在离线机上进行分析和处理，把数据还原为粒子种类、能量、动量等物理量[5]。

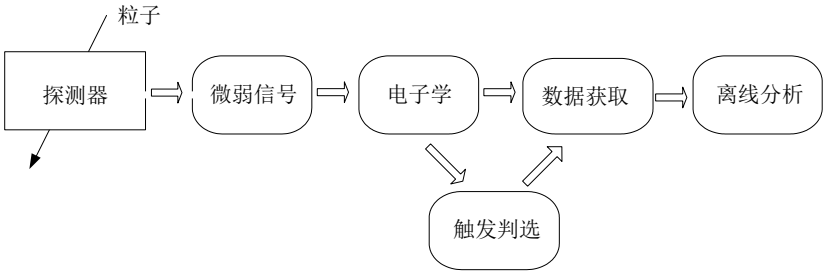


图 1-7 DAQ 在高能物理实验中的位置

高能物理实验中，数据获取系统的任务是读出通过触发判选的好事例数据，并使系统死时间尽可能地小，事例数据段迅速地汇集起来，进行事例包装和过滤，整理成为完整的有效事例，最终将标记的事例数据记录到永久介质上。同时给出反映探测器性能的各种统计图形以及所获事例的分类统计图形，监测探测器与电子学工作状态[2]。

在系统构造方面，DAQ 系统必须成为高可靠性、高稳定性、易维护、易扩充的系统。系统设备配置和软件开发工具着眼于未来技术发展趋势，尽可能采用市场上性价比最好的商业化产品。

1.3.2 数据获取系统的任务

数据获取系统的任务归结起来可列举为以下几点[2]：

- 电子学的刻度和记档
- 运行时的初始化，前端各微处理器的加载，各状态控制寄存器的设

置

- 从前端电子学读数
- 数据的预处理和装配
- 全事例数据的重建分析（在线事例筛选）
- 数据的记录（磁盘或磁带）
- 探测器运行情况的监测（抽样分析事例，建立各种直方图并作单事例和直方图显示）
- 运行界面的操作控制
- 运行条件的显示和记档（如运行 RUN 号等）
- 错误显示，报警和记档
- 磁带或磁盘上的记录的数据的回读及处理

1.3.3 数据获取系统随高能物理实验的发展

在高能物理实验迅速发展的几十年间，数据获取系统也经历着架构从简单到复杂，性能快速提升的进化过程。在上世纪六十年代，高能物理实验中仅有一百个左右的数据通道需要读取，且只有较低的事例率（几个赫兹），采用单处理器计算机就能够处理。而现代大型高能物理实验则有上百万个通道，而且要以兆赫兹频率完成数据获取。

数据获取系统的发展可以分为三个阶段，如图 1-8 所示。在 20 世纪 70 年代，前端电子学放置在 CAMAC 总线机箱内。所有的机箱插件都由一个小型计算机控制，而所有读出的数据也都集中到这个小型计算机。这种组态可以支持 Kbyte / s 的数据率；80 年代发展了较快的 VME、FASTBUS 等总线，允许多个智能插件构成分布式智能系统，并引入缓冲存储器，可以支持 Mbyte / s 的数据率；90 年代后期采用由通信网络构成的事例组装器和用计算机群作全事例过滤，系统规模和并行度更高，可以支持 Gbyte / s 的数据率[6]。

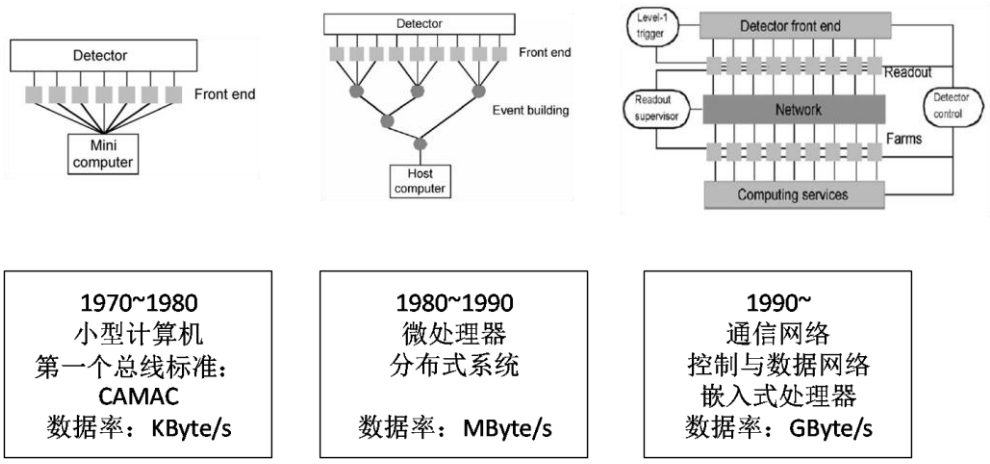


图 1-8 数据获取系统的三个发展阶段

1.3.4 国内外高能物理实验中的数据获取系统

1.3.4.1 CMS 实验的数据获取系统

LHC (Large Hadron Collider)是在 LEP(Large Electron Postion Collider)正负电子对撞机 27km 隧道的基础上兴建的 2×（8～20）TeV 质子-质子对撞机，设计亮度为$1\times10^{34}\mathrm{cm}^{-2}\mathrm{s}^{-1}$，质子束团对撞频率为 40MHz。LHC 对撞机上有 CMS，ATLAS，LHCb，ALICE 四个大型探测器[2]，这些探测器的数据获取系统都使用了当前最先进的技术。

CMS（Compact Muon Solinoid Detector）的目标是要在很大的能量区间和很高亮度的条件下，鉴别和精确测量 μ 子， 电子和光子。CMS 探测器产生的每个完整事例长度约为 1Mbytes，经过一级触发判选之后事例率从 1GHz 降到 100KHz。但是紧接着的物理分析和在线事例过滤需要从大约 1000 个前端读出缓冲区中将数据汇集到一起，考虑到存储能力和离线分析等原因， CMS 的数据获取系统必须将最终输出事例率控制在 100Hz。另外 CMS 的数据获取系统还需要做到不间断探测器监视和故障检测。

CMS 数据获取系统中有一基本模块——读出单元（Readout Unit），起到内部连接前端电子学和数据获取系统的作用，读出单元由几个 Readout Driver 和一个双端口的内存组成，直接连接与事例重建部分相通的全局交换机。整个 CMS 数据获取系统大约有 1000 个这样的单元，每个单元可以至少缓存 100000 个事例片段和达到每秒 100Mbytes 的数据传输速率。事

例重建和在线事例过滤计算单元均是基于网络交换的机群实现[7]。图 1-9 为 CMS 数据获取系统示意图。

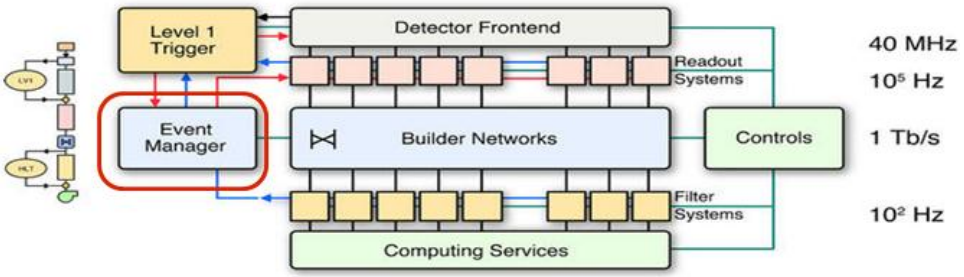


图 1-9 CMS 实验的数据获取系统示意

整个 CMS 数据获取系统的设计做到了很大程度的标准化，如光纤数据传输在触发系统和数据获取系统均得到了使用，基于公共元件和接口的前端驱动器，使得整个实验的具有统一的读出单元结构。

1.3.4.2 ATLAS 实验的数据获取系统

ATLAS 是 LHC 上四个探测器中最大的一个，也是迄今为止世界上最大的探测器[2]。ATLAS 包含三级触发系统，第一级为硬件触发， 直接连前端电子学，将事例率从 40MHz 压低至 100KHz，二级触发基于软件触发，使用高性能计算机机群处理，将事例率降到 2KHz，通过二级触发的事例会被组装成完整事例。第三级同样为软件触发，约 1600 台高性能计算机组成的机群，将事例率压至 200Hz，最终存盘数据率为 300Mbytes/s。图 1-10 为 ATLAS Trigger/DAQ 系统结构[8]。

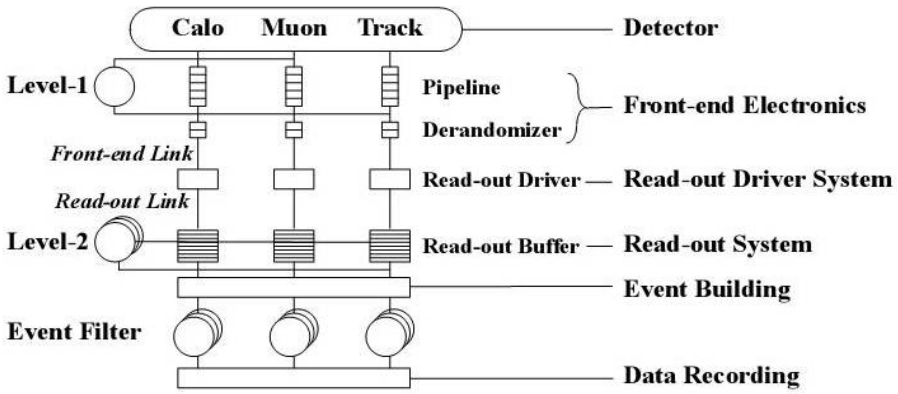


图 1-10 ATLAS 实验的数据获取系统架构示意图

ATLAS 实验的 DAQ 系统主要有两个组成部分：数据流（DataFlow）和在线软件（Online Software）。DataFlow 部分包括了 RCD(The ROD Crate DAQ), ROS（The ReadOut Subsystem）和 DataCollection 三个主要组件。其中RCD提供了在探测器电子学和DAQ之间的完整的数据获取功能的框架；ROS 负责从ROD接收数据、存储，并将数据提供给L2和EB (Event Building)；DataCollection 负责控制数据的移动，包括了从 ROS 到 LVL2 (Level-2), EventFilter 以及 MassStorage。在线软件部分是指全局的系统软件，提供了配置、控制、同步以及在 DAQ 各组件内部交换信息的功能。

1.3.4.3 BESIII 的数据获取系统

北京正负电子对撞机（Beijing Electron Positron Collider，简称 BEPC）由束流能量为 1.3 GeV 的直线加速器、束流输运线、束流能量为 1-2.8 GeV 的储存环、安装在南对撞区的探测器——北京谱仪（Beijing Spectrometer，简称 BES）和北京同步辐射装置（Beijing Synchrotron Radiation Facility，简称 BSRF）组成，是我国在国际粒子物理实验领域的代表性成果[1]。升级之后的 BEPCII 设计亮度达到$10^{33}\,\mathrm{cm}^{-2}\mathrm{s}^{-1}$，预期在 J/Ψ 能区通过一级触发判选后的事例率达到 4000Hz。实际数据处理过程中，总体数据通过速率已经超过 80 Mbytes/s。因此 BESIII 数据获取系统的设计目标就是完成高事例率与高数据通过率情况下的数据读出和处理。

BESIII 数据获取系统的整体结构如图 1-11 所示。BESIII DAQ 的主要任务是获取通过一级触发判选后的前端电子学事例数据，经过两级计算机预处理和高速网络传输，将分布在各电子学读出机箱（VME 机箱）中的事例数据段迅速地汇集到在线计算机系统上进行事例包装和过滤，整理成为完整的有效事例，最终将标记的事例数据通过网络传送到计算中心记录到永久介质上[9]。

根据系统与电子学硬件的紧密度，以及实现体系结构的不同，可以将 DAQ 系统分为前端读出系统和后端处理系统。前端读出系统主要负责数据从电子学插件的高速读出、处理到发送至后端系统的过程，可以称作 VME 机箱级处理。后端系统则负责对前端读出系统的控制以及数据从前端的接

收、组装、过滤和存盘整个过程。考虑到性能需求和当前可用的技术，BESIII DAQ 前端读出系统主要配置 VxWorks 实时操作系统和基于 PowerPC 的 VME 单板计算机。DAQ 后端系统是基于网络交换的集群实现，主要配置运行 Linux 操作系统的计算机，通过千兆以太网互连，组成高性能集群。

为了从前端电子学系统中快速读出数据并使系统死时间尽可能地小，BESIII 数据获取系统设计将大量采用多级数据缓冲技术、并行处理技术、VME 总线高速读出技术以及网络传输技术。多级数据缓冲可以有效地减小由于高能物理实验事例产生的随机性而引起的死时间，基于网络交换机的并行数据传送可以提高数据流量和完成事例的并行处理。

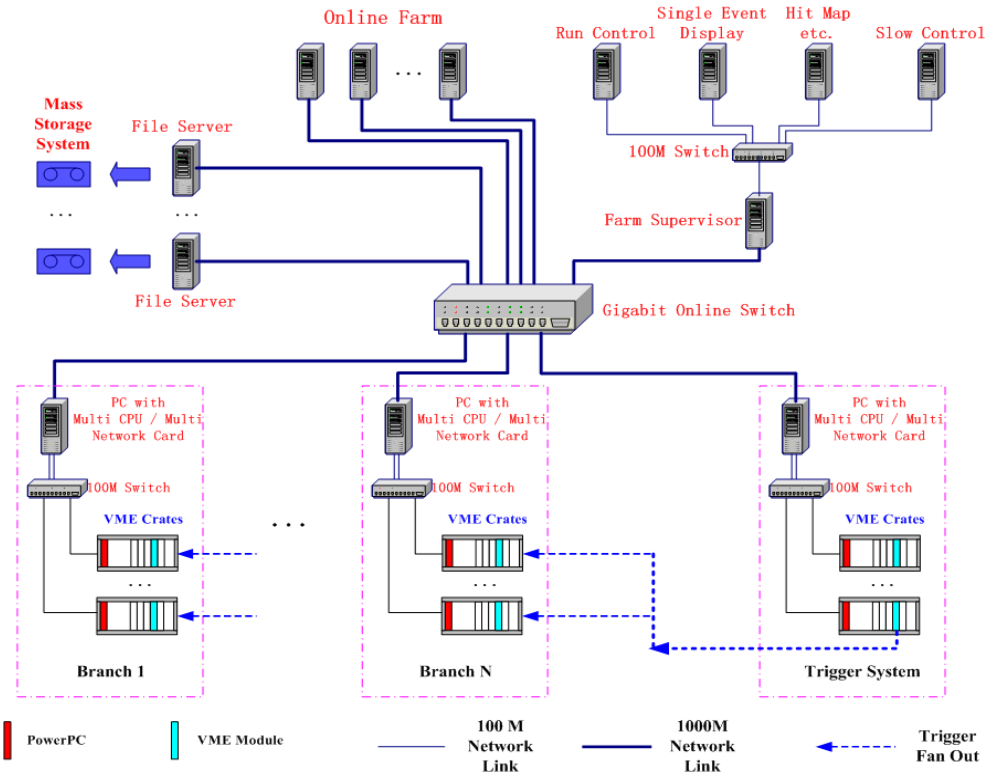


图 1-11 BESIII 数据获取系统示意图

1.4 论文工作

数据获取系统在任何一个高能物理实验中都是不可或缺的重要组成，本篇论文将对大亚湾中微子实验的数据获取系统的研究与实现展开讨论。

本文作者作为系统开发的核心成员，全程参与了数据获取系统的设计

与研发工作。其中，全面负责了数据获取读出系统的设计、研究与实现的工作，从系统调研初期模型平台的建立与基于性能的研究，到最终工程软件的实现，涉及的内容包括电子学配置数据库的建立与使用，对电子学系统工作状态的控制，数据的读取、预处理与组装，数据的传递工作等，具体内容在论文的 3-6 章中将有详细阐述。在实现读出系统的基础上，也全程承担了数据获取系统与电子学系统的联调工作。除此之外，作者还重点参与了系统开发过程中的一系列相关工作，包括数据格式的设计与实现，运行时的操作控制，消息传递与错误显示的实现，系统性能与稳定性研究，存盘数据的回读与处理等，有关内容在本篇论文的全文中均有体现。

概括的说，论文工作主要涉及到如下几个方面：

- 从 ATLAS 和 BESIII 的 DAQ 软件汲取大型软件开发所必需的辅助工具和方法，如状态机控制、数据库配置、软件通信和同步、信息服务、报错控制、代码管理等先进的软件开发手段。
- 参加 BESIII 的系统联调，在系统调试的过程中深入理解数据获取系统的工作机制，为大亚湾数据获取系统的开发奠定实践基础。
- 综合需求、硬件环境和现有技术，研究适合大亚湾数据获取的读出方案。
- 调研嵌入式 Linux 应用于数据获取读出系统的可行性，并解决实际应用中由于操作系统或驱动软件引入的问题。
- 研究和学习在数据获取系统的软件开发中需要用到的一些通用技术，如嵌入式技术、C、C++编程开发、ROOT[58]、Linux 应用和内核技术等。
- 根据实际条件开展各种功能与性能测试，定位系统瓶颈，发现潜在问题并寻找解决方案。
- 参加大亚湾系统联调，与电子学系统人员密切合作，在联调的过程中逐步加深对整个电子学读出系统的理解，从而进一步完善数据获取系统的设计。
- 利用 miniDryRun 实验中获取的 LED 数据进行触发效率相关的分析，从而加深对整个实验的理解。并且通过存盘数据文件的分析，进一

步验证数据获取系统功能的正确性。

1.5 论文组织

本论文共分为九个章节和四个附录。第一章介绍了课题背景，包括大亚湾中微子实验的简要概况以及高能物理实验中的数据获取系统的地位、任务以及发展情况。第二章分析了大亚湾中微子实验数据获取系统的系统需求以及硬软件的部署情况。第三章着重描述大亚湾实验数据获取读出系统的框架设计。第四章详细研究了嵌入式 Linux 操作系统在数据获取读出系统中的性能表现。第五章重点探讨了大亚湾实验数据获取读出系统的工程软件的实现过程与主要的软件机制。第六章详细介绍了基于单 AD 机箱的数据获取系统的工作情况与性能研究。第七章提出了基于多探测器的数据获取系统的设计方案。第八章介绍了大亚湾 miniDryRun 实验的情况，并且利用实验数据对触发效率进行了初步研究，同时也对实验中数据获取系统的工作情况进行了简要说明。第九章对论文工作进行了整体上的总结。

附录 A 给出了论文中常见的词汇解释。附录 B 和 C 分别介绍了大亚湾数据获取系统的在线事例格式与数据文件的格式。附录 D 是电子学配置数据库的 schema 文件示例。

2 大亚湾中微子实验的数据获取系统

数据获取系统在现代高能物理实验是一个非常重要的课题。大亚湾中微子实验的特点是真实事例稀有，本底事例较高，需要更高的测量精度和更多的事例样本，因此必须采用更高效的方法收集、存储数据。建造一个高能物理实验的数据获取系统，有几个重要的因素需要综合考虑：

(1). 触发率。在事例大小相同的情况下，触发率越高，数据获取系统在单位时间需要处理的数据量也越大，因此需要更高的数据处理能力；

(2). 事例大小。一个事例是指在一个触发事件到达后，由前端电子学从探测器读出，并经由前端电子学处理（如压缩等）的粒子对撞信息（如时间、电荷、径迹等）。在触发率相同的情况下，一个事例越大，数据获取系统在单位时间内需要处理的数据量也越大，从而对数据处理能力的需求也越高；

(3). 造价。高能物理实验对数据获取系统的性能的要求提出很高的要求，构建一个高能物理实验中使用的数据获取系统需要昂贵的硬件和软件支持。而这些设备的价格往往随性能的增长而指数增长。因此，在设计数据获取系统方案之前需要根据特定的物理实验目标，估计其所要达到的性能指标，选用合理的方案。

(4). 扩展性。大型的高能物理实验装置的寿命一般大于 10 年，在建成运行后，往往需要根据最新的物理发展进行实验的改造。因此，一个完善的数据获取系统需要有足够的冗余能力，并且易于扩展，以适应未来新的探测器要求。

(5). 容错性。来自前端电子学的异常数据、数据获取系统本身的不确定因素（如网络故障、系统缓冲区出错等），会对实验数据的获取造成影响。如果这些错误被延迟发现，可能导致自第一个错误发生时到错误被发现时的数据均无效。因此数据获取系统必须能及时发现错误数据，并且采取相应措施。大亚湾中微子实验是一个远程控制

实验，实验站点分散，并且远离研究基地，因此对系统可靠性和可用性提出了更高的要求。

触发率和事例率是跟高能物理实验所需要达到的工作能量和束流亮度密切相关的。建造一个数据获取系统需要同时综合考虑物理学、电子学、计算机科学等几方面的知识和技术。

2.1 大亚湾数据获取系统的任务

从探测器出来的信号经过前端电子学系统的甄别成形后，形成原始数据片。在大亚湾中微子实验中，数据获取系统的主要任务是记录反中微子探测器观察到的反中微子事例，以及反符合探测器所记录到的本底事例。根据实验需要，数据获取系统记录下的反中微子事例与本底事例都必须包含有精确的时间信息。除了记录事例数据以外，数据获取系统另外一个重要功能是监测取数状态。

因此，概括起来说大亚湾实验的数据获取系统主要需要实现以下功能：

- 控制前端电子学插件的工作状态
- 从前端电子学设备读取数据
- 把从电子学插件取到的数据片段组装成完整事例
- 在线监测数据流质量
- 实现事例数据的存储

2.2 系统需求

2.2.1 功能需求

2.2.1.1 结构需求

大亚湾中微子实验的探测器分布在 3 个站点，共有 17 个独立的探测器，包括 8 个 AD 探测器，6 个水切伦科夫探测器以及 3 个 RPC 探测器，其中远厅有 4 个 AD 探测器、2 个水切伦科夫探测器、1 个 RPC 探测器；两个近厅各有 2 个 AD 探测器、2 个水切伦科夫探测器、1 个 RPC 探测器。在

结构上要求三个站点有独立的数据获取系统，并且每个探测器都有独立的数据获取读出系统。这就意味着每个探测器都需要配备一个独立的 VME 读出机箱，机箱内除了系统控制器外，还将包括触发插件、电子学读出插件以及扇出插件（FANOUT）。

数据获取系统应当允许对每个探测器的独立操作，也可以允许对任意探测器组集的统一控制，即参与取数的探测器是可配置的。例如，可以允许同一类型的探测器共同取数，如反中微子探测器的机箱同时取数；或者是同一个站点的所有探测器共同取数，如远厅所有探测器同时取数。通过 DAQ 软件的适当配置，不同组集的探测器之间可以互不干扰地进行独立取数。

2.2.1.2 数据流需求

数据获取系统需要读出每个 VME 机箱内的各相关插件的数据片，并且根据需求组装成一个完整的探测器事例。对于大亚湾中微子实验，有两种读出类型不同的 VME 电子学机箱，分别是 PMT 电子学读出机箱以及 RPC 电子学读出机箱。在 PMT 电子学读出系统中，一个 VME 机箱中的读出电子学插件由 FEE（FrontEnd Electronics）插件、FlashADC 插件以及 LTB（Local Trigger Board）插件组成；而在 RPC 读出 VME 机箱中，读出电子学插件由 ROM（Readout Module）、RTM（RPC Trigger Module）组成。

参与取数的每个探测器都有独立的数据流，数据获取系统需要能够进行数据流合并，合并的依据是根据每个事例的时间标签来对事例进行时间排序，这样便可将各探测器的独立数据流合并成单一数据流，有关数据流合并的详细介绍，请参见 7.4 节。

2.2.1.3 运行模式需求

截至目前为止，大亚湾中微子实验中已经确定的运行模式有六种，如表 2-1 所示，分别是：

- 物理取数模式（physics）：这是正常取数时的运行模式，以记录反中微子事例以及本底为目标。

- FEE 电子学诊断模式（FEEDiag）：利用 FEE 板自身产生的标准信号输入，通过分析输入输出信号的关系，诊断 FEE 电子学插件的各通道是否工作在正常范围，是否存在死道冒道等异常情况。
- 台阶取数模式（pedestal）：在没有信号输入的情况下，设置阈值为 0，观察电子学自身的基线与噪声水平，目前这种方式仅适用于 PMT 读出电子学。
- 反中微子探测器刻度模式（ADCalib）：用于对反中微子探测器及其读出电子学整体性能的研究。
- 水切伦科夫探测器刻度模式（WPCalib）：用于对水切伦科夫探测器及其电子学整体性能的研究。
- 矿物油监测模式（MOMonitor）：用于对反中微子探测器中矿物油的性能情况的监测。

大亚湾实验的数据获取系统需要能够根据这六种运行模式各自的需求设计相应的取数机制，包括配置参数的设置、取数过程的控制等。

表 2-1 实验中各种取数模式需求统计

	Name	Interval	Time	AD	WP	RPC
physics	physics	-	-	Y	Y	Y
FEE Diagnosis	FEEDiag	one day	3 minutes	Y	Y	-
pedestal	pedestal	One day	1 minute	Y	Y	-
AD Calibration	ADCalib	One/two weeks	2 hour	Y	physics	physics
WP Calibration	WPCalib	One/two weeks	<1 minute /LED	physics	Y	physics
MO Monitoring	MOMonit	One/two weeks	10000 events /wavelength	Y	physics	physics

2.2.1.4 其他功能性需求

除了前面几节介绍到的结构需求、数据流需求以及运行模式需求之外，

为了更好的满足实验需要，提高实验的稳定性与可靠性，大亚湾数据获取系统还应当能够满足其他的一系列功能性需求，主要包括以下几点：

- 大亚湾数据获取系统需要设计成足够灵活的系统，根据需求将相关参数设计成可配的
- 应当提供友好的用户图形界面与清晰的说明文档，方便用户的使用
- 应当具有可靠的报错功能，能够第一时间报告异常
- 提供完整的数据库记录运行参数，包括软硬件配置情况等
- 有监测系统运行状态的功能，包括直方图显示、数据流监测等
- 有完整的运行日志，包括运行情况、错误信息等
- 与电子学、慢控、离线等其他各个系统有稳定的接口

数据获取系统的各项功能将会在设计与实验检验的过程中依据实验的具体需求与用户的反馈进行不断地升级与完善。

2.2.2 性能需求

大亚湾数据获取系统需要处理的数据量需求如表 2-2 所示，三个站点总体数据吞吐速率估计会在 3000KB/s 左右。这样的结论是假定由放射性本底和噪声决定的触发率等于最大估计量时所做出的，并且假定了其他影响事例率的因素仅是由宇宙线μ 介子引起的，反中微子事例率忽略不记[3]。

2.2.2.1 事例率

在三个站点（大亚湾、岭澳以及远点）受各种源影响的触发事例率总结于表 2-2 中。

表 2-2 大亚湾实验触发率和数据量需求估算

Detector	Description	Trigger Rates (Hz) DB	LA	Far	Occ	Ch No.	data rate (kB/s)
$\bar{\nu}$ module	cosmic-μ	36×2	22×2	1.2×4	100%	192	1408
	Rad.	50×2	50×2	50×4			
Inner water shield	Rad & noise	50	50	50	10%	123/169	31
	cosmic-μ	250	160	15	70%		526
Outer water shield	Rad & noise	50	50	50	10%	168/212	41
	cosmic-μ	250	160	15	30%		309
RPC	Rad.	2000	2000	3000	10%	32/module	217
	cosmic-μ	186	117	11			11
site totals	(kB/s)	1042	788	714			2544

上表中给出了宇宙线μ 介子在内外水池探测器系统的总事例率。在远点的反中微子探测器的触发率主要由天然放射性决定（每个探测器小于50Hz），而在近点则主要由宇宙线和天然放射性两者共同影响。由 PMT 噪声和γ 射线引起的水池探测器中的触发率，能够通过触发条件限制到<=50Hz。RPC 的噪声率可以通过触发条件的选择压到很低，因此表 2-2 中所表示的信号率是来自不同站点 RPC 系统的天然放射率。

每个站点的反中微子探测器的触发率估计约为 100Hz，3 个μ 子系统（内外水池、RPC）可产生最大 1 kHz 的触发率。因此大亚湾触发和数据获取系统需要设计到能够承受最大触发率为 1kHz。此外，为了获取反中微子探测器中相互关联的一组中微子和快中子的信号，数据获取系统需要能够能获取间隔大于 300 纳秒的事例。

2.2.2.2 系统带宽

表 2-2 概括了每个站点中每个探测器系统的触发率，通道数，数据吞吐量。假设每个 PMT 通道所需要的最大数据块是 14bytes, 具体估计如下：

- 地址： 8 bits
- 时间：TDC+时间标志：32bits
- 电荷：6 个模数转换采样，每个采样 12bits，总共是 72bits。
- 对于 RPC 读出，我们假设一个通道 1bits+头信息（64bits）+时间标记（64bits）。

在估算中，内外层水切伦科夫探测器将采用零压缩，如表 2-2 中所示。

而反中微子探测器触发后将读出所有路 PMT 数据。预期的来自每个站点的数据吞吐量要综合考虑读出通道的数目，触发率和占空比，如表 2-2 所示，数据包还将包括一个长度为 128bit 头信息，头信息包括时间标志和触发类型。根据以上信息可以估算出，每个站点的数据吞吐量估计大约是< 1.5 Mbytes/second。

2.2.2.3 死时间

实验要求数据获取系统带来的死时间必须很小（<0.5%），可以忽略不

计。这就要求数据获取系统具有高速的数据传输能力以及多级内存缓冲区。在每个数据获取层都需要有一个缓冲区用来控制不规则的数据率，以避免由于数据率的短期波动而带来系统的死时间。

2.3 系统硬件架构

大亚湾实验的数据获取系统是由高级商业计算机和网络技术设备构建的多层系统，采用了分布式的并行计算机系统完成实验数据的获取。如图2-1所示，系统由多台处理器构成，各个处理器分别完成相对独立的任务，可以并发地工作。

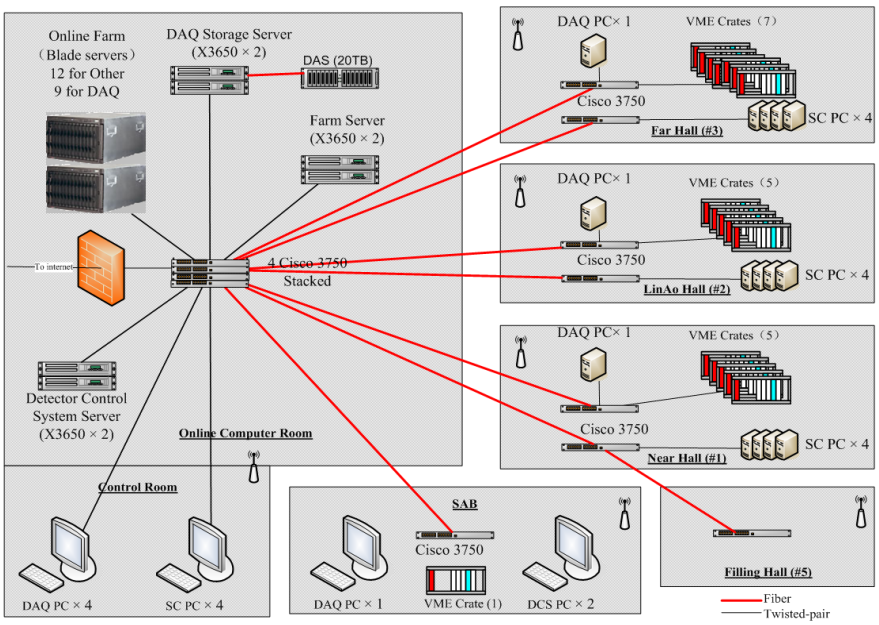


图 2-1 大亚湾数据获取系统总体框图

大亚湾数据获取系统的硬件框架设计如图 2-2 所示。整个系统设计由三套数据获取系统组成，每个站点各有一套硬件独立的且功能完整的数据获取系统。

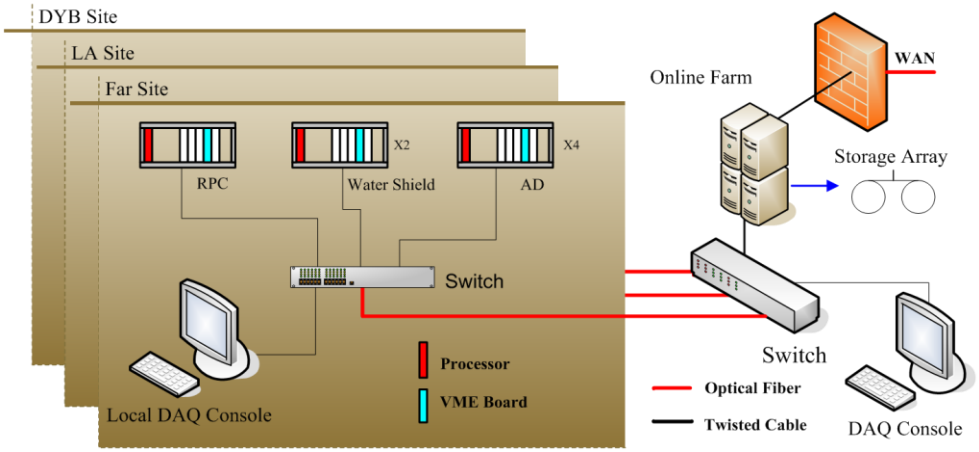


图 2-2 大亚湾实验数据获取系统框架

从硬件结构以及功能的划分考虑可以概括的将大亚湾实验的数据获取系统分为 VME 前端读出、DAQ 控制、数据流传输与数据存储三个主要部分。

2.3.1 VME 前端

数据获取系统的最低一级是基于 VME 的前端读出系统。每一个 VME 机箱对应一个反中微子探测器或者一个μ 子探测器，负责接收相应探测器的数据。每个 VME 机箱内除了必须插有一块系统控制器之外，还可能插有电子学读出板（FEE、ROM），触发板（MTB、LTB、RTM），FlashADC 板以及信号扇出板等，每个 VME 机箱最多可容纳 20 块电子学插件，所有的 VME 读出插件板都是 9U 标准设计的。

远点和近点探测厅都将使用同样的系统构架，但有不同数量的 VME 读出机箱，以配合在远/近厅不同数量的读出通道。VME 机箱的处理器是一块嵌入式单板计算机，用来收集、预处理并传送数据。每个站点所有探测器的读出 VME 机箱通过异步以太网连接到对应的本地控制计算机。

2.3.2 DAQ 控制

每个站点都有一台 DAQ 的本地控制终端，可以控制本站点所有探测器的运行状态。在地面装配大厅还有一台控制 PC，可以远程控制三个站点的运行状态。运行时，控制终端的用户通过 DAQ 的用户界面可以对取数进行

配置以及控制。

2.3.3 数据流传输与数据存储

大亚湾实验数据获取系统的在线机群由九台刀片服务器与两台文件服务器组成，数据从前端 VME 机箱读出后，数据流都是独立的，与探测器一一对应，数据流将在在线机群实现合并。在线机群还负责如直方图显示等数据流监测的任务。

在大亚湾实验中，每天将会有大约几百 Gigabyte 的数据量需要被保存。目前的方案是：首先，本地磁盘将能够存储至少一个月的数据文件以防止网络故障；同时，数据将会通过高速网络直接从大亚湾实验站发送到高能所。

2.4 系统软件框架

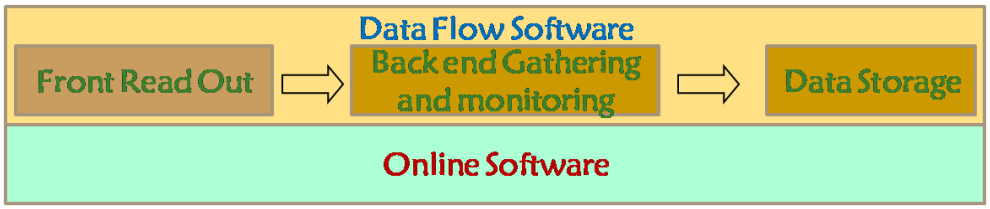


图 2-3 数据获取系统软件框架

大亚湾实验的数据获取系统采用分布式并行计算系统，用特定的软件负责计算机群的管理，如节点之间的通讯，消息的同步，和负载平衡等[10, 11, 11]。大亚湾数据获取系统的软件框架是在 ATLAS TDAQ 和 BESIII DAQ 的基础上构建而成的。数据获取系统总体软件框架可以分成两个部分，如图 2-3 所示：一部分是在线软件（Online Software），它的主要组成有包括配置、控制、系统监测等为数据流传递提供服务的组件；另一部分是数据流软件（DataFlow Software），顾名思义它的主要功能就是负责数据的处理和传送。大亚湾数据获取系统的在线软件部分基本重用了 ATLAS TDAQ 的相关组件，而数据流部分在借鉴 ATLAS 和 BESIII DAQ 软件的基础上，根据大亚湾实验的具体要求设计完成。

2.4.1 数据流部分

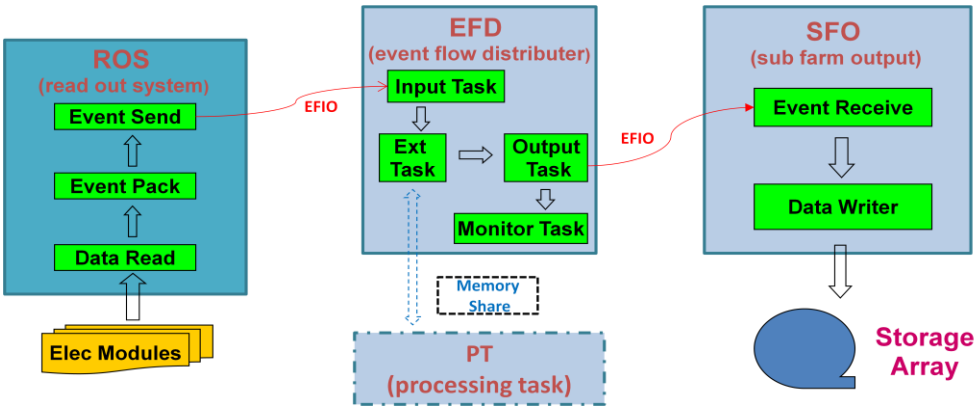


图 2-4 数据流结构图

数据获取系统的数据流部分的主要功能是进行数据的处理与传输。如图 2-4 所示，读出系统 ROS（readout system）接受来自电子学插件的数据，经过组装后，通过 EFIO 协议将事例发送给 EFD（eventflow distributer）组件，EFD 组件可以提供一些例如直方图显示等相关的数据流任务，同时 EFD 也通过 EFIO 协议将数据发送给 SFO（sub-farm output）进行存盘处理。这就是单个探测器进行取数时数据流的传递情况。

多探测器的情形如图 2-5 所示，从图中可以看到，大亚湾实验的三个站点可以有各自独立的数据获取系统，由不同的软件配置决定。三个站点共有 17 个 VME 读出机箱，9 个刀片服务器以及两台文件服务器。每个读出机箱上都有一套负责读出的 ROS 组件，每个站点也都有一套 SFO 组件负责文件的存盘。当然，图中只是软件部署框架的一种具体实现的情况，整个系统的数据流传递是有多种方案可供选择的，可以根据实验的具体要求灵活做出配置。

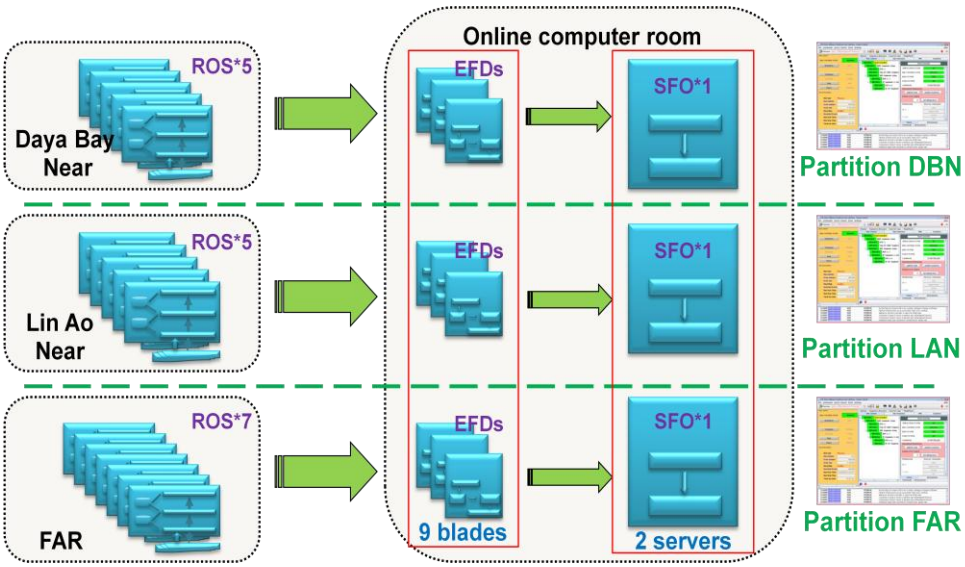


图 2-5 数据获取系统软件部署框图

2.4.2 在线软件部分

在线软件整合了整个数据获取系统的所有组件，是数据获取系统的结构基础。它提供了状态控制、配置数据库、分布式进程管理、信息服务、运行监测等服务，与数据流部分以及 DAQ 系统的外部用户都有定义良好的接口。在线软件可以看作是 DAQ 的管理者，它像胶水一样将不同的子系统粘合在一起，为 DAQ 系统的有序运行提供保障。在线软件的操作系统平台为 Linux，主要编程语言有 C++，C 以及 java，内部通讯基于 CORBA（Omni 实现）机制。

在线软件的架构基于组件模型，包含了三个高级的组件包。每个包都联系着一组在线软件的功能服务。在线部分的三个包是：配置数据库、控制以及监测，它们之间的交互如图 2-6 所示[12]。

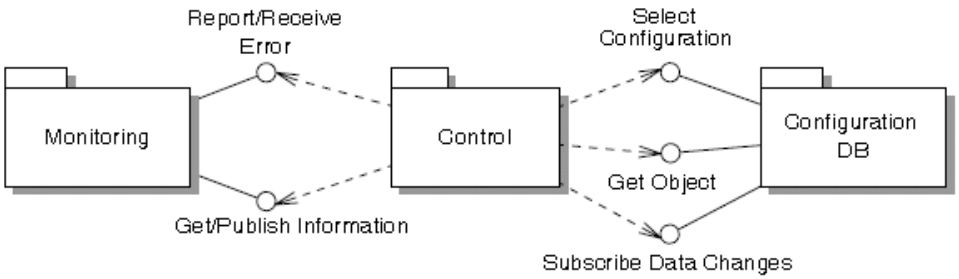


图 2-6 在线软件之间的交互

2.4.3 大亚湾 DAQ 主要组件及其功能

■ Readout System (ROS)

如图 2-7 所示，读出组件 ROS 主要负责配置前端电子学插件，读取电子学插件的数据片段进行事例组装，并且通过网络发送给 EFD 组件。ROS 组件是 DAQ 系统中唯一与前端电子学有接口的组件，是数据获取系统与电子学系统之间的桥梁。每个探测器系统都有一个对应的 ROS 读出系统。读出系统的设计与实现将在下面几章中有详细的介绍。

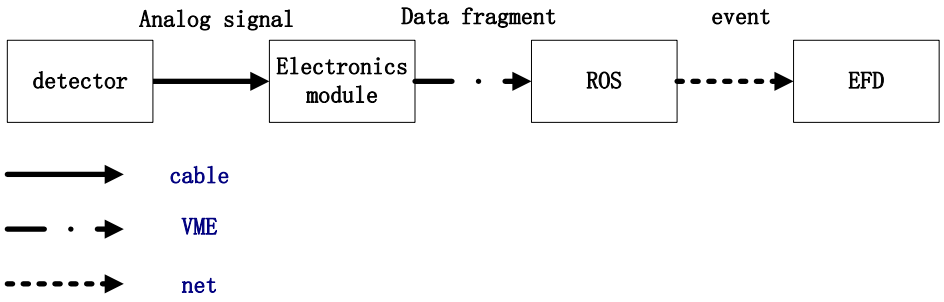


图 2-7 读出系统 ROS 功能示意图

■ Eventflow Distributer (EFD)

如图 2-8 所示，EFD 组件利用基于 TCP/IP 的 EFIO 协议[13]，负责接受 ROS 传送来的事例数据，并且发送给 SFO。EFIO 协议使用了握手机制，并且图中 1，2，3，4，5 标记处均设有超时（timeout），这样的方案保证了数据传输的过程中不会因为网络连接的失败而造成事例数据的缺失。

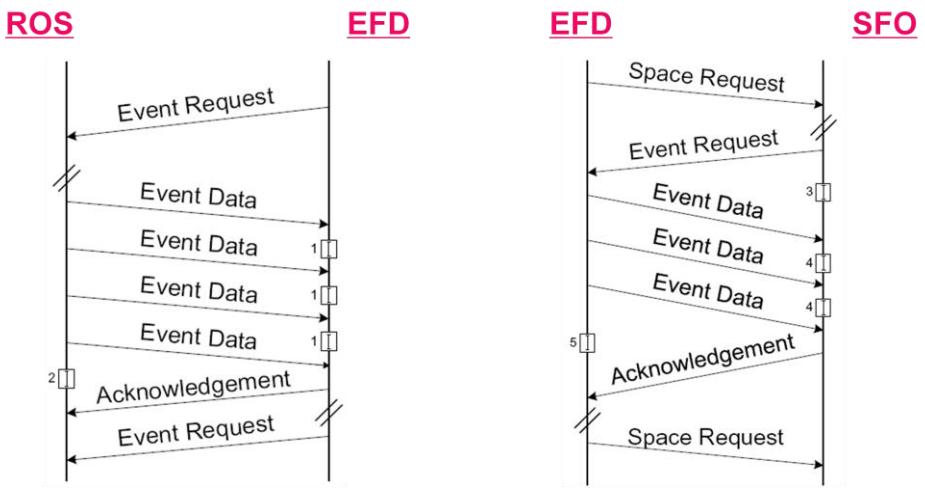


图 2-8 EFD 与 ROS、SFO 之间的消息传递时序图

■ 事例输出组件（SFO， SubFarmOutput）

SFO 组件主要负责事例数据的存盘。如图 2-8 所示，EFD 会向 SFO 发送空间请求（Space Request）的消息，当 SFO 准备好数据空间以后，会返回一个事例请求（Event Request），EFD 响应这一请求，将事例发送给 SFO，同时 EFD 会继续在内存中保留该事例直到收到来自 SFO 的事例成功接收的确认应答（Acknowledgement）。

■ 运行控制（RC， Run Control）

RC 组件，即运行控制组件，是 DAQ 软件的重要组成部分。它为用户提供接口控制、管理数据会话，也为 DAQ 其他组件提供软件接口。通过这些接口，RC 可以在各个组件之间交换命令、状态和信息，协调各部分进程的步调。RC 被组织成一个树状的层次阶梯，其节点称为 Controller，图 2-9 展示了一个运行控制树的实例。Controller 运行在各个节点机上，控制其上的应用程序，它可以接收外部命令，发生相应的状态变化。当然 DAQ 内部组件发生的本地事件也会触发它，引起状态转移。RC 组件基于专家系统 (Expert System)，采用专家系统开发工具 CLIPS 实现状态机。

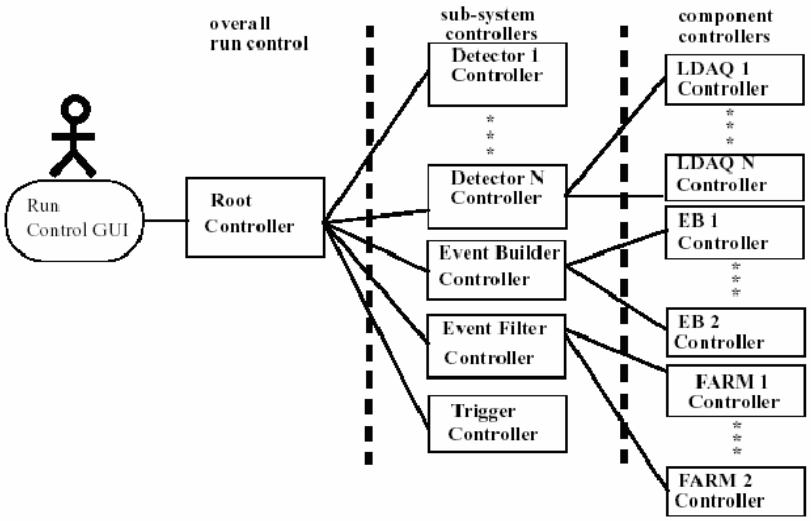


图 2-9 运行控制树实例

■ 进程管理（PMG， Process Manager）

如图 2-10 是 PMG 的体系结构图，PMG 是一个分布式体系，目的是对 DAQ 内部的进程进行基本的作业控制。它包含了三个元素：客户对象、代

理、动态数据库。PMG 基于 CORBA 通信机制，可以方便的通过代理(agent)控制各个目标节点机上应用程序的创建和撤销，并监控程序的基本状态（运行或已经退出）。

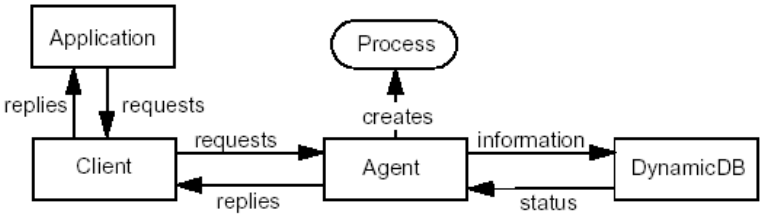


图 2-10 进程管理（PMG）体系结构

■ 信息服务（IS， Information Service）

IS 在分布式系统的各个进程之间提供共享信息服务。采用信息提供者—信息仓库—信息订阅者—信息阅读者的方式，IS 被广泛的使用在数据获取系统的各个组件中。例如用户可以通过 IS 了解到事例组装速度、当前组装好的事例总数等各种统计信息，可以查阅各类直方图和部分事例数据，对于监测数据流状态起到重要作用。

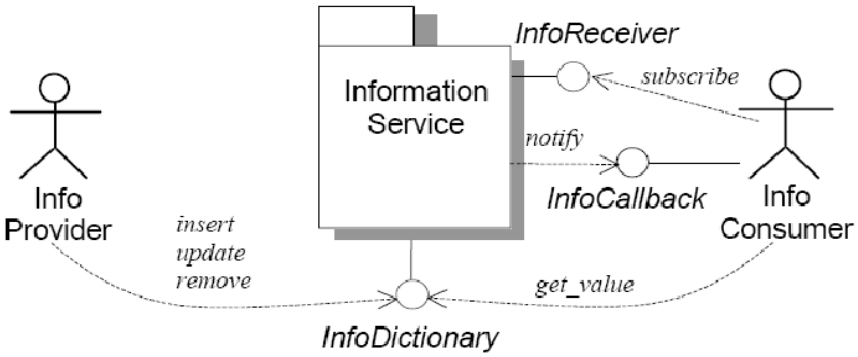


图 2-11 信息服务组件接口

如图 2-11 所示为信息服务组件的接口，信息的提供者使用 InfoDictionary 接口插入、更新和删除信息。信息用户有两种访问 IS 中信息的方法，一种是主动的使用 InfoDictionary 接口的 get_value 方法，另一种是使用 InfoReceiver 接口的 subscribe 方法，同时提供回调函数 InfoCallback 接口，等待信息更新后发送过来[14]。

■ 消息报告系统（MRS， Message Reporting System)

MRS 组件用来提供给 DAQ 各个组件报告和接收消息以及错误信息的手段。MRS 会负责运送、分发以及过滤这些信息。各应用程序可以灵活的定义自己发送与接收的消息集。

消息报告组件接口如图 2-12 所示，任何错误提供者都可以使用 MRSStream 接口发送错误消息给 MRS 的服务器 mrs-server；错误消息用户为了接受错误消息，首先需要通过 MRSReceiver 接口订阅（subscribe），并提交回调（MRSCallback）函数，然后在有错误消息产生时，mrs-server 就会将适合的消息转发给订阅者[14]。

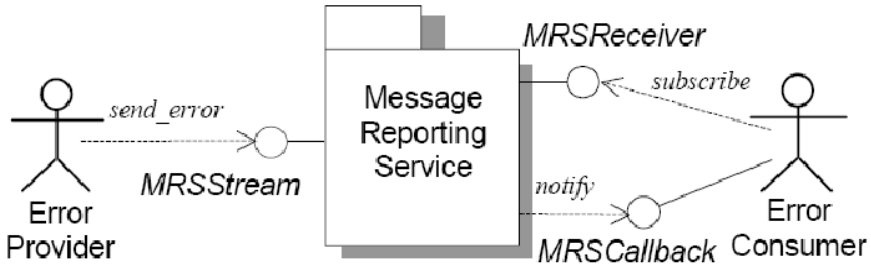


图 2-12 消息报告组件接口

■ 用户界面(Integrated Graphical User Interface, IGUI)

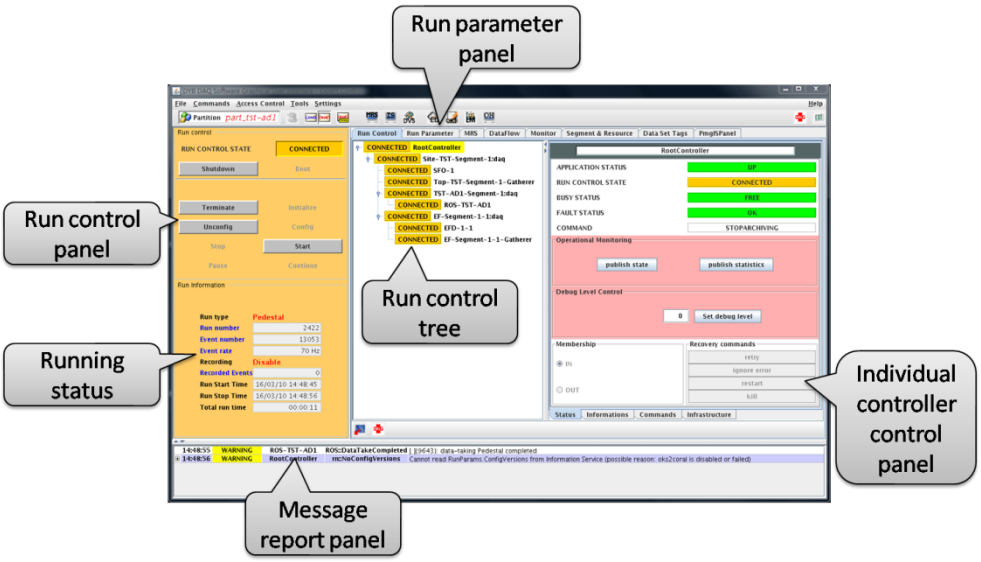


图 2-13 DAQ 用户界面的主要功能组成

大亚湾数据获取系统的用户界面如图 2-13 所示，界面上提供了运行模式选择、运行控制、运行参数设置、运行状态显示、数据流检测等等功能

按钮，基本上用户所需要的功能和信息都被集成在了这个用户界面中。

2.5 DAQ 数据文件格式

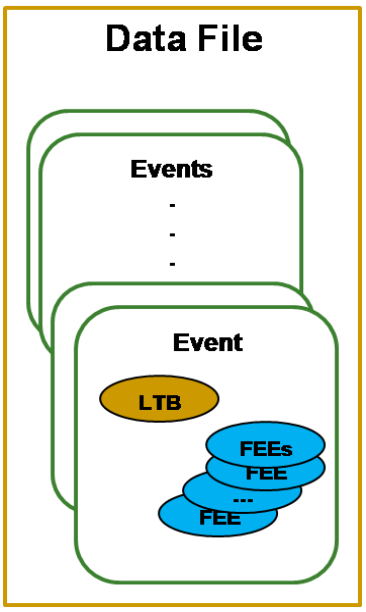


图 2-14 数据文件的三级结构示意图

大亚湾实验中，来自电子学插件的原始数据片段经过数据获取系统的读出与相应的组装处理，最终以文件形式记录在磁盘中。大亚湾数据获取系统记录的数据文件，可以分为三级结构，如图 2-14 所示。其中第一级即最低级的数据是来自前端电子学插件的读出数据，如图中的 LTB、FEE 等，文件中插件级的数据段包括了原始数据块与描述插件属性的头部信息。在插件级的上一级是事例级的数据，这一级的数据是来自同一个探测器内的所有插件级的数据的集合，这一集的组装是在读出系统中完成的，在组合所有读出插件数据的基础上，添加了相应的包含了探测器与事例属性的头部信息。最终一级的结构便是数据文件，一个数据文件包含了一系列的事例数据，并且在此基础上添加了相应的运行状态信息与必要的文件信息，这一级的组装是在 SFO 组件中完成的。

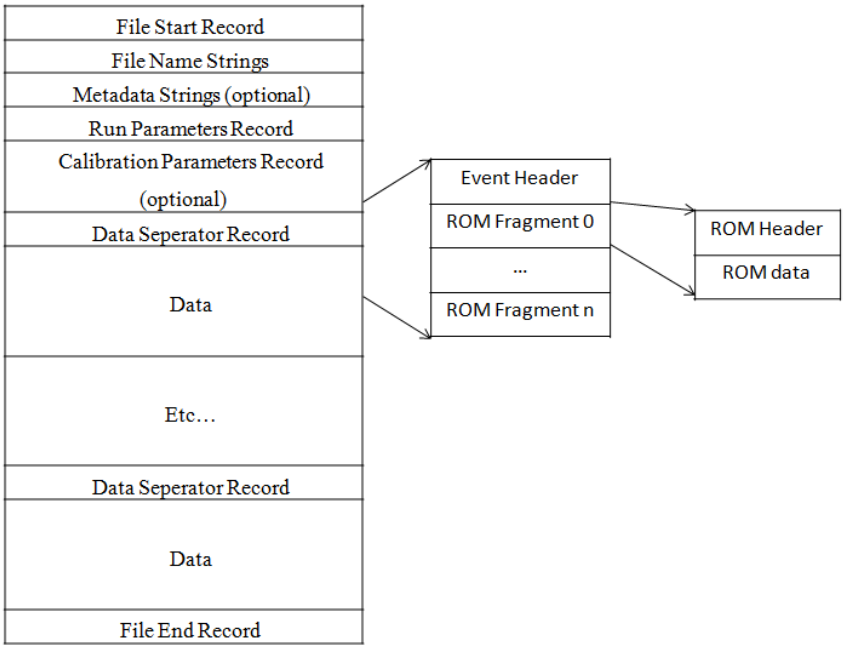


图 2-15 大亚湾 DAQ 数据文件格式

如图 2-15 所示是大亚湾实验 DAQ 数据文件的格式[15]，所有电子学读出数据片经过 DAQ 系统的读出组装后，最终存盘的数据文件都必须遵循这一格式，具体实现过程将在 5.4 节中进行详细介绍。

2.6 系统部署

2.6.1 系统状态转换

大亚湾实验的数据获取系统有着严格的状态控制，在某个状态下允许和禁止哪些操作都是有严格规定的，状态之间迁徙的细节都有明确的定义，DAQ 各组件之间必须同步的进行状态的转换。

图 2-16 是大亚湾数据获取系统的基本状态图，这是一个面向 DAQ 系统用户（例如值班人员）的状态转换示意图。可以看到 DAQ 的基本状态有六种：NONE（空闲）、BOOTED（启动）、INITIAL（初始化）、CONNECTED（连接）、RUNNING（运行）、PAUSED（暂停），图中展示了这些状态之间最基本的跳转步骤与实现这些跳转所接收的命令（Command），这些命令都可以通过用户界面（IGUI）上的状态控制按钮通过 RC 发送给 DAQ 各组

件。

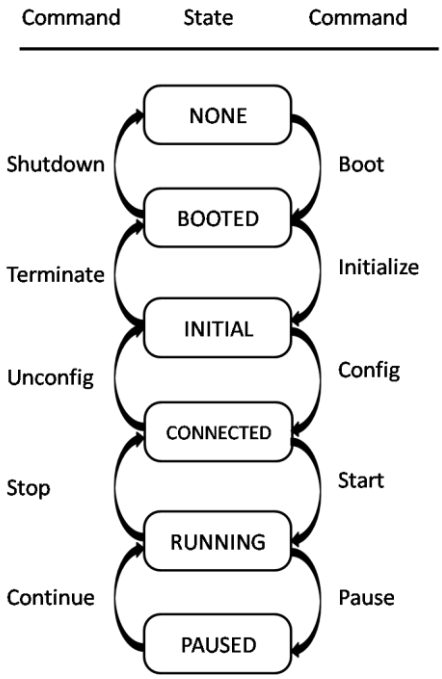


图 2-16 数据获取系统基本状态图

概括的说，当系统要开始一次运行时，首先 DAQ 系统会接收到 Boot 命令，开始进行系统启动的准备工作，例如启动进程管理组件 PMG。之后收到 Initialize 命令，PMG 负责将相关进程加载到内存中。接着会收到 Config 命令，此时所有组件进行配置以及相关的网络连接。在收到 Start 命令后便正式进入取数环节。Pause 和 Continue 的命令是为了提供暂停取数的功能，目前这个设计还是一个备选方案，以后将根据实际情况决定是否采用。当系统需要停止运行时，过程完全和之前相反，这里就不再仔细介绍了。需要注意的是，停止取数，并不一定要退回到 NONE 的状态，通常如果不改变运行模式，只需要 Stop 退回到 CONNECTED 状态，开始下一次取数时只需直接 Start 即可。如果下一次取数需要改变当前运行模式，则系统必须退回到 INITIAL 状态，通过用户界面更改运行模式后，系统进行重新配置后才能开始再次取数。

表 2-3 数据获取系统状态表

State / Command	NONE	BOOTED	INITIAL	CONFIGURED	CONNECTED	SFOSTOPPED	EFSTOPPED	EBSTOPPED	READY	PAUSED	RUNNING
Boot	BOOTED										
Initialize		INITIAL									
Configure			CONFIGURED								
Connect				CONNECTED							
Config			CONNECTED								
PrepareForRun					READY						
StartTrigger									RUNNING		
Start					RUNNING						
StopTrigger											READY
Pause											PAUSED
Continue										RUNNING	
StopEB								EBSTOPPED	EBSTOPPED		
StopEF							EFSTOPPED				
StopRecording						SFOSTOPPED					
Unconfig					INITIAL						
Stop									CONNECTED	CONNECTED	CONNECTED
Disconnect					CONFIGURED						
Unconfigure				INITIAL	INITIAL						
Terminate			BOOTED								
Shutdown		NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE	NONE

除了以上提到的这些主要的状态和命令，在 DAQ 系统内部还定义了更

丰富的状态及对应的转化命令。如表 2-3 所示，该表中列举了 DAQ 系统中存在的所有状态（State）和命令（Command），以及它们之间的关系。例如在状态 INITIAL 时，接收到 Configure 命令，系统便会进入 CONFIGURED 状态，而接收到 Shutdown 命令，系统便会进入 NONE 的状态。

表 2-3 中有一类特殊的命令：链命令（Chain Command），一条链命令其实是一组元命令的集合，当系统处于合适的状态下，接收到一条链命令，就相当于依次接收了一组相应的命令。例如系统在 CONNECTED 状态下接收到了 Start 链命令，即相当于依次收到了 PrepareForRun 命令与 StartTrigger 命令，系统也相应的先转入 READY 状态，再转为 RUNNING 状态。表 2-4 中列举了系统中定义的链命令和相关的状态：

表 2-4 数据获取系统链命令

状态	链命令	元命令	目的状态
CONNECTED	Start	PrepareForRun StartTrigger	RUNNING
INITIAL	Config	Configure Connect	CONNECTED
CONNECTED	Unconfig	Unconnect Unconfigure	INITIAL
READY	Stop	StopEB StopEF StopRecording	CONNECTED
RUNNING	Stop	StopTrigger StopEB StopEF StopRecording	CONNECTED

2.6.2 Partition

Partition 是以取数为目的的数据获取系统的一个子集，它包含了 DAQ 系统的所有功能，是 DAQ 组件的一个功能完备集。各个 Partition 可以并行工作，同一个程序可以独立的运行在不同的 Partition 中而互不干扰。一般可以将探测器的各个子部分映射到不同的 Partition。具体的，一个 Partition 可以包含一个或者几个子探测器的前端部分以提供源数据完成数据获取。数据获取系统中存在着一个全局的 Partition，包含整个系统的取数任务。

2.6.3 Run

一个 Run 被定义为在一组稳定条件下的数据获取时段。当累积至一定

数据量或者到达某个预设的情况或者某个条件需要发生变化时，就要重新启动一个 Run 进行新一轮的数据获取。

每当启动一个新 Run 就要赋予实验数据一个 Run 号，与这个 Run 号相联系的数据是有着相同的条件的，这些条件都要被记录到数据库中去。Run 号在整个实验的生命周期中应该是唯一的，并且需要记录在数据中。

2.7 DAQ 与外界的交互

大亚湾实验的数据获取系统与实验中的其他系统有着密切的交互，包括

- 通过 VME 总线与电子学系统的交互。
- 以 NFS 方式为离线分析提供数据文件。
- 通过在线数据库为相关人员提供运行信息，包括 Run 号、运行配置、运行参数以及存盘信息等。
- 通过 DIM[48]与其他系统实时交换命令与状态信息，参与交换信息的系统有 AD 刻度系统、矿物油监测系统、水池刻度系统等。

2.8 本章小结

大亚湾实验的数据获取系统提供了将探测器数据从前端电子学读出，经过一系列处理后将事例数据传送到大容量存储器这一主要功能，另外还包括控制、监测整个实验运作的功能。本章介绍了大亚湾实验对数据获取系统提出的需求以及根据需求所制定的数据获取系统整体设计方案，包括软硬件的框架结构以及相关的系统部署等。

3 大亚湾实验数据获取读出系统的设计

所谓读出系统，在各高能物理实验的数据获取系统中有着各自不同的定义与功能范围的区分。在本文中，大亚湾实验数据获取系统的读出系统指的是在 2.4.3 节中介绍的数据获取系统框架中的读出系统 ROS。

3.1 读出系统任务

从上一章的介绍我们已经知道，读出系统 ROS 是数据获取系统的前端组件，也是 DAQ 系统中唯一与前端电子学系统直接交互的组件。所有的事例数据片段都经过了读出系统的组装后，再由读出系统发送给后端组件进行相应的处理与存储。前端电子学系统的配置、控制、以及工作模式的选择等，都是通过读出系统直接执行的。可以看出，读出系统的运行性能、正确性以及可靠性对于整个数据获取系统，甚至整个实验的成功都有着重要的影响。

读出系统的任务主要包括：

- 配置前端电子学插件，控制电子学插件的工作状态
- 通过 VME 总线从电子学插件读取事例数据片段
- 检查数据的格式，对数据的正确性做出初步判选
- 按照指定格式将数据片段组装成完整事例
- 将事例数据发送给后端接收组件 EFD
- 监测数据流的情况，将相关内容发送给 IS 组件
- 及时地将状态信息与报错消息发送给 MRS 组件

3.2 读出系统性能需求

在读出系统一级，普遍采用以下手段减少系统死时间[2]：

(1). 使用较快的总线，如 VME 总线。

(2). 使用多个控制器并行读出。由于数据是并行的读出，各个控制器所要读出的数据量减少，总体上可以获得更高的数据吞吐量。

(3). 多级缓冲存储器。

(4). 存储器直接访问 DMA（Direct Memory Access）或数据块传输 BLT（BlockTransfer）。DMA 方式可以减少数据传输中的 CPU 开销，获得更高的传输速度。

(5). 使用嵌入式实时系统。使用高性能的嵌入式实时计算机和操作系统，可以减短中断响应时间，提高响应速度。

读出系统的性能主要取决于读出机箱的数据通过能力，是VME总线数据读出、数据预处理与组装、数据网络发送三方面的综合能力的体现。

根据 2.2.2 节中介绍的大亚湾实验的数据率需求，数据获取系统必须设计到能够承受的最大触发率为 1KHz，三个站点的总体数据吞吐率预期大约在 3Mbytes/second。大亚湾实验中两个近厅各有 5 个 VME 读出机箱（2 个 AD、RPC 与内外水池各 1），远厅有 7 个 VME 读出机箱（4 个 AD、RPC 与内外水池各 1），根据表 2-2 可以计算出，AD 单机箱的最大数据吞吐率约为 0.23Mbytes/s，RPC 单机箱的最大数据吞吐率约为 0.09Mbytes/s，水池探测器单机箱的最大数据吞吐率约为 0.36 Mbytes/s。由于所有的读出机箱是并行工作且各自独立的将数据读出，只要每个机箱都能满足水池探测器的最大单机箱数据通过能力需求，也就能满足所有探测器机箱的需求，因此单个 VME 机箱从前端电子学插件读取数据的能力必须不能小于 0.36Mbytes/s。

需要指出的是，上面算出的探测器单机箱每秒通过的数据大小是指原始读出数据的大小，经过读出系统的事例组装，在此基础上还要添加相关的插件头信息以及事例头信息才构成完整的事例通过网络发送给 DAQ 的后端组件，根据 2.5 节中 DAQ 数据格式的介绍，可以粗略的估计在事例组装过程中添加的头信息部分数据量约为小于 0.15Mbytes/s。因此为保证网络发送的顺利进行，单个 VME 机箱通过网络向 DAQ 后端组件发送数据的能力必须不小于 0.5Mbytes/s。

综上所述，大亚湾实验的数据率需求决定了读出系统的性能必须满足以下条件：

- 读出机箱必须有 1000Hz 的事例处理能力，能够快速的响应前端电子学插件发送的中断信号，从而保证系统的实时性与确定性。
- 前端 VME 机箱的数据通过能力必须达到 0.5Mbytes/s，以满足所有探测器的需求。考虑到系统的抖动因素，DAQ 读出系统单机箱的读出能力至少要达到 0.75Mbytes/s，并且数据组装和网络发送的能力同样必须不低于 0.75Mbytes/s。

3.3 读出系统硬件环境

目前世界上各大高能物理实验的前端电子学系统普遍是基于VME总线设计，大亚湾实验各探测器的电子学系统也是基于VME总线标准设计，插件采用9U VME标准，安装于9U VME读出机箱中。这些读出机箱构成了读出系统的硬件平台。如图 3-1所示，数据获取读出系统设计为基于VME总线的计算机系统，在每个VME机箱中采用Motorola公司生产的基于PowerPC的嵌入式单板计算机MVME5500作为VME总线的系统控制器（System Controller）。

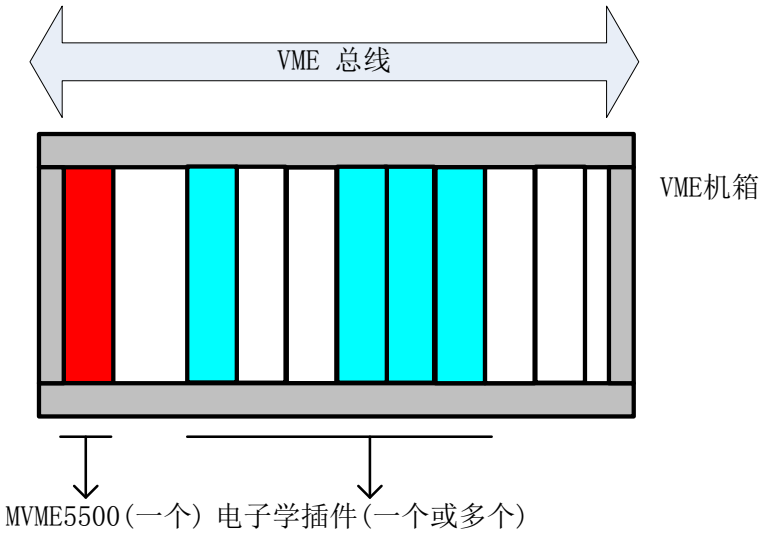


图 3-1 读出系统硬件结构示意图

整个读出子系统由 17 个 VME 机箱组成，每个机箱负责单个站点一个探测器系统的部分（AD、WP）或全部（RPC）的读出信息。机箱与机箱之间无直接连接，所有机箱通过 MVME5500 连接到千兆交换机，通过交换机与读出控制计算机通信。

图 3-2 说明了整个大亚湾数据获取系统的读出系统的结构。

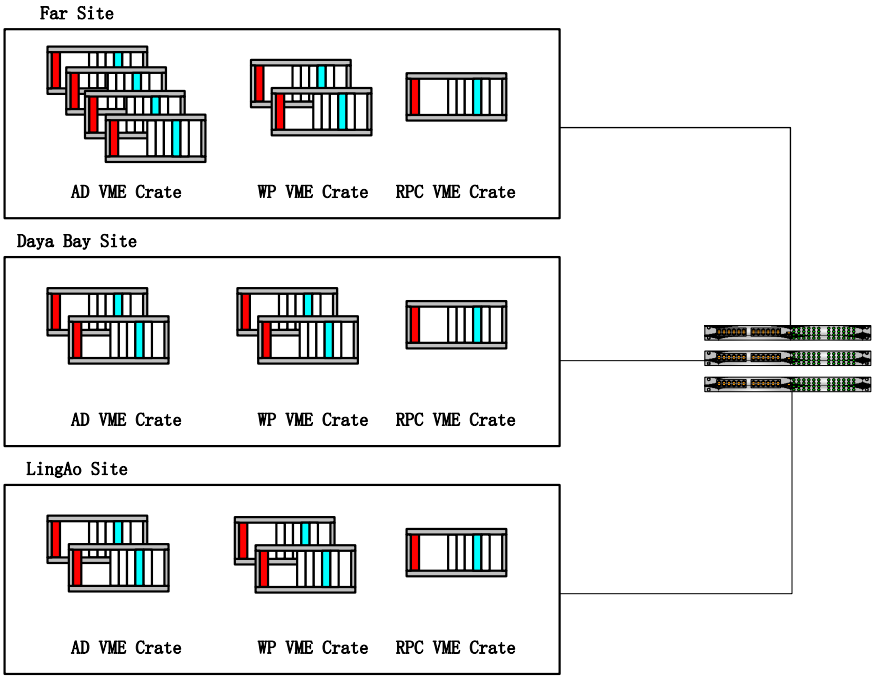


图 3-2 数据获取读出系统结构框图

3.3.1 VME 总线介绍

VME (VersaModule Eurocard)总线[16]是 I/O 总线的一种，它结合了 Motorola 公司 Versa 总线的电气标准和在欧洲建立的 Eurocard 标准的机械形状因子，是一种开放式架构。它定义了一个在紧密耦合（closely coupled）硬件构架中可进行互连数据处理、数据存储和连接外围控制器件的系统。经过多年的改造升级，VME 系统已经发展的非常完善，围绕其开发的产品遍及了工业控制、军用系统、航空航天、交通运输和医疗等领域。

VME 的数据传输机制是异步的，有多个总线周期，地址宽度是 16、24、32、40 或 64 位，数据线路的宽度是 8、16、24、32、64 位，系统可以动态选择。它的数据传输方式为异步方式，因此只受制于信号交换协议，而不

依赖于系统时钟；此外，还有 UnalignedData 传输能力，误差纠正能力和自我诊断能力，用户可以定义 I/O 端口；其配有 21 个插卡插槽和多个背板。传统的 VME 规范下，背板总线的传输速率达到 40Mbyte/sec，随着 VME 总线的发展，在 1997 年，使用 2eSST 协议的 VME320 背板达到了 320Mbyte/sec 的传输速率。

VME 提供了灵活的多主总线机制。一个 VME 系统可以有一个或者多个主设备。每个主设备都可以请求总线并且进行数据传输。其他的不具备称为主设备能力的插件被称做从设备。每个插进背板的插件都可以设计为主设备、从设备或者两者皆可。主设备具有请求总线的能力，而从设备没有，从设备只是监视地址总线，响应读或写的操作。

3.3.2 MVME5500 简介

VME 总线是一个共享资源，然而必须存在一个裁决机制来决定总线的使用权，这样的机制被称为仲裁，它是由 VME 总线系统控制器来执行的。系统控制器永远处于机箱第一槽。在 DAQ 读出系统中，MVME5500 就是这样的一个系统控制器。

MVME5500 是摩托罗拉公司的 VME 产品系列中的一个成员，它使用超过 1GHz 的处理器 MPC7455，适用于密集型的数据应用。MVME5500 的内部结构如图 3-4 所示，可以看到它的内部除了处理器 MPC7455 之外，还有一个 Tundra Universe II 控制器作为与 VME 总线的接口，一个千兆以太网接口、一个 10/100BaseTX 以太网接口，1GHz 的内存以及其他一系列与处理器相适应的高性能组件[17]。

图 3-3 MVME5500 外观图

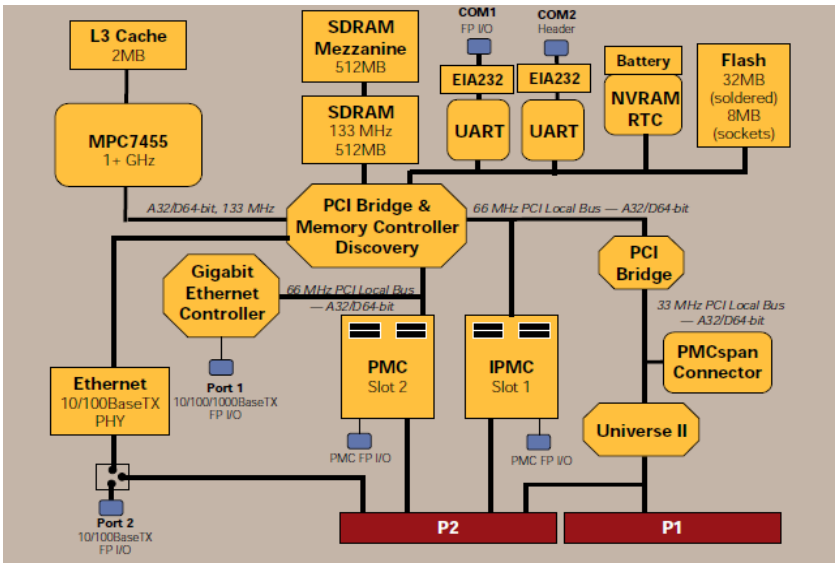


图 3-4 MVME5500 内部结构图

3.3.3 UNIVERSEII 芯片简介

UniverseII(CA91C142)是工业标准的 PCI 总线到 VME 总线的桥，换句话说，MVME5500 的 CPU 对 VME 总线的访问是通过这块 UniverseII 芯片来执行的。它是 MVME5500 的主要内部组件之一，提供了非常多的功能，以下只列出了一些重要的功能[18]：

- 64 位、33MHz 的 PCI 总线接口
- 高性能 64 位 VME 总线接口
- 双向的 FIFO 传输
- 支持链表的可编程 DMA 控制器
- 宽泛的 VME 总线地址和数据传输模式，。
- A32/A24/A16 的地址支持
- D64/D32/D16/D08 数据宽度支持
- MBLT, BLT, ADOH, RMW, LOCK, location monitors,
- 9 个用户可编程的 slave images
- 7 条中断线
- 灵活的寄存器设置

图 3-5 UniverseII 芯片外观图

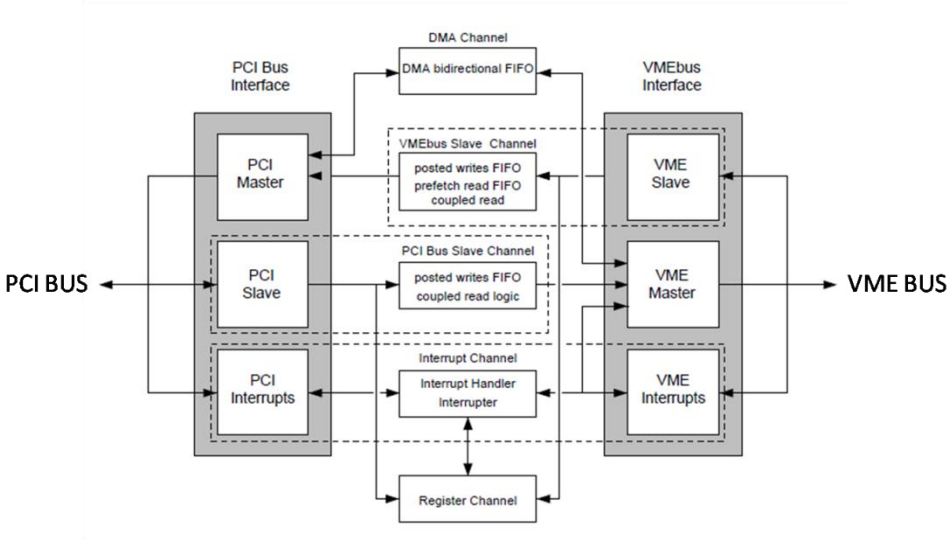


图 3-6 Universe II 芯片结构图

3.4 读出系统软件框架

读出系统的软件操作系统采用基于 Linux 的商业嵌入式实时操作系统 TimeSys Linuxlink，其内核版本为 Linux 2.6.9。通过对 Linux 内核的剪裁及驱动模块的加载，采用基于网络的根文件系统（NFS），构造嵌入式实时操作系统。

读出软件系统是一个实时系统，其主要任务包括：与 DAQ 主控计算机群通讯、配置电子学插件、数据采集、数据组装以及网络传输等。各任务间采用多线程并行的方式，线程间通讯采用消息队列和信号量机制。采用

状态图来实现系统的控制状态转换。线程间的数据传递采用环形缓冲技术。数据读出采用 CBLT（Chained Block Transfer）方式。网络传输采用 1000M 以太网，实现高速数据传输。

3.4.1 嵌入式实时操作系统 Linux

3.4.1.1 Linux 操作系统简介

Linux 操作系统以它的高效性和灵活性著称。Linux 是在 GNU 公共许可权限下免费获得的，是一个符合 POSIX 标准的操作系统。Linux 的主要特点有[19]：

- 代码公开
- 支持多用户
- 支持多任务
- 有良好的用户界面
- 具有设备独立性
- 具有丰富的网络功能
- 系统可靠性强
- 具备良好的可移植性

3.4.1.2 嵌入式 Linux 简介

嵌入式系统被定义为：以应用为中心、以计算机技术为基础、软硬件可剪裁、适应应用系统对功能、可靠性、成本、体积、功耗严格要求的专用计算机系统[20]。嵌入式 Linux 是指标准 Linux 发布包(Linux Distribution)在不同种类处理器上的应用[21]。

嵌入式 Linux 具有以下几点优势[22]：

- 嵌入式 Linux 系统是层次结构且内核完全开放，用户可以根据自身需求对内核进行裁减和定制，从而设计和开发真正满足需求的嵌入式系统
- 支持多种嵌入式硬件体系结构，如 X86, ARM, StrongARM, PPC, MIPS 等， 从而为用户提供了广阔的硬件选型空间

- 提供强大的网络支持功能。Linux 支持所有标准因特网协议，并且可以利用 Linux 的网络协议栈将其开发成为嵌入式的 TCP/IP 网络协议栈
- Linux 具备一整套工具链，容易自行建立嵌入式系统的开发环境和交叉运行环境，并且可以跨越嵌入式系统开发中仿真工具的障碍
- 成本优势。相对于私有操作系统收取高昂的软件使用版税的商业模式，嵌入式 Linux 的商业模式更注重于软件服务费用，从而有效的控制用户成本，这对于成本敏感的嵌入式设备而言尤为重要

3.4.1.3 Timesys Linuxlink 简介

LinuxLink 是 Timesys 公司开发的一个软件框架，为嵌入式 Linux 提供一套完整的解决方案，它可以针对具体需求量身打造 Linux 平台、工具集与应用软件。使用 LinuxLink 可以帮助缩短建立一个基于 Linux 的嵌入式产品的开发时间，减少开发风险与成本[23]。

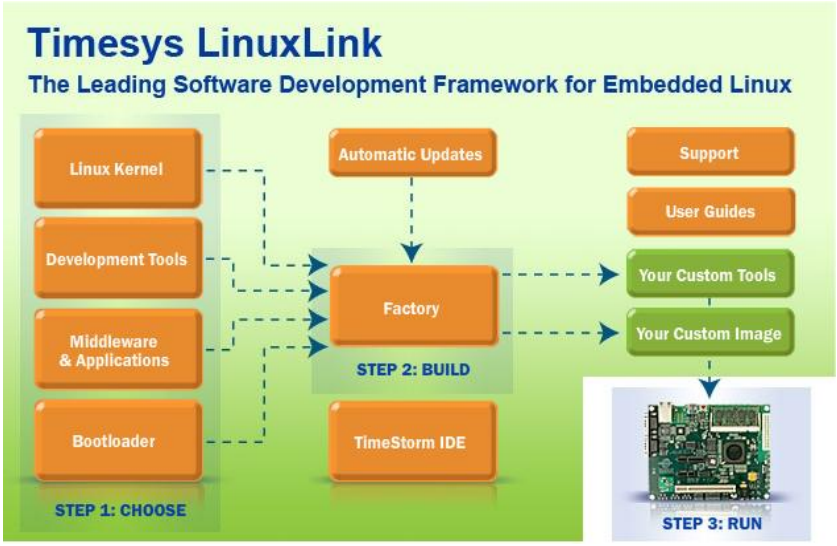


图 3-7 Timesys Linuxlink

LinuxLink 包含了嵌入式 Linux 的各种架构，最新的 GNU 工具链，多种可供选择的 Linux 平台，包含常用工具的工具集，说明文档以及技术支持等等。

3.4.2 VME 总线访问

3.4.2.1 VME Universe 驱动

在 3.3.3 节中已经介绍过，Tundra Universe II 芯片是 VME 总线与 PCI 总线的桥接芯片。LinuxLink 使用了 VMIC 公司为 Tundra Universe II 芯片所编写的 Linux 设备驱动，并且根据内核版本的需求做出了相应的少量修改。

VME Universe[24]驱动提供了能够与 Tundra Universe II 芯片交互的易用的软件框架。通常说来用户只需要使用驱动程序提供的接口编写自己的上层代码，而省去了和芯片直接交互的麻烦[25]。

3.4.2.2 单周期 VME 总线读写

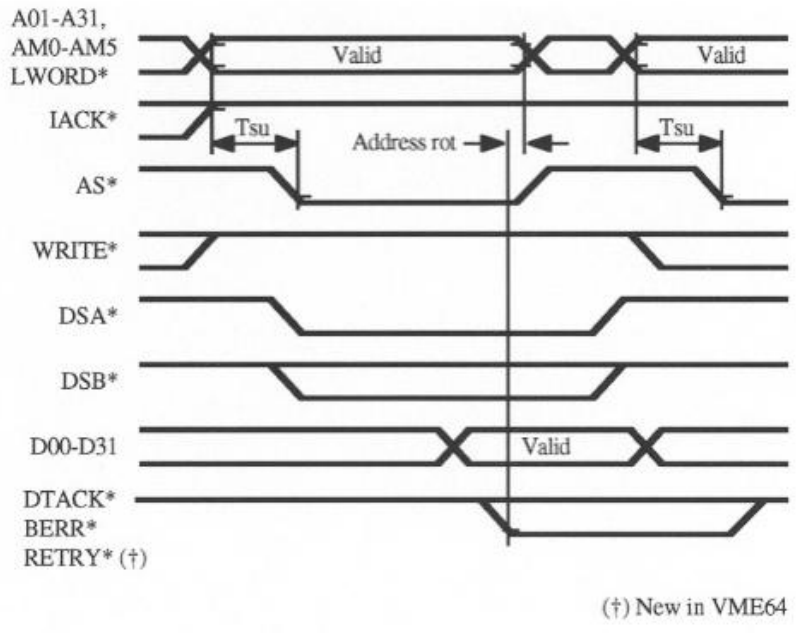


图 3-8 VME 总线读周期[26]

如图 3-8 所示，这是一个典型的 VME 总线的读周期的过程。写周期也是一个与此类似的过程。单周期的 VME 读写操作在数据获取读出系统中用于对前端电子学插件的配置以及状态访问等操作。在这些操作中，控制器 MVME5500 是 VME 总线上的主设备(Master)，而电子学插件是从设备(Slave)。

在读出系统中使用单周期总线读写的优点是实现简单，直接通过

VME-PCI 的映射关系便可完成，如图 3-9。这一做法的缺点是效率低，并且在进行读写操作时 CPU 完全被占用，因此不适宜传输大量数据块。

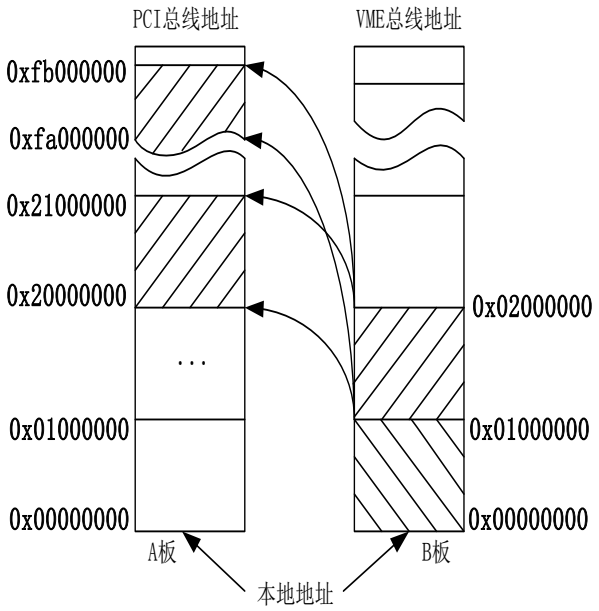


图 3-9 PCI 总线与 VME 总线的映射关系示意

3.4.2.3 DMA 传输

Universe II 芯片中存在一个 DMA 控制器，用来在 VME 总线与 PCI 总线之间进行高速数据传输，它的实现包括了一系列的寄存器的配置，包括数据源、目的地、传输长度以及传输协议等。DMA 的传输有两种方式，分别是 Direct 方式与 Linked List 方式。在 Direct 方式中，相关寄存器的值直接由外部的PCI主设备控制写入，它的工作流程如图 3-10所示；而在Linked List 方式中，相关寄存器会由 UniverseII 从 PCI 内存中自动加载，然后执行由这些寄存器所描述的传输，它的工作流程如图 3-11 所示。每次传输所需要配置的完整的一组 DMA 寄存器被称作一个 command packet，每个 command packet 可以链接到下一个，如图 3-12 所示。因此，当 DMA 完成一个 command packet 所描述的传输后，可以自动的跳转到链接的下一个 command packet 执行相应的传输。可以看到，Linked List 方式相对于 Direct 方式要更灵活高效[27]。

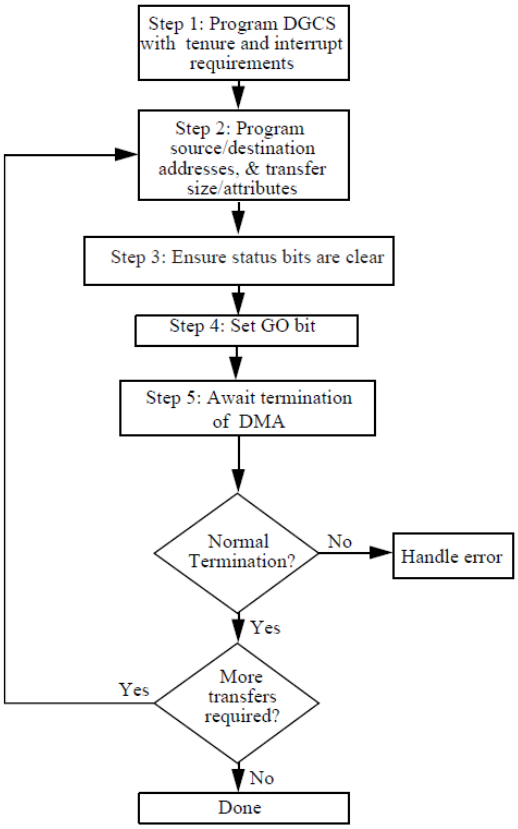


图 3-10 Direct Mode DMA 传输流程示意图

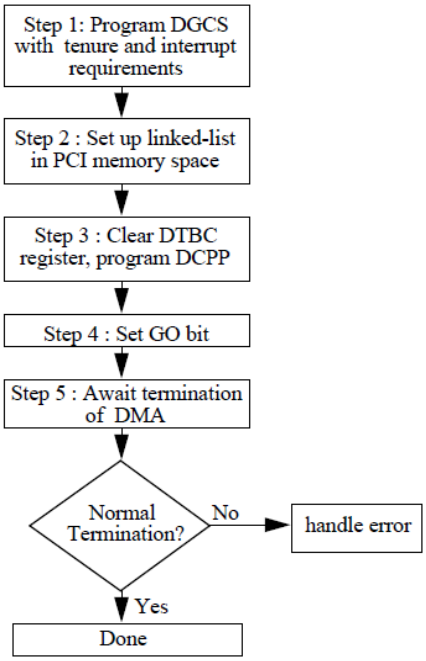


图 3-11 Linked List Mode DMA 传输流程示意图

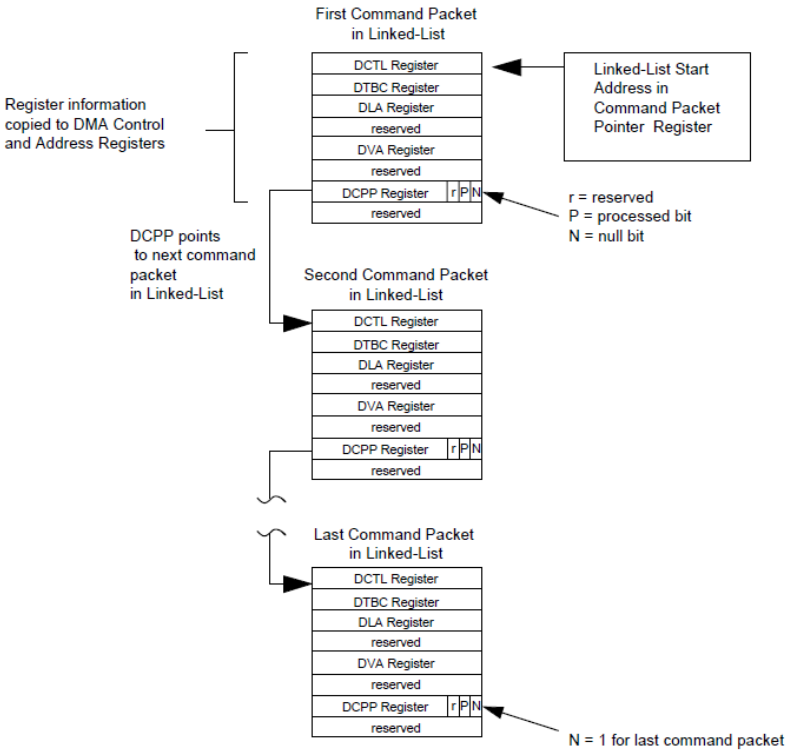


图 3-12 Command Packet 结构示意图

相对于单周期读写的方法，采用 DMA 的方式进行大量的数据传输时，能显著的提高传输速率，并明显的减少由于寻址和等待响应所耗费的 CPU 时间。然而 DMA 方式并不适用于高能物理实验的电子学读出系统。如果采用 Direct 方式的 DMA 传输，每个插件都需要启动一次 DMA，需要耗费系统控制器较多 CPU 时间，会严重影响系统的工作效率；而如果采用 Linked List 方式进行 DMA，需要提前设置好每一个数据块的大小，对于读出系统来说，每个事例的数据长度不固定，不可能提前指定好每个数据块的大小，因此这种方式在高能物理实验中也是不可行的。

3.4.2.4 CBLT 传输

CBLT(Chained BLock Transfer)，即链式块传输，是一种针对高能物理实验开发的数据传输方式[28]，它是一种特殊的 DMA 传输。在高能物理实验中，通常一个 VME 读出机箱中有多块电子学插件参与取数，只要启动一次 CBLT 就可以连续传输多个插件的数据块，传输过程中不需要软件的干

预，这种方式相对于启动多次普通 DMA 传输能够节省 CPU 的时间。CBLT 的原理是利用 VME 总线的 Daisy Chain Signals-IACKin/IACKout 传递令牌，令牌在各个参与传输的电子学插件之间逐个传输。也就是说，启动数据传输后，第一块读出插件自动获得令牌，在其数据传输完成后，将令牌传递给下一个插件,获得令牌的插件开始传输，如此传递下去，当最后一个插件的数据传输完成后会发出中断信号通知 CPU 本次传输完毕。

因此，CBLT 传输具有 DMA 传输所不具备的优点：首先，由于令牌之间的传递不需要系统控制器的干涉，因此除了传输启动前的寄存器设置，在整个传输的过程中系统控制器的 CPU 被释放，可以进行其它运算，从而实现真正意义上的多任务并行的系统；其次，每次传输之前不需要提前指定数据的长度，只需在传输结束后通过访问寄存器得到实际传输的字节数，这一特点能够很好的适应高能物理实验的需求。

3.4.2.5 中断响应

前端电子学数据的读出是由 VME 总线中断（interrupt）驱动的。VME 总线的中断总线提供七条中断请求线，一条中断应答线以及一条中断应答菊花链，如表 3-1 所示。

表 3-1 VME 总线中断的分类

Name	Description
IRQ1*	Interrupt Request 1
IRQ2*	Interrupt Request 2
IRQ3*	Interrupt Request 3
IRQ4*	Interrupt Request 4
IRQ5*	Interrupt Request 5
IRQ6*	Interrupt Request 6
IRQ7*	Interrupt Request 7
IACK*	Interrupt Acknowledge
IACKIN*/IACKOUT*	Interrupt Acknowledge Daisy-Chain

产生中断的设备利用拉低中断请求线来请求中断，这七条中断请求线优先级不同，以 IRQ1*的优先级最低，而 IRQ7*的优先级最高。中断应答线贯穿整个背板，并且链接到第一槽的 IACKIN*。一旦中断应答线被拉低，IACKIN*信号驱动第一槽的中断应答菊花链的控制器开始中断应答的菊花链传递，整个传递的过程如图 3-13 所示。

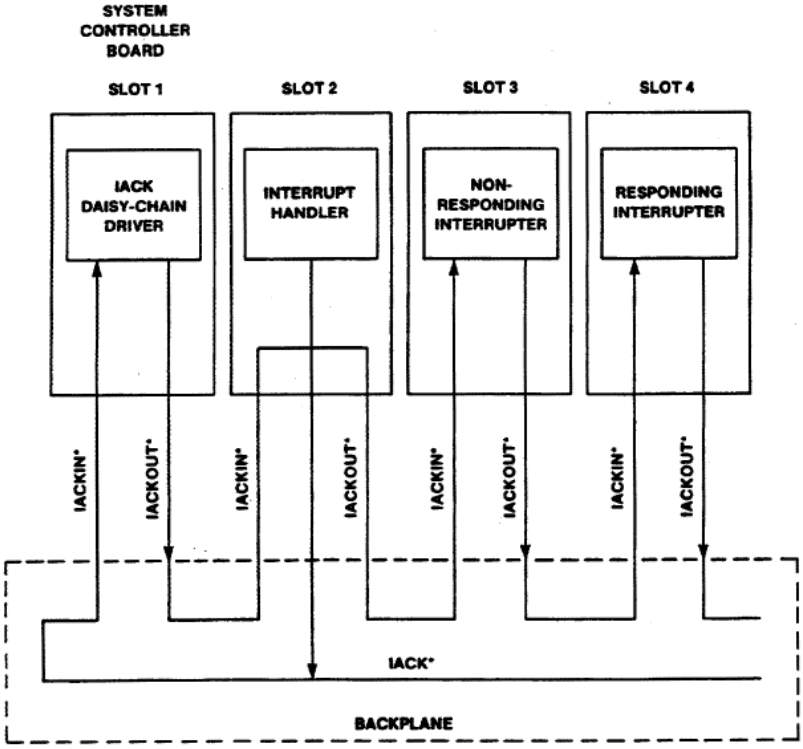


图 3-13 IACKIN*/IACKOUT*菊花链

3.4.3 多任务并行的读出系统

读出系统中存在一个应用程序控制线程，即 ac（application control）包，它提供了运行状态调度、数据库访问接口、程序监控等等功能，是其他高级线程的实现基础。

读出系统在取数周期中，在一个读出节点（读出机箱）内部，有四个数据流相关的并发执行的任务，每个任务都对应了一个独立的线程，分别是：

- ReadoutThread（CBLT 读出任务）：主要负责读出电子学插件的数据片段。

- DatapackThread（事例组装任务）：对原始数据片段进行初步的检验，并且将数据片段按照指定格式组装成完整事例数据。
- OutputThread（网络传输任务）：将组装好的完整事例数据发送给后端接收组件 EFD。
- MonitorThread（数据流监测任务）：该任务以指定频率更新读出系统的相关数据流信息，并提交给 IS 组件以供监测。

以上四个任务之间的关系如图 3-14 所示，CBLT 读出任务、事例组装任务与网络传输任务构成了读出系统的完整数据流，而数据流监测任务实时收集来自每个任务的相关信息，以便随时掌握数据流的状态。这四个任务在逻辑上是并发执行的，具体的实现机制将在 5.3 节中进行论述。

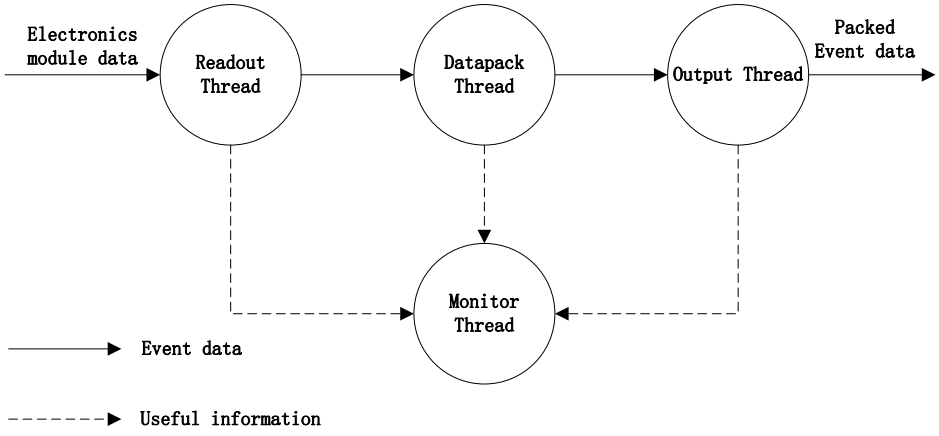


图 3-14 数据获取读出系统各任务间交互关系图

3.4.4 与其他系统的接口定义

读出系统的任务决定了它和电子学系统以及数据获取系统内部其他各组件都有着密切的接口，如图 3-15 所示。这些接口大致可以分为以下几类：

- 与前端电子学系统的交互接口，其中又主要包括了两方面：第一，读出系统与电子学插件的配置接口，即读出系统需要通过对电子学板上寄存器的操作配置电子学插件的工作状态；第二，数据接口，即读出系统需要读取电子学插件缓冲区中的数据。
- 状态机控制接口。读出系统在运行控制组件（RC）的整体控制下，实现状态机的跳转、完成相应的任务以及保持与 DAQ 系统其他组

件的同步。

- 与配置数据库的接口。读出系统需要读取并解析电子学配置数据库中的相关参数，以确保电子学系统能够工作在指定状态。
- 与后端数据流部分的接口。电子学插件提供的原始数据在读出系统经过事例组装后，需要通过网络协议将组装好的事例送给 DAQ 的后端系统进行存盘等后续处理，因此与后端组件的接口是必不可少的。
- 与信息服务（IS）以及消息报告组件（MRS）的接口。读出系统需要与信息服务组件交换相关的运行信息与重要参数，并且应当及时将系统运行中出现的各类警告或是错误信息提交给消息报告组件。

上述这些接口的具体实现方案将在第 5 章中有详细的描述。

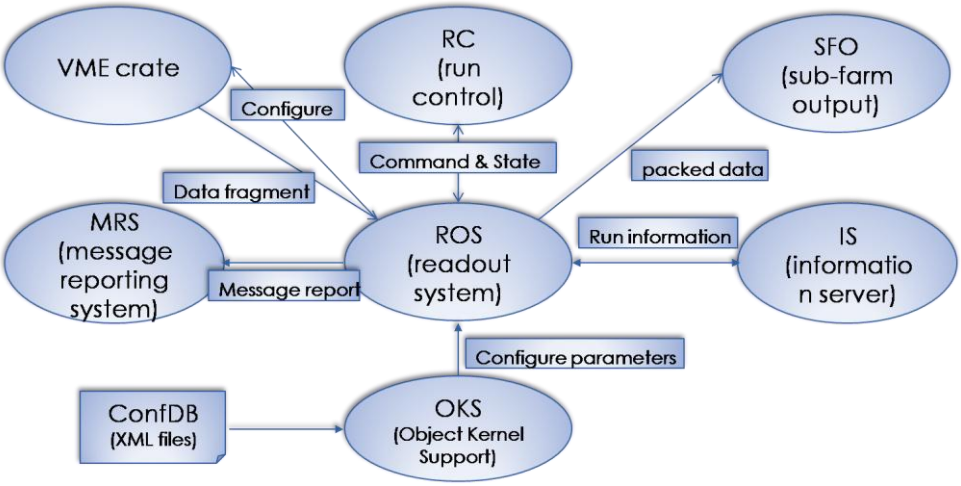


图 3-15 读出系统 ROS 和其他组件的接口

3.5 国内外高能物理实验中的读出系统

3.5.1 ATLAS 实验的读出系统

在 ATLAS 的 DAQ 系统中，执行数据获取前端功能的子系统由两部分组成：RCD（the ROD Crate DAQ）和 ROS（Readout System）。

3.5.1.1 RCD

RCD 提供了在探测器电子学和 DAQ 之间的完整的数据获取功能的框

架，RCD 由 Readout Driver （ROD），以及其它的一些功能部件组成[29]。在事例数据流（Event Data Flow）中，ROD 处于 Front-end Electronics 和 Readout System 之间，用以从 Front-end Link (FELs) 读取事例，并通过 Front-end Links 将其发送给 Readout Buffer (ROB)。每个 ROD 的实现跟特定的探测器有关，针对不同的探测器有不同的实现。在物理部署上，ROD 安装于 9U VME 机箱的电子学插件，一个机箱可能有多个 ROD 插件，这些插件由 ROD Crate Controller (RCC)管理。RCC 是一个单板计算机（Single Board Computer），运行 Linux 操作系统。

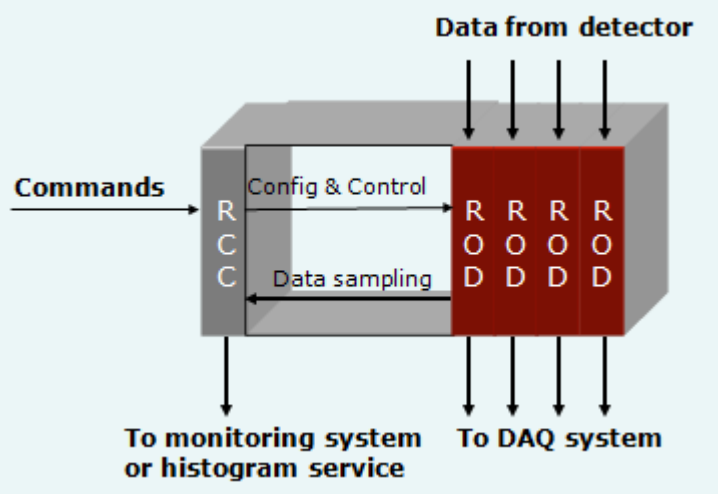


图 3-16 RCD Monitor 功能示意

在实验过程中，RCD 系统的主要功能有配置插件、模拟数据、数据监测（如图 3-16 所示）等。

3.5.1.2 ROS

Readout System[30]作为 Atlas 数据流系统中的一部分，从 ROD 接收并临时存储数据，为二级触发（Level 2 Trigger）和事例组装（Event Builder）提供可用的数据源。在硬件上，ROS 的实现采用了 PC 机群和千兆以太网适配器。PC 机群负责管理定制缓冲（ROBINs）。来自 ROD 的事例片段（Event fragments）首先通过 S-Link Readout Links (ROLs)进入 ROBINs。每个 ROBIN 有 3 个通道，每台 PC 上有 4 个 ROBIN。在 ROS 上临时存储的事例片段在二级触发和事例组装系统发出数据请求后，通过千兆以太

网被发送给二级触发和事例组装系统。在软件上，ROS 采用多线程的面向对象的框架，该框架的三个主要的插件为提供触发动作的 TriggerIn、提供数据输出的 DataOut 以及提供 ROS 和 ROL 之间接口的 ReadoutModule 插件。

3.5.1.3 对 ATLAS 实验读出系统的总结

Atlas 实验中，在 level-1 一级的触发事例率，按设计要求是 100 KHz。在 level-1 触发之后，每个读出连接的带宽为 160 Mbytes/s，总的读出带宽达到 160GBytes/s[10]。实验的数据量较大，对读出带宽要求非常高，因此在一些关键组件如 ROD、ROBIN 上采用了 FPGA 设计，用以满足高速数据读出的需求。这种方法对读出系统设计要求较高。

3.5.2 BESIII 实验的读出系统

BESIII DAQ 前端读出系统主要指 VME 机箱级的数据读出与运行控制部分。前端读出系统是基于 VME 总线的读出实现。BESIII 各子探测器电子学系统插件均基于 VME64x 标准总线，插件被设计成 9U 或 6U 高度，分别安装在 9U 和 6U VME 机箱中。每个 VME 机箱装有一块基于 VME 总线的系统控制器，与机箱内的电子学插件一起组成前端读出系统的一个读出单元。读出单元的系统控制器采用 Motorola 公司生产的基于 VME 总线的单板计算机 MVME5100。为保证前端读出系统的实时响应性能，MEME5100 上运行 WindRiver 公司开发的嵌入式实时操作系统 VxWorks5.4，并基于该操作系统运行 DAQ 前端读出系统软件。

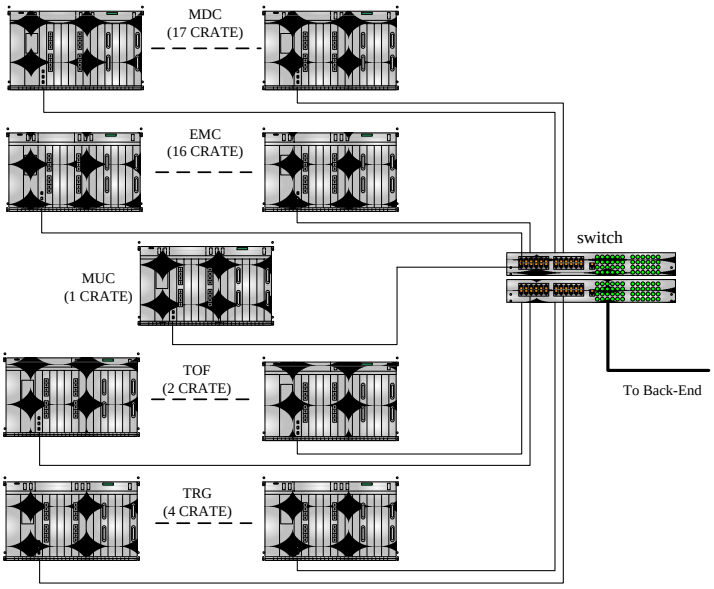


图 3-17 BESIII 读出系统整体组成结构示意

前端读出系统根据电子学部分所完成的功能可分为 5 个子系统，对应 BESIII 的五种探测器，如图 3-17 所示[35]。各读出子系统负责对应电子学的数据读出，各读出单元的系统控制器独立、并行的运行 VxWorks 操作系统和 DAQ 前端读出系统软件，但对于不同子系统所加载的软件模块有所不同。读出软件负责插件配置、VME 总线数据读出、组装与网络发送。

前端读出机箱的数据通过能力至少要达到 4Mbytes/sec，进而数据组装和网络发送速度也必须不低于 4Mbytes/sec。同时前端读出系统需要有 4000Hz 的高频率事例处理能力，必须能够快速响应前端电子学控制插件发来的中断，并保证实时性和确定性。

3.5.3 BELLE 实验的读出系统

位于 KEK 的 BELLE 实验，其设计目的是为了观察 CP 破坏以及给出 CKM 矩阵元素的精确测量值。根据实验的设计需要，DAQ 系统需要达到的事例率为 500Hz。

BELLE 实验的数据获取系统需要获取来自不同探测器的数据，因此根据读出系统的不同设计，将数据读出系统分为 TDC readout system 和 SVD(silicon vertex detector) flash ADC readout system[31]。

3.5.3.1 TDC readout system

TDC 读出系统负责读出除 SVD 探测器以外的所有探测器的数据。如图 3-18 所示，该部分的读出系统包含了两个部分：FASTBUS 与 VME。FASTBUS 处理器的接口 FPI 用来控制产生数据的 TDC 插件，并且 FPI 本身受到来自 VME 机箱的读出控制器的管理。VME 机箱中的系统控制插件采用了 MVME162，读出软件运行在实时操作系统 VxWorks 上[32]。读到的数据直接发送给同一个 VME 机箱内部的事例重建(Event Builder)发送器。

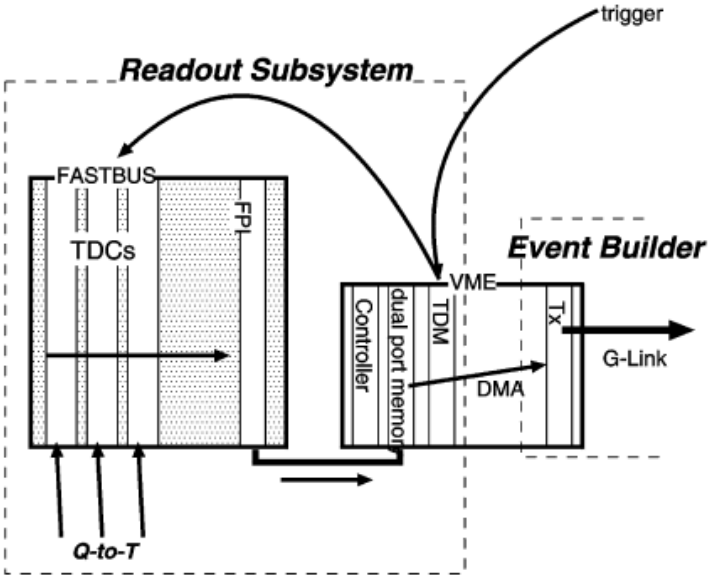


图 3-18 TDC readout system 结构示意图

3.5.3.2 SVD flash ADC readout system

SVD 探测器有着与其他探测器不同的读出系统结构，SVD 探测器需要读出 82000 路电子学通道的数据，相对应的事例大小约为 10~20KB，超出了单个事例重建发送器的承受范围。因此如图 3-19 所示，SVD 探测器的读出系统分割为四个 VME 机箱，四个读出机箱同时工作，并行地将数据发送给事例重建组件。SVD 读出系统的时序同步由单独的控制机箱负责。FADC 的数据使用低压差分信号（LVDS）进行传输，而以数据监测为目的的读出软件的运行环境为 SPARC VME CPU。

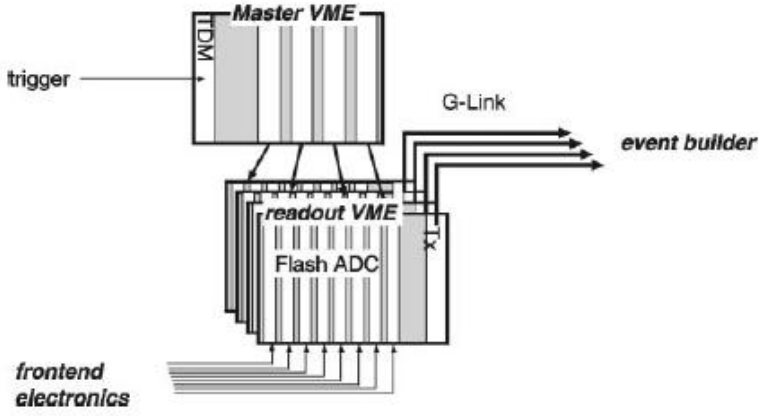


图 3-19 SVD readout system 结构示意图

3.5.3.3 对 BELLE 实验读出系统的总结

高能物理实验中的数据获取系统的设计归根结底是服务于实验的，而读出系统作为数据获取系统的前端，与电子学系统以及 DAQ 系统的后端组件都有着密切的接口。因此 BELLE 实验中的读出系统根据探测器电子学的设计，以及 DAQ 后端重建组件的需求，设计了两种相应的读出系统。

3.5.4 小结

本节介绍了 ATLAS、BESIII 以及 BELLE 实验中的数据获取读出系统的设计。通过对这几个读出系统的介绍，结合本章前面几节中对大亚湾数据获取读出系统的设计思想的描述，可以看出读出系统在高能物理实验中所起的重要作用。读出系统的设计应当以满足高能物理实验的具体需求为首要任务，在此基础上综合考虑造价、扩展性、人力等各方面因素，最终选取最为适宜的开发方案。

综合考虑国内外高能物理实验的读出系统，可以发现部分实验是直接采取了基于 PCI 总线的数据传输模式，而采用基于 VME 总线进行数据传输的实验中，较多的采用了 VxWorks 实时操作系统。大亚湾实验的读出系统创新性的使用嵌入式 Linux 操作系统作为读出平台，这一设计的合理性必须通过充分的实验来验证，在接下来的两章中，都将会对基于嵌入式 Linux 的大亚湾数据获取读出系统进行细致的测试与研究。

3.6 本章小结

本章阐述了大亚湾数据获取读出系统（ROS）的任务和性能需求，以及为了满足这一需求所进行的整体的设计考虑，详细描述了系统的硬件环境与软件框架的构建，并简要介绍了所涉及到的相关技术原理。最后结合国内外高能物理实验中的一些典型的读出系统的设计，提出了大亚湾实验的读出系统的特点与面临的挑战。

4 基于嵌入式 Linux 的读出性能研究

4.1 研究背景

在之前的高能物理实验中，通常采用成熟的商业软件作为读出系统的开发平台，如美国风河（WindRiver）公司开发的嵌入式实时操作系统 VxWorks[33]，它具备良好的持续发展能力、高性能的内核以及友好的用户开发环境，以良好的可靠性和卓越的实时性得到了广泛地应用，例如 BESIII 的数据获取系统便采用了 VxWorks 作为读出平台。

近年来 Linux 操作系统得到了巨大发展，也应用到了各个领域。嵌入式 Linux 的优势在上一章已经有过详细的介绍，但是 Linux 操作系统的实时性与稳定、可靠性以及功能的完善性相对成熟的商业软件还有一定差距，这就使得 Linux 在高能物理实验的数据获取读出系统中还罕有应用。因此要在大亚湾实验的数据获取读出系统中使用嵌入式 Linux 操作系统作为读出软件的开发平台，必须通过充分的测试来检验这一方案的可行性，以确保能够满足大亚湾实验的需求。

4.2 单项性能测试

首先通过一系列与读出系统密切相关的单项性能测试来检验基于嵌入式 Linux 的数据获取读出系统的性能指标。

4.2.1 测试环境介绍

4.2.1.1 硬件环境

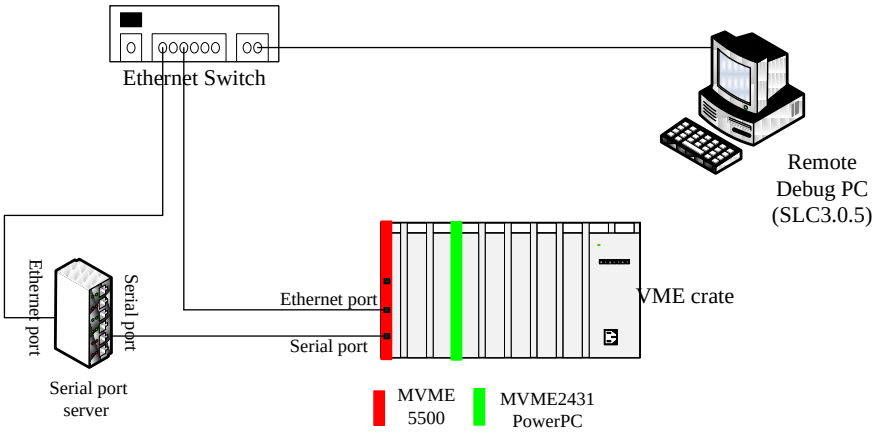


图 4-1 单项测试硬件环境

为了测试各项基本读出性能，搭建了如图 4-1 所示的测试平台。在一个 VME 机箱中，使用基于 Linux 操作系统的 MVME5500 作为系统控制器，用另一型号的单板计算机 MVME2431[34]模拟前端电子学插件，作为测试的中断源以及数据源。控制器 MVME5500 的网口通过千兆交换机连接调试主机。MVME5500 通过网络启动，通过串口进行调试（关于串口调试的具体实现，可以参见文献[35] 6.4 章节）。

4.2.1.2 软件平台

调试主机安装 SLC（Scientific Linux Cern）操作系统，内核版本 2.6.9，GCC 版本 3.4.6；MVME5500 安装 Timesys LinuxLink 产品，Linux 内核版本 2.6.9，GCC 版本 3.4.1，并且 LinuxLink 软件包中提供了 UniverseII 芯片的驱动 VME Universe。

4.2.2 单周期读写性能测试

单周期读写是最基本的 VME 总线数据传输方法，它的基本原理在 3.4.2.2 节中已经介绍了。本节中将测试基于嵌入式 Linux 的 VME 总线单周期读写的性能情况。

4.2.2.1 测试方法

测试流程如图 4-2 所示，为了降低误差，采用多次循环读写操作，利用总时间与循环次数来计算平均的读写时间。

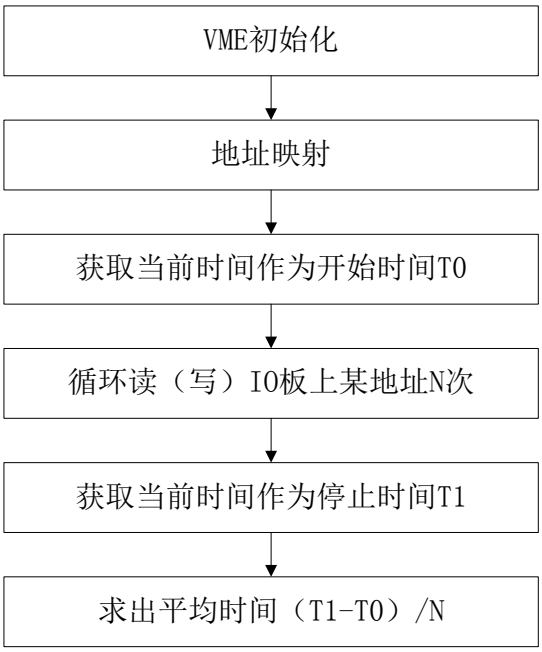


图 4-2 单周期总线读写时间测试程序流程图

4.2.2.2 测试结果

使用以上的测试方法得到的 VME 读周期的时间是 1638ns(一次读 32bit)，而 VME 写周期的时间是 405ns(一次写 32bit)。

从测试结果看，读操作比写操作耗费更多的时间。这是因为MVME5500的芯片 UniverseII 采用了 Posted Write 的方式来执行写操作，数据被写进 UniverseII 的 RXFIFO，而不是直接写入目的地址，因此测到的时间较短。而对于读操作，UniverseII 采用的是带握手机制的 Coupled Transfer 方式，意味着在这一操作中没有任何 FIFO 的介入。

根据上述结果，以32位的数据宽度进行一次读取操作的时间是1638ns，那么每秒钟传输的数据量约为 2.4Mbytes，并且如果使用这种方式进行数据传输，CPU 完全被占用，而不能进行其他的运算任务，这就限制了单周期总线传输方式在实验中的应用。在大亚湾实验的读出系统中，单周期的总线传输方式只是应用于对电子学插件的配置、控制以及对状态寄存器的访

问等，这些操作一般在正式取数阶段之前或者之后进行，因此不会占用 CPU 太久而影响正常取数的进行。

4.2.3 DMA 传输性能测试

4.2.3.1 测试方法

测试使用的流程和单周期读写的流程类似，如图 4-3 所示，也是采用多次循环传输来计算平均传输时间，以降低误差。大亚湾实验计划使用 32 位数据传输，因此测试中也采用了 32 位 DMA 传输方法。

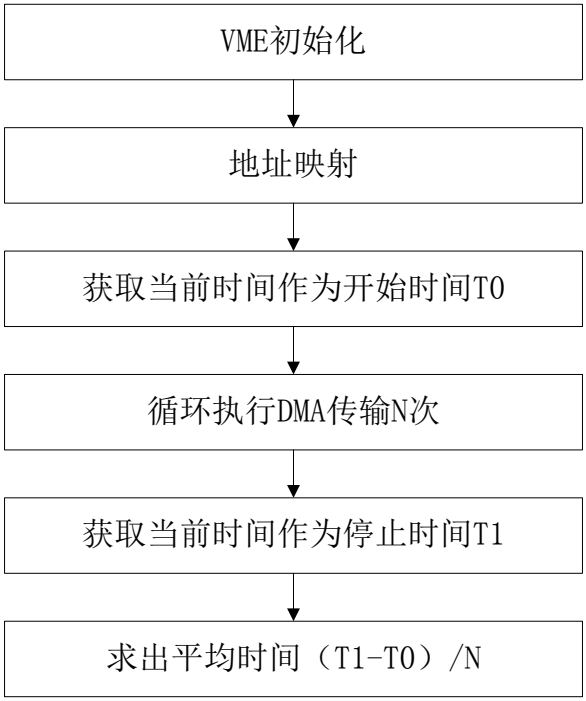


图 4-3 DMA 传输时间测试程序流程图

4.2.3.2 测试结果

表 4-1 32 位 DMA 传输测试结果

数据块大小 (bytes)	10000 次传输总时间(us)	传输速率 (bytes/us)	传输单字节所需时间(us/byte)
1	149649	0.067	14.96
2	150810	0.13	7.54
4	151542	0.26	3.79
8	154191	0.52	1.92
16	158888	1.01	0.99
32	166894	1.92	0.52
64	186433	3.43	0.29
128	222298	5.76	0.17
256	296184	8.64	0.12
512	410860	12.46	0.080
1024	683269	14.99	0.067
2048	1186314	17.26	0.058
4096	2224180	18.42	0.054
5120	2793119	18.33	0.055
6144	3253977	18.88	0.053
7168	3784294	18.94	0.053
8192	4287968	19.10	0.052
16384	8423475	19.45	0.051
32768	16696669	19.63	0.051

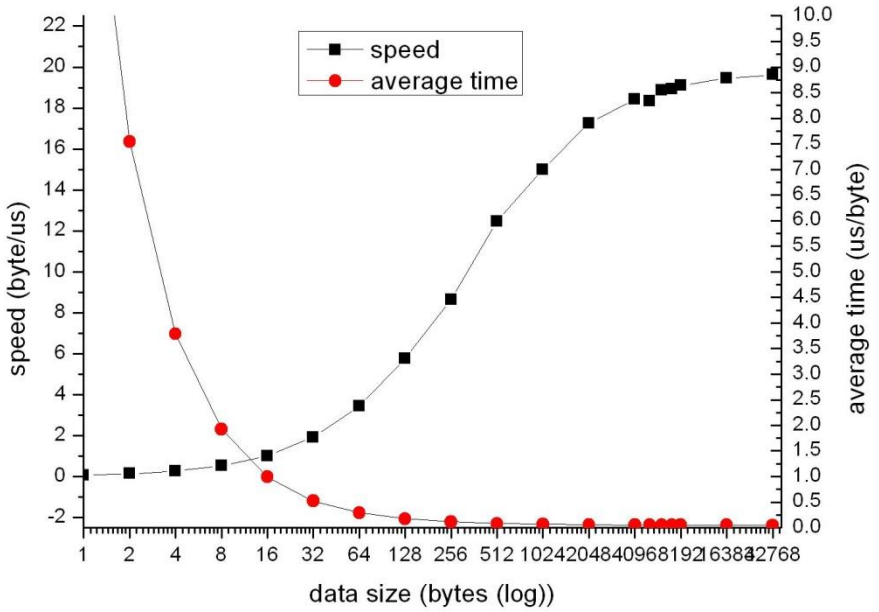


图 4-4 32 位 DMA 传输速率与传输每字节的平均时间随传输数据块的变化情况

通过测试传输不同大小的数据块的传输时间算出传输平均速率以及传输单个字节所需要的时间，测试结果如表 4-1 所示。通过图 4-4 可以更直观的观察 DMA 传输速率随传输数据块大小的变化趋势。从表 4-1 中可以看到，当采用 32 位的 DMA 传输方式时，每次传输 2048 字节约耗时 118us，每秒的传输速率是 17.26Mbytes；如果传输 32768 字节的数据块，传输耗时约 1669us，每秒的传输速率可达 19.63Mbytes。

采用 DMA 传输的另一个优点是，在 DMA 传输的过程中是不占用 CPU 资源的。每次 DMA 传输时，CPU 运算只包括发生在传输数据之前的寄存器设置，以及数据传输完成后响应中断，在数据传输期间 CPU 是被释放的，所以在进行数据传输时可以同时进行其他的运算任务。DMA 传输时 CPU 占用时间与被传输数据块大小无关，由此可以将一次 DMA 传输 4 字节时所花的总时间近似看作每次 DMA 传输的 CPU 占用时间，从表 4-1 可以看出，这个时间大约为 15us。也就是说，无论每次传输的数据量是多少，这个 15us 的基本开销都是不可避免的，这就解释了为什么 DMA 方法在传输大数据块时性能优势会更加突出。

通常高能物理实验的数据获取读出系统会采用 CBLT 方式读取数据。CBLT 方式本质上说就是一种优化后的 DMA 传输，详细实现机制请参加 3.4.2.4 章节。

4.2.4 中断开销测试

在大亚湾实验的 PMT 电子学读出系统中，当指定数目的事例到达后，触发板（LTB）会向 VME 总线发出中断请求，读出系统一旦响应到这个请求，便标志着可以开始本次 CBLT 取数过程。并且每次当 CBLT 传输结束时，也会通过 VME 总线给出中断，读出系统通过响应该中断来判断数据传输的结束。因此，响应 VME 总线中断的能力是数据获取读出系统的一个重要指标。

4.2.4.1 测试方法

在用于测试的 VME 机箱中，使用一块 MVME2431 做中断源产生中断，MVME5500 负责响应中断，中断处理完毕后即通知中断源产生下一个中断。循环多次，计算平均每次所需时间。

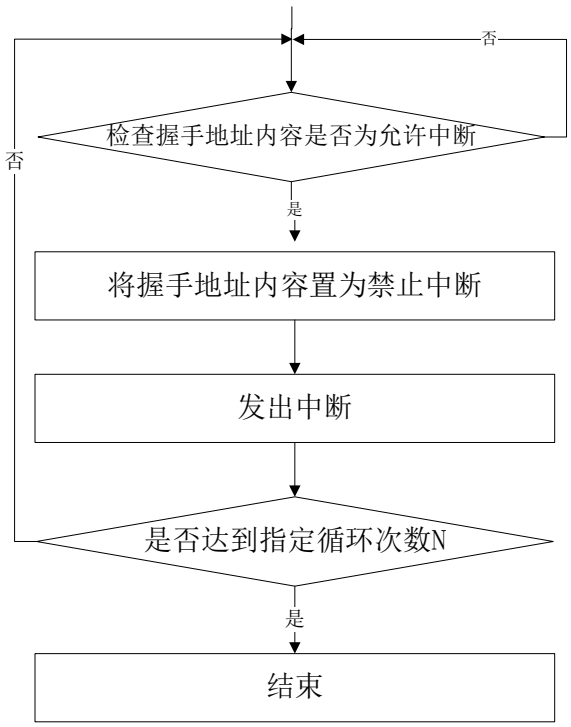


图 4-5 MVME2431 产生中断流程图

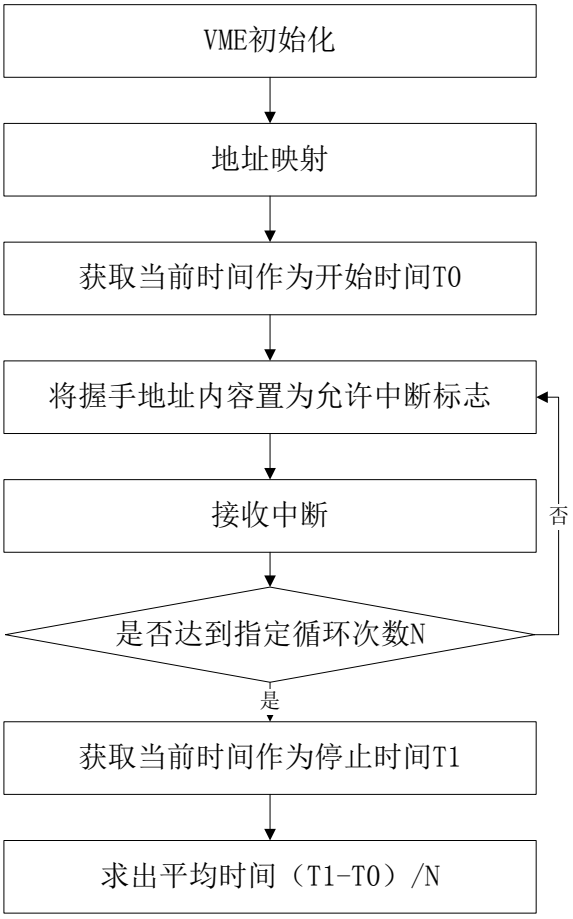


图 4-6 MVME5500 接收中断示意图

如图 4-5 和图 4-6 所示，为了保证每个中断都被 MVME2431 正确发出并且被 MVME5500 正确响应，MVME5500 和 2431 通过读写同一 VME 地址来进行握手通讯。

4.2.4.2 测试结果

使用如上的测试方法测到中断响应时间约为 17us，而上述的含有握手机制的测试方法决定了测到的一次中断响应的时间中至少包含了：

- 两次 32 位 VME 总线写操作时间
- 一次 MVME2431 产生中断的时间
- 一次 MVME5500 响应中断的时间

因此实际上响应一次中断的时间是要小于测得的这个实际值的。作为保守估计，可以认为 MVME5500 响应一次前端电子学中断的时间为 16us。

那么在事例率为 1000Hz 时，若采用单事例读出，即一个事例产生一次中断的前提下，CPU 时间每秒钟内有 16ms 用于响应中断，CPU 占用率约为 1.6%；如果采用多事例读出模式（N 个事例一次中断），相应的 CPU 占用率会更低（(1.6/N)%）。

4.2.5 上下文切换

数据获取读出系统是设计为多任务并行的，在基于嵌入式 Linux 的系统中，是通过多线程编程来实现任务的并行，因此线程间的切换时间也是决定读出系统工作效率的一个重要的指标。

4.2.5.1 测试方法

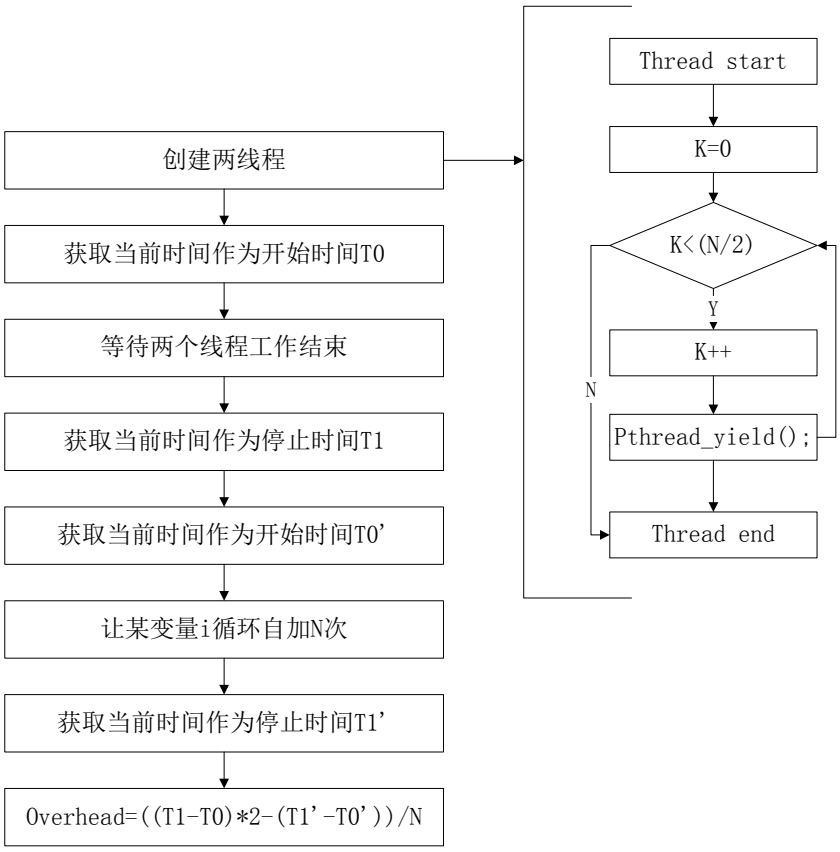


图 4-7 线程切换开销测试流程图

测试方法如图 4-7，创建两相同线程，每个线程都是一个循环自加的函数，从 0 到 N/2，每次加一，并且每次自加后调用 pthread_yield()函数让出 CPU 控制权，相应的另一个线程便会获得 CPU 开始执行。如此循环直

到变量达到 N/2，线程结束。获取线程开始前的时间与工作结束的时间，差值作为两个线程的工作时间和，并且获取某变量自加 N 次的时间，由此求得上下文切换的基本开销[36]。

4.2.5.2 测试结果

根据以上的测试方法，测到的在基于 MVME5500 的 Linux 操作系统中上下文切换的时间是 5.78us。那么当事例率为 1000Hz 时，若采用单事例读出，并且假设系统调度方式为最基础的基于时间片的轮转调度的前提下，每处理一个事例至少进行三次上下文切换（三个处理数据的任务），CPU 时间每秒钟内约有 18ms 用于上下文切换，则 CPU 占用率约为 1.8%；如果采用多事例读出模式，一次处理多个事例，相应的 CPU 占用率会更低。

4.2.6 网络性能测试

读出系统组装完成的事例数据，最终会通过网络发送给 DAQ 后端组件进行存盘以及其他处理，因此网络性能是保证数据流能够得到及时传输的重要参数。

4.2.6.1 测试方法

使用 TCP/IP 协议，Client/Server 模式进行网络传输。MVME5500 作为 client 端，持续向作为 server 端的 PC 机发送指定大小的数据段。Client 端的测试流程如图 4-8 所示，而 server 端的任务很简单，就是负责接受来自 Client 端的数据。

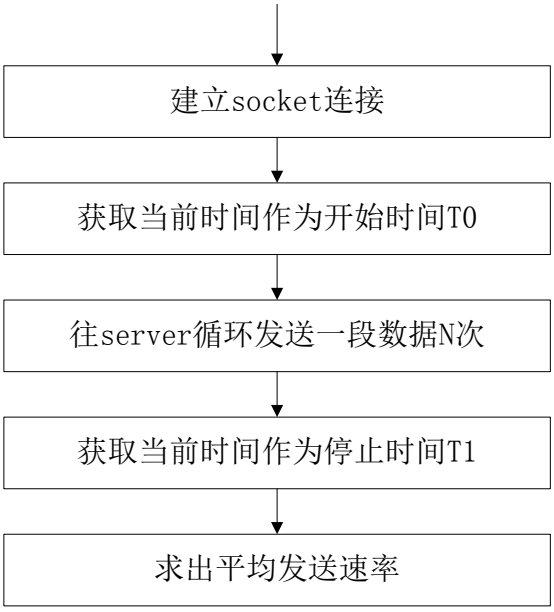


图 4-8 网络速率测试流程图（client 端）

4.2.6.2 测试结果

利用上述测试方法，使用百兆网口的情况下，改变每次传输数据块大小时，测到的通过网络发送的数据带宽如表 4-2 所示。

表 4-2 百兆网口传输性能测试结果

传输字节	128	256	1024	4096	8192	16384	32768
MByte/s	11.64	11.64	11.73	11.74	11.75	11.76	11.76

MVME5500 有百兆和千兆两个网口，同样依据以上的测试方法，当传输数据块大小为 1024Bytes 时，分别测得的传输性能如表 4-3 所示。

表 4-3 双网卡网络传输性能测试结果

client	server	thoughput	CPU occupancy
100M	100M	92Mbps	14%
1000M	1000M	550Mbps	67%

4.2.7 单项测试小结

综合以上几节的测试内容，可以得出结论：在基于 MVME5500 的嵌入

式 Linux 操作系统上所进行的单项测试中得到的各项性能指标良好，完全可以满足大亚湾实验数据获取系统的需求。各项性能指标测试结果如表 4-4 所示，表中最右边一列是使用基于 VxWorks 操作系统的 MVME5100[37] 为平台所测到的对应性能指标[28]，除了控制器以及控制器使用的操作系统不同，其他测试环境基本相同，列在这里可以有一个较为直观的比较。

表 4-4 基于 Linux 的 MVME5500 单项测试结果

	TimeSys Linux	VxWorks
	MVME5500 & 2431	MVME5100 & 2431
VME Read	1638 ns	1300 ns
VME Write	405 ns	500 ns
DMA(4096bytes)	18.42 MB/s	18.90 MB/s
DMA CPU Overhead	15 us	9 us
Interrupt CPU Overhead	16 us	15 us
100M Network Speed(1024bytes)	92 Mbps	86 Mbps[38]
Network CPU Usage	14%	57%
Context Switch Overhead	5.78 us	—

4.3 综合性能测试

4.3.1 测试平台介绍

上一节中对 Linux 操作系统中各项与读出相关的单项性能的测试，均给出了良好的性能指数，为在大亚湾数据获取读出系统中应用嵌入式 Linux 操作系统建立了初步的信心。为确保 Linux 操作系统在高能物理实验中的可行性，在此基础上，仍然需要进一步了解 Linux 操作系统下相关重要参数的综合性能表现。在大亚湾的读出电子学机箱还没有准备好时，为了全面了解 Linux 平台在读出系统中的综合表现，搭建了基于 BESIII 量能器电子学系统的综合性能测试平台。

4.3.1.1 硬件环境

测试中使用了 BESIII 的量能器（Electro-Magnetic Calorimeter, EMC）机箱作为电子学读出机箱。如

图 4-9 所示，在一个量能器的读出 VME 机箱中除了一块基于 Linux 操作系统的控制器 MVME5500，还有 15 块电子学读出插件以及一块读出控制插件 ROC（ReadOut Control）。控制器 MVME5500 的网口通过千兆交换机连接调试主机（Remote Debug PC），同时在线控制主机（Online Readout PC）也通过千兆交换机连接调试主机。

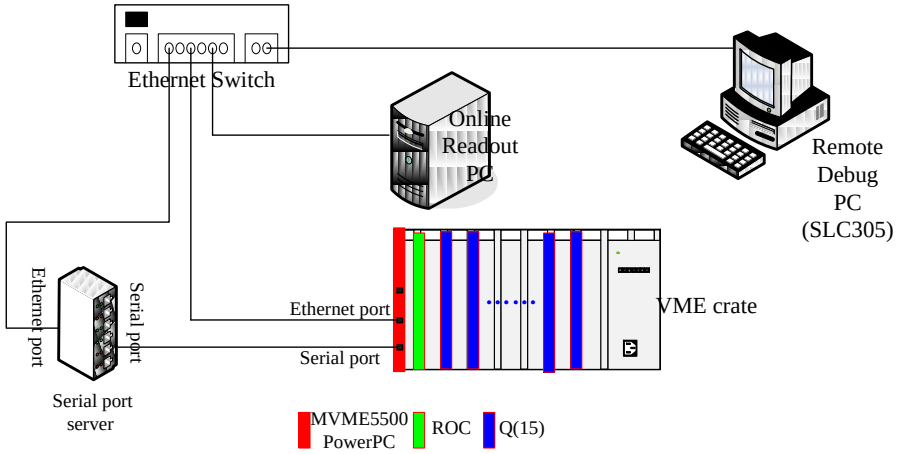


图 4-9 综合测试硬件环境

4.3.1.2 软件平台

调试主机安装 Linux 操作系统，内核版本 2.6.9，GCC 版本 3.4.6；在线控制主机安装 SLC（Scientific Linux CERN） release 4.6，内核版本 2.6.9，GCC 版本 3.4.6；MVME5500 安装 Timesys LinuxLink 产品，Linux 内核版本 2.6.9，GCC 版本 3.4.1。

整套测试的数据获取软件使用了 BESIII 的软件框架。而 BESIII DAQ 系统的读出部分是采用了 VxWorks 操作系统，因此需要将读出程序的代码从 VxWorks 平台移植到 Linux 平台。而 DAQ 的后端 Online 软件则完全使用 BESIII 的 DAQ 软件。

在对读出代码进行移植的过程中，不可避免的会遇到由于操作系统的差异性带来的一系列问题。例如：头文件名称或者路径的差异性、宏定义

的差别等，而最主要的差异是部分 VxWorks 的系统函数在 Linux 系统环境下没有同名的函数实现。对于这类问题，主要有以下两种情况[39]：有些 VxWorks 下的系统函数，在 Linux 系统中可以找到与之功能相似的函数，对于这类情况，只需要用这些功能相似的函数来等价的实现对应的 VxWorks 系统函数。例如，Linux 操作系统下的 usleep 函数对应实现 VxWorks 操作系统下的 taskDelay 函数的功能。另一种情况是在 Linux 系统中无法直接用功能相似的函数来等价实现，对于这种情况，在研究 VxWorks 程序源代码的基础上，用 Linux 系统函数改写这些函数段，使之完成与原程序段完全相同的功能，在移植过程中有大量的函数段需要用 Linux 的系统函数来改写。

在移植工作完成后，便可以开始利用该基于 BESIII DAQ 软件框架以及 BESIII EMC 硬件平台的数据获取系统进行基于嵌入式 Linux 的数据获取读出系统的综合性能表现的测试。

4.3.2 综合测试性能结果

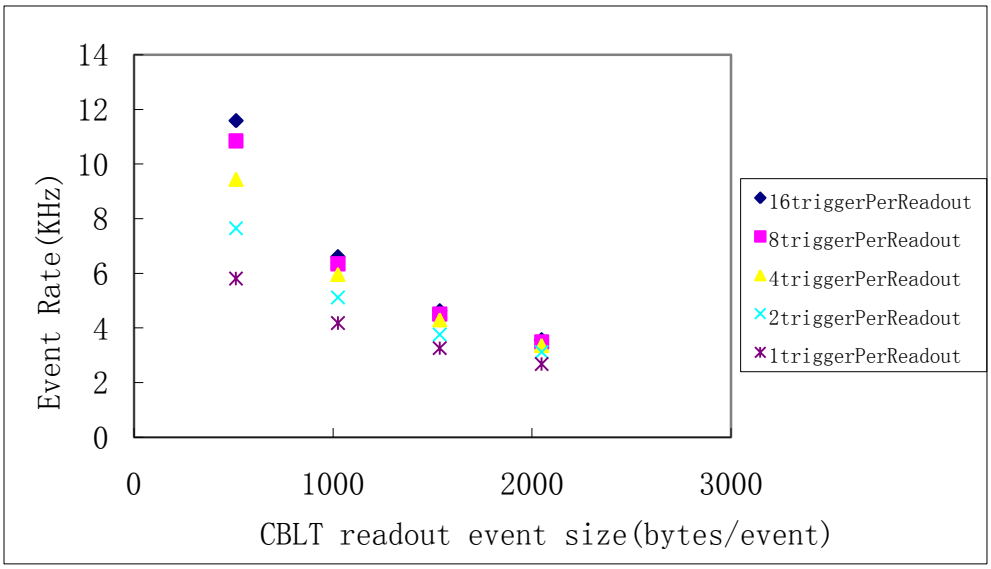


图 4-10 综合测试性能图

综合测试的性能结果如图 4-10 所示，基于嵌入式 Linux 的读出程序可以稳定地运行，配合后端在线软件，完成数据片断从电子学机箱的读出、数据的检查与第一级事例组装以及通过网络将事例片断发送给在线软件。

当系统采用单事例读出模式，每次读出数据长度为 2Kbytes（约为大亚湾实验 AD 机箱全击中情形下的单事例长度）时，可以达到约 2.7KHz 的事例率，即数据吞吐量为 5.4Mbytes/s，此时 MVME5500 的 CPU 有 50%的空闲。这样的性能指标完全满足大亚湾实验的事例率与数据带宽的需求。后面两节将详细地分析测试过程中遇到的问题以及解决的方法。

4.4 VME Universe 驱动的研究与优化

4.4.1 中断的处理

4.4.1.1 两种中断方式

产生中断的设备对于何时撤销中断通常有三种选择[40]：

(1). 当处理器接收到来自设备的中断请求，会读取设备的状态寄存器并转入对应的中断服务程序，设备响应到状态寄存器上的读操作，便释放中断。

(2). 当处理器接收到来自设备的中断请求，会转入对应的中断服务程序并往设备的控制寄存器写值，设备响应到控制寄存器上的写操作，便释放中断。

(3). 当处理器接收到来自设备的中断请求，会从设备读取 status/ID。设备响应到状态寄存器上的读操作，便释放中断。

使用 1、2 两种方式进行中断撤销的被称为 RORA(Release On Register Access)方式的中断，而使用第三种方法的则被称为 ROAK(Release On ACKnowledge)中断。图 4-11 示意了 ROAK 和 RORA 分别是在怎样的条件下释放中断请求的。

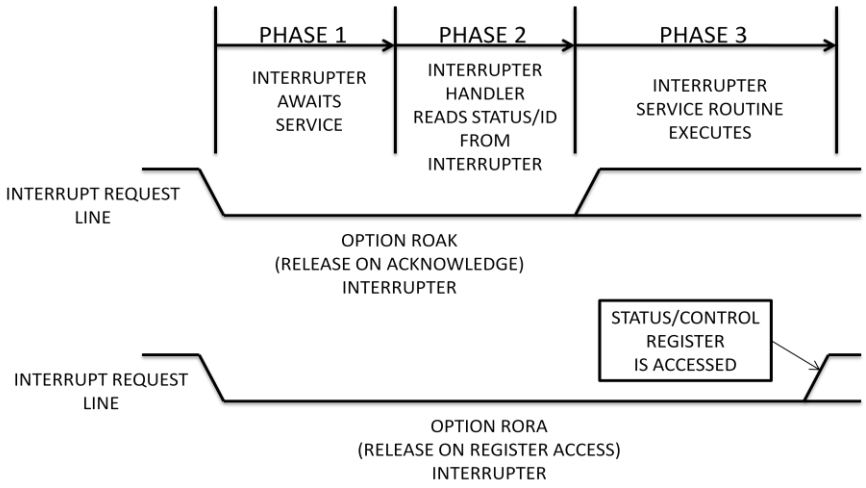


图 4-11 两种中断释放方式示意图

4.4.1.2 VME Universe 的中断处理

UniverseII 芯片驱动 VME Universe 对于中断的处理如图 4-12 所示，当内核态的中断处理例程 ISR(Interrupt Service Routine)接收到中断以后，会立即禁止中断。接下来的工作是由中断下半部(bottom half)来完成的。中断下半部会首先读取中断的 Status/ID，随后清空中断寄存器，然后重新将寄存器设置为允许响应中断，最后通知用户进程。

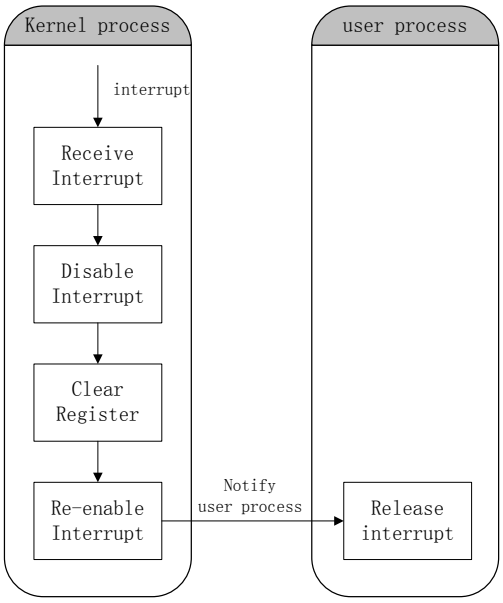


图 4-12 VME Universe 驱动的中断处理流程

4.4.1.3 存在问题

4.4.1.1 节中已经介绍了中断的释放方式有两种：RORA 和 ROAK。如果是 ROAK 的方式，那么在中断下半部执行的第一步，读取中断 Status/ID 时，中断就已经得到了释放。而如果是 RORA 的中断，从图 4-12 可以看出，在内核态通知用户进程之前，已经将寄存器设置为允许中断响应，因此，在用户态接收到来自内核态的通知，从而对设备的状态寄存器进行读写操作以释放中断之前，内核态有可能已经再次响应了同一次中断源。

读出系统对电子学数据的读取是由中断驱动的，电子学插件准备好了指定的事例便会送出中断请求。而如果发生了上述的情况，对于一次中断请求，由于没有及时的释放中断而造成多次的中断响应，这样多出的中断响应所驱动的数据传输所得到的数据必然是异常的。同时因为没有及时释放中断而导致内核态忙于响应中断，也会给 CPU 造成额外的负担。综上所述，对于 RORA 的中断模式，必须对驱动的中断处理流程进行相应的优化设计。

4.4.1.4 解决方法

为了避免上述问题的产生，提高系统效率，并且不影响 VME Universe 驱动的整体工作流程，对驱动做了以下针对性的修改，即在收到约定级别的中断时，一旦中断位清零后便直接通知用户态，而不打开中断服务程序，将再次打开中断功能的操作置于用户态，即在用户态释放中断、结束 CBLT 传输后，准备接收下一次的中断时，才将中断重新打开，如图 4-13 所示。

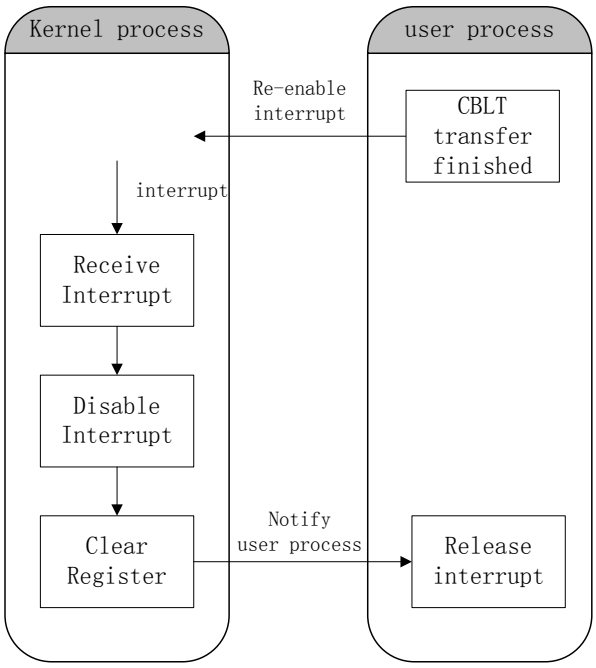
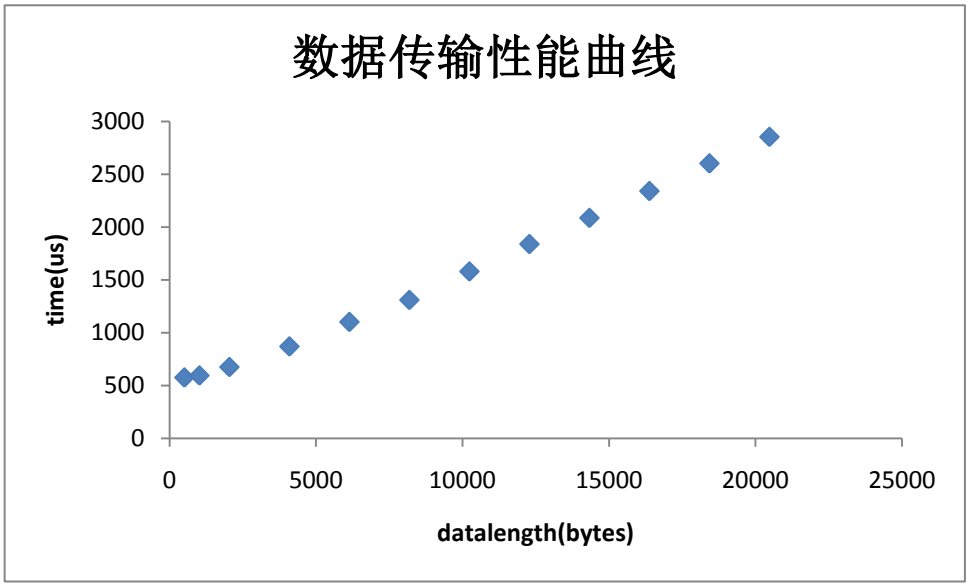


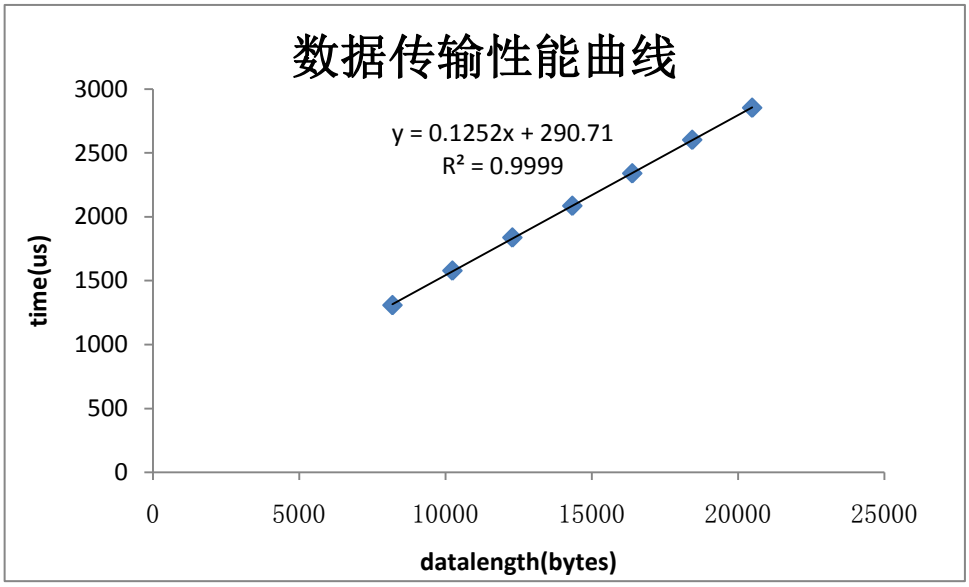
图 4-13 修改后的中断处理流程

4.4.1.5 优化前后对比

在 BESIII 量能器电子学插件的设计中，中断与数据是密不可分的，有数据才会引起真实的数据中断。因此为了测试中断的接收情况，必须借助于中断加上数据读出这样的工作模式，程序才能得到循环运行。通过示波器测量已知量能器读出机箱的传输速率可以达到 8Mbytes/s（即每微秒传输 8 个字节），与 4.2.3 节中 DMA 传输测试的结果对照能够看出，当每次传输数据块的大小超过 256 字节时，量能器机箱的数据读出速率便能够稳定在 8bytes/us（即每个字节的传输时间为 0.125us）。每次传输过程的完整 CPU 开销应当是数据传输的 CPU 开销加上接收中断的基本开销，这两个数据可以从表 4-4 中查到实测的结果，两者相加大约为 31us（15us+16us）。由此我们可以得出结论，如果对传输数据块大小与对应传输时间构成的数据对进行线性拟合，期望得到的数据传输性能曲线应当为 time(us) = 0.125*datalength(bytes) + 31us（其中最小的数据块应当大于 256 字节）。



（a）散点图



（b）直线拟合

图 4-14 修改驱动前数据传输性能情况

图 4-14 示意了在原先的中断处理模式下，数据传输的性能情况。图（a）所示是测试的传输数据块（均大于 256 字节）与对应传输时间所构成的散点图，可以清楚的看到传输性能在低端偏离直线较为明显，因此舍弃了低端的部分点对后，重新作出传输性能曲线并采用直线拟合，如图 4-14（b）所示，从图中看到拟合公式中斜率（即每字节的传输时间）为 0.1252us/byte，和预期值 0.125 us/byte 基本一致，但直线的截距（即传输一次的完整开销）

为 290.71us，远远超过预期值 31us。这正验证了之前提出的观点：在这样的中断处理模式下，因为没有及时释放中断而严重加剧了 CPU 的负担。

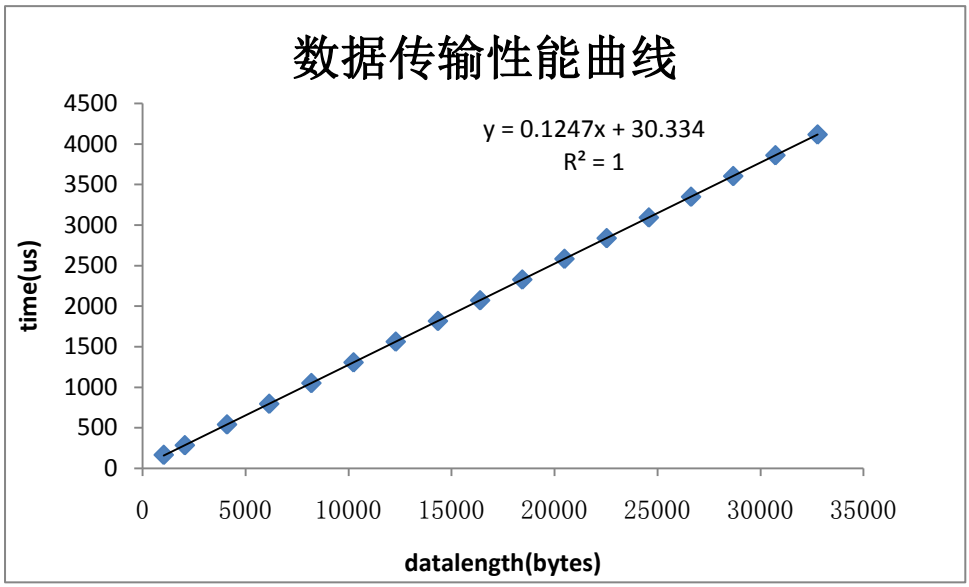


图 4-15 驱动优化后数据传输性能情况

图 4-15 是采用了之前介绍的优化方案以后数据传输的性能情况，可以看出优化之后数据传输的性能得到了明显改善，根据传输数据块大小与传输时间拟合得到的直线为 time(us) = 0.1247*datalength(bytes) + 30.334us，与预期的传输性能 time(us) = 0.125*datalength(bytes) + 31us 完全符合，并且拟合直线的 R^2 = 1，说明该拟合直线非常好的涵盖了实验数据，线性特征非常明显。并且值得一提的是，经过这样的优化处理，读出的正确性得到了有效的保障，不会出现本节前面提到的因为重复响应同一中断而造成的读到异常数据的情况。

最后，需要说明的是以上的优化是针对 RORA 的中断方式所进行的，如果是 ROAK 的中断方式，原先的驱动的设计是可行的，而不需要做这样的优化处理。

4.4.2 内存泄漏的解决办法

Linux 内核独立于普通应用程序，它一般处于系统态，拥有受保护的内存空间和访问硬件设备的所有权限。而应用程序在用户空间执行，不能直接访问硬件。如图 4-16 所示[43]，应用程序通过系统调用和内核通信来运行。

应用程序通常调用库函数—再由库函数通过系统调用界面让内核代其完成各种不同任务。

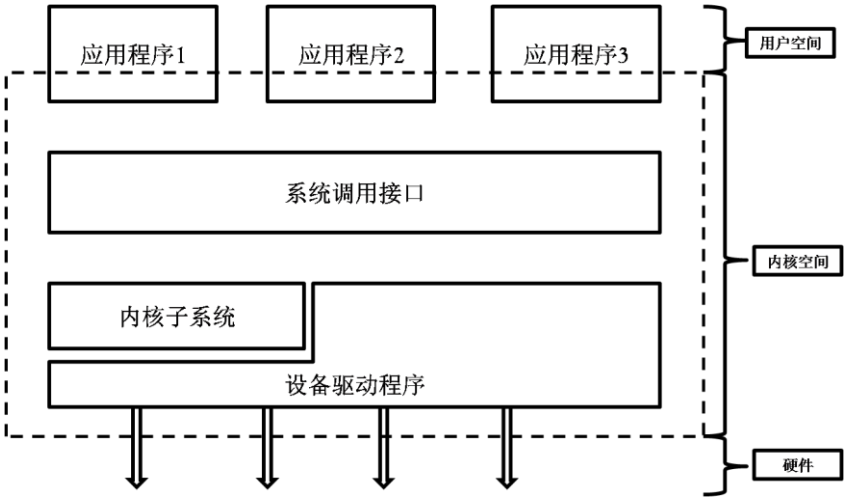


图 4-16 应用程序、内核与硬件的关系

对应于 VME Universe 的驱动，上述关系可以这样理解：用户态的应用程序直接调用驱动提供的 API，而 API 通过系统调用接口—即 VME 库函数—再由库函数调用设备驱动程序，由内核进行对硬件的访问。

4.4.2.1 存在问题

在系统长时间运行的过程中发现有内存泄漏导致程序崩溃的现象，经过分析排查发现是由中断响应函数导致的。首先检查了用户态的中断处理程序，确认了没有可能造成内存泄漏的情况，而设备驱动程序的中断处理部分经过确认也没有内存泄漏的情况。那么焦点就集中在了系统调用接口即 VME 库函数的部分。

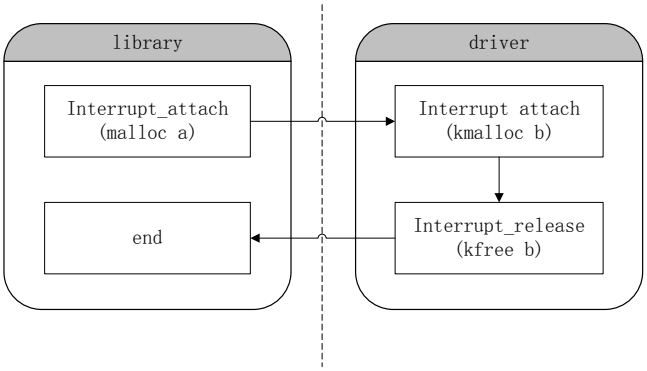


图 4-17 中断函数调用示意图

从图 4-17 中可以看出，库函数的中断响应函数 interrupt_attach 中为用户态的中断 handle 开辟了一段内存 a，而库函数所调用的驱动程序中也为内核态的中断 handle 开辟了内存 b，通过图中所示的调用关系可以看到，内存 b 在内核态中得到了及时释放，而内存 a 在库函数执行返回时并没有得到释放。因此在读出程序中，当反复接收中断，即循环执行 interrupt_attach 函数到达一定次数后，便会因为内存泄漏而导致程序崩溃。

4.4.2.2 解决方案

对于上述的问题，解决方案当然是有多种的，这里只选用了一种最简便易行的方法。如图 4-18 所示，只需在驱动程序执行完毕返回库函数中后，在库函数执行完毕退出之前加上释放内存 a 的语句即可。按照这样的逻辑修改库函数之后，重新运行读出测试程序，便不再出现原先的内存泄漏的现象，从而验证了上述解决方案是有效的。

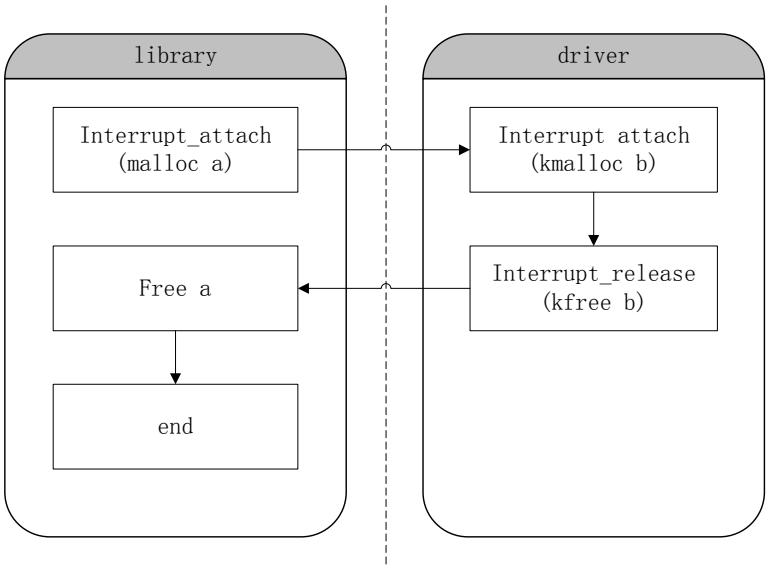


图 4-18 修改后的中断函数调用示意图

4.4.3 cache 相关的处理

4.4.3.1 存在问题

在测试过程中发现实际测到的读出效率和预期值（这里的预期值来源于基于 VxWorks 的 DAQ 读出系统）存在一定的差距，通过仔细的分析查

找到原因是因为 CBLT 读出的数据需要从系统默认的内存拷贝到程序中的环形缓冲区，在传输数据量增大时，这一内存拷贝的过程耗费了非常多的时间，导致了读出效率的明显下降。通过对 VME Universe 驱动的研究发现，为了避免 cache 不一致性的问题，底层函数为 CBLT 分配的内存缓冲区是一段禁止 cache 的区域。因此在读出系统中，每当 CBLT 传输数据完成后，在进行内存拷贝这一过程中，这段分配给 CBLT 的内存缓冲区由于 cache 的被禁，导致拷贝过程变得非常耗时（尤其是大数据量时），对读出系统的效率产生了很大影响。

4.4.3.2 优化方案

为了解决这个问题，很容易想到这样的优化方案：即分配一段允许 cache 的内存缓冲区替代 VME Universe 驱动为 CBLT 分配的禁止 cache 的缓冲区，便可解决内存拷贝耗时过多的问题。但是 CBLT 传输所需要的内存空间必须是物理地址连续的，因此不能通过常用的内存空间的分配函数（如 malloc 等）来分配这段内存地址。于是这里采用了 ATLAS 提供的 CMEM 驱动[41]，该驱动的主要目的就是为 Linux 用户分配一段物理地址连续的内存，这就满足了实验的需求：首先这是一段物理上连续的内存，同时这是允许 cache 的内存。在使用 CMEM 驱动分配的内存来作为 CBLT 传输地址后，通过实验检验发现读出系统的效率立刻得到明显提高。然而，这样的处理方案也带来了新的问题，即 cache 不一致性的问题。

由于 DMA 传输过程中不受 CPU 的控制，因此传输的数据只存放在内存缓冲区里而不是 cache。当数据拷贝到环形缓冲内时，由 CMEM 开辟的这段允许 cache 的区域会将数据同时复制到 cache 中。当下一次 CBLT 传输完成后，内存缓冲区内的数据得到了实时的更新，而 cache 里存有的过期数据便导致了 cache 不一致性的问题，也就是说拷贝到环形缓冲区内的数据可能有部分过期数据。为解决这个问题，需要在每次内存拷贝以后，进行下次 CBLT 传输之前，令 cache 失效，这样便可保证每次 CBLT 传输结束后，cache 中不会存在过期数据。在 Linux 内核中有实现这样功能的函数（invalidate_dcache_range），但是需要修改内核源代码来导出该函数，以便

实现内核模块的调用。根据以上方案，按照系统要求对 VME Universe 驱动以及 Linux 内核加以适当的修改后，便彻底解决了上述 cache 不一致的问题。

4.4.3.3 优化前后性能对比

根据上一节的内容完成这一解决方案后，随即用实验检验了该优化方案的可行性，在其他实验条件均同等的情况下，实验测到优化前后两种不同方案下，读出事例率随传输数据包大小变化的情况，具体测量结果如表 4-5 所示。

表 4-5 优化前后事例率对比

Data Size(byte)	Event Rate(kHz) (original method)	Event Rate(kHz) (CMEM method)
221	10.39	7.98
343	8.86	7.32
465	7.68	6.9
587	6.81	6.41
710	6.12	5.98
832	**5.69**	**5.69**
954	5.16	5.36
1076	4.78	5
1198	4.48	4.87
1320	4.17	4.62
1686	3.49	4
3151	2.07	2.65
4616	1.47	1.94
6080	1.14	1.55
7498	0.94	1.29

为了更直观地比较优化方案与系统原有工作方式的优劣，将表 4-5 中的数据绘图，如图 4-19 所示，可以看到两条曲线有一个交点。在传输数据

量约为 800 字节时，两曲线相交，当传输数据量小于这个值时，系统默认的工作方式性能更优；反之，在传输数据量大于该值时，优化方案的采用对于系统性能有明显提升。造成这个情况的主要原因是当传输数据量变小时，内存拷贝本身占用的时间变少，而为了防止 cache 不一致的令 cache 失效的操作在每次取数中占用的时间比重相对加大，所以这一优化方案在传输数据量小于一定量（约为 800bytes）时，便不能起到提升性能的作用，反而降低了原本的系统性能。

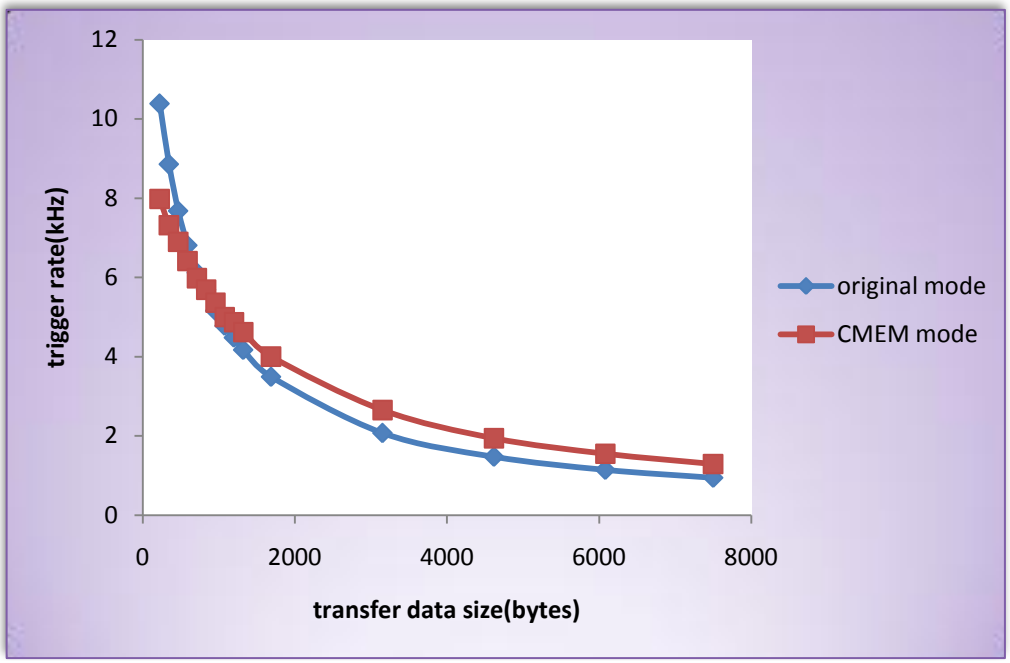


图 4-19 优化前后事例率对比

4.4.3.4 结论

根据前面几节的分析，可以得出这样的结论：在一次传输数据量小于 800 字节时，采用系统默认的内存分配方案性能更好；反之，采用优化后的方案管理内存会使整体性能有明显的改善。因此在实验的应用中，用户可以根据不同的实验以及探测器的具体情况灵活采取较为适宜的方案，以最大程度上保证性能的最优化。

4.5 其他提高系统性能的措施

4.5.1 多事例读出

多事例读出是指，当电子学插件准备好指定的 N（>=2）个事例后，发出中断通知 DAQ 系统一次读走这 N 个事例。当 N 为 1 时，也就是最简单的情形，即单事例读出方式。通常，电子学允许的 N 值最大为 16。采用多事例读出的优势主要有以下两点：首先，通过 4.2 节的介绍已经知道读出系统每次接收中断都有大约 16us 的系统开销，如果每 16 个事例才发出一次中断，就可以节省了 15 次接收中断的基本开销时间；同时，通过 4.2.3 节中 DMA 传输的性能曲线（图 4-4）也可以看出，传输的数据量越大，DMA 的传输速率也越高，当一次传输 16 个事例时，数据量变成原先的 16 倍，传输的速率也会得到明显提高。因此，根据实验的具体情况，如果单事例读出模式不能够满足实验的需求，需要进一步提高系统的性能时，一个通常采取的重要的方法就是采用多事例读出。从图 4-10（4.3.2 节）中可以看到采用多事例读出模式可以有效提高系统的读出性能。图 4-20 所示是采用多事例读出模式对 CPU 占用率的影响。综上所述，采用多事例读出模式，可以节省系统开销，降低 CPU 的占用率，有效提高 DAQ 系统的综合读出性能。

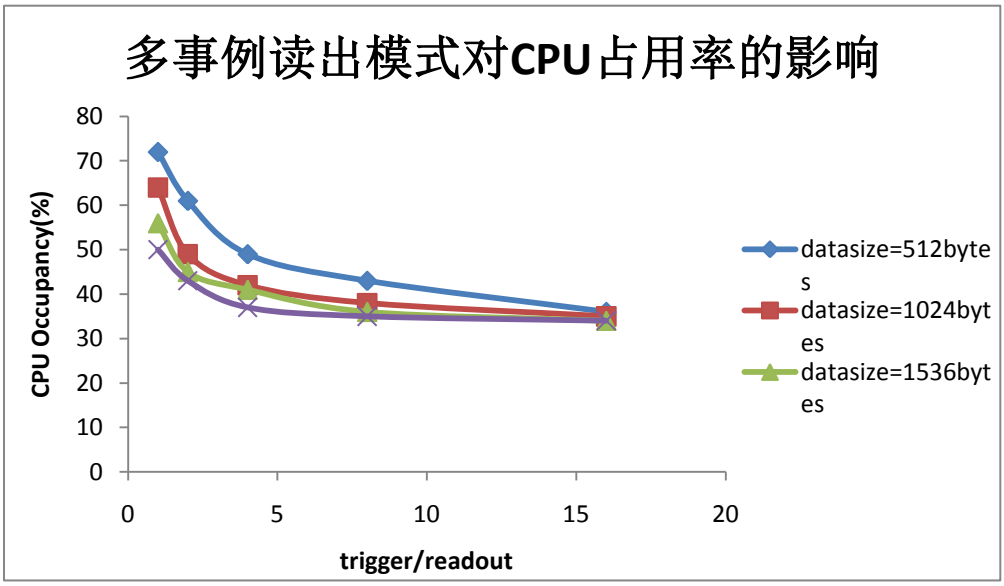


图 4-20 采用多事例读出模式对 CPU 占用率的影响

要在数据获取系统中实现多事例读出的模式，首先前端电子学插件的设计必须满足这一要求，即电子学插件能够缓存指定数目的事例后才向 DAQ 系统发出取数中断申请；其次在读出系统进行事例的拆解时，必须能够区分来自同一次触发的不同事例，并且按照事例的顺序将所有插件的数据重新进行组装，这部分的详细内容参见 5.4 节。

4.5.2 并行的多任务系统与任务优先级的设置

3.4.3 节中已经介绍过，读出系统由四个主要任务组成，其中最核心的负责数据流传递的三个任务是：CBLT 读出任务、事例组装任务与网络传输任务。首先考虑最简单的情形，即三个任务以串行方式实现，具体说就是一次 CBLT 读出任务完成后进行一次事例组装，组装好的事例立刻交由网络传输任务进行发送，网络发送完成后才允许开始下一次的 CBLT 读出任务。在本章的实验条件下，串行工作的读出性能如表 4-6 所示。

表 4-6 串行读出性能

Data Size(bytes)	Event Rate(kHz)	Bandwidth(MB/s)
2048	2.16	4.42
1024	3.6	3.69
512	5.6	2.87

任务间串行工作的方法实现起来较为简单，但是效率较低。在以上三个任务中，CBLT 传输任务是最耗时的，但是执行 CBLT 传输的过程中是不占用 CPU 的，因此希望在 CBLT 传输的过程中，CPU 可以执行数据组装与网络传输任务，这样便可真正意义上实现三个任务的并行，从而有效利用 CPU 资源，最大限度地提高系统的效率。

Linux 2.6 提供了抢占式的多任务模式，由调度程序来决定多进程运行的顺序。调度算法中最基本的一类就是基于优先级的调度。相同优先级的进程按轮转的方式进行调度，而优先级高的进程会被优先选择。在数据处理的三个任务中，CBLT 读出任务是整个系统数据获取的源头，并且传输过程中不占用 CPU，其他任务一样还是可以得到 CPU 运行，因此在三个任务

中 CBLT 传输任务应该拥有最高的优先级。

为了验证这一想法，将读出程序由单线程的串行工作改为多线程并行工作，设置三个任务为同样的优先级，在这样的设置条件下得到的系统读出性能和表 4-6 中的串行方式下的结果相差无几，也就是说系统并未得到真正意义上的并行工作，这正是因为前面提到的系统的调度策略决定的，相同优先级的进程是按照轮转方式进行调度的，同样优先级的任务之间不会发生抢占的情形。因此为了实现任务间的并行工作，需要为任务设置不同的优先级。于是按照上面的分析提高 CBLT 读出任务的优先级（在测试中，令 CBLT 任务优先级（nice 值）为-18，事例组装与网络发送任务的优先级相同，均为默认的 0）再次进行系统测试，结果如表 4-7 所示，与表 4-6 对比可以看出将 CBLT 任务设置为三个任务中最高的优先级后，读出系统的事例率有了明显提高。并且通过相应的一系列测试与计算可以证明，数据组装和网络传输任务基本上是在 CBLT 传输过程中同时进行的，没有对系统读出性能造成明显下降的影响，换句话说即是实现了预期中的并行工作的情形。

表 4-7 线程优先级对读出性能的影响

Data Size(bytes)	Event Rate(kHz)	Bandwidth(MB/s)
2048	2.39	4.89
1024	4.1	4.20
512	6.3	3.23

从上面的讨论可以看出，任务优先级的设置对于实现系统的并行工作，提高系统的工作性能有着重要的意义，因此如何为每个任务选取合适的优先级，从而达到最优的配置，需要在今后的实验过程中进行仔细的分析检验。

4.6 本章小结

为了实践检验高能物理实验的数据获取读出系统中应用嵌入式 Linux

的可行性,首先搭建了一套测试平台,测试了基于 MVME5500 的 Linux 操作系统的单项性能情况。随后在 BESIII 量能器机箱的基础上搭建了完整的读出系统的测试平台，进行了整体上的功能与性能的测试。通过对 LinuxLink 提供的 Linux 内核以及 UniverseII 芯片的驱动 VME Universe 的修改与完善，使得嵌入式 Linux 可以较好地应用于读出系统，并且提出了提高系统性能的可行措施,最终在系统的综合测试中给出了理想的运行结果。利用 BESIII 的软件框架与硬件平台，基于 Linux 操作系统开发的读出系统可以配合 DAQ 后端的在线软件稳定运行，系统的整体性能完全符合预期的效果，满足大亚湾实验的数据获取需求，从而充分验证了在大亚湾实验的数据获取系统中采用基于嵌入式 Linux 的读出系统的可行性。

5 大亚湾实验数据获取读出系统的实现

第 3 章中介绍了大亚湾实验的数据获取读出系统的详细设计方案，第 4 章中通过搭建测试平台验证了基于嵌入式 Linux 的读出系统的性能指标可以满足大亚湾实验的需求，本章将详细讨论读出系统的具体实现过程以及相关技术细节的处理。

5.1 读出系统的状态转换与实现

在 2.6.1 节中详细介绍了大亚湾数据获取系统整体框架软件中关于状态转换的设计与实现，本节中将会针对读出系统进行具体的分析。在读出系统，有一类命令是不做响应的，如 BOOT 命令，此时读出系统还没有被启动，因此也不存在相对应的 BOOTED 状态。还有一类命令虽然读出系统需要响应，以便配合其他 DAQ 组件进行同步的状态跳转，但是在状态跳转的过程中并没有实现具体操作，完全是状态的空转，例如 StopEF、StopSFO 等。第三类命令是和读出系统密切相关的，读出系统需要响应命令，并且根据所处的状态和接收到的命令，进行相关的处理工作，本节中将重点讨论这第三种情况的具体实现。

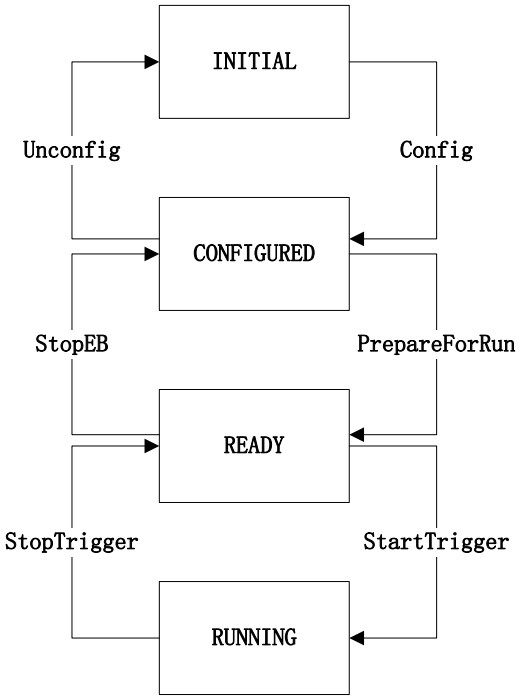


图 5-1 简化后的读出系统状态转换图

如图 5-1 所示，这是一张简化的读出系统 ROS 状态转换图，这张图中仅仅示意了跟读出系统密切相关的部分状态和命令，即上面提到的第三类情况。当 PMG 在接收到 Initial 命令将 ROS 加载到内存中去以后，此时的 ROS 处于 INITIAL 状态。接收到 Config 命令后，ROS 会开始配置过程，包括配置自身相关的参数，配置电子学插件，配置网络接口等等，当配置过程完成后，ROS 进入 CONFIGURED 状态。在 PrepareForRun 命令到后来，ROS 会启动相关任务（线程），包括 CBLT 读出任务、事例组装任务、网络传输任务、监测任务等，此时触发源还没有打开因此没有数据产生，但是相关数据处理的任务已完全就绪，ROS 进入 READY 状态。此时 ROS 一旦接收到 StartTrigger 命令，便立即通过写硬件寄存器打开触发源，整个系统进入取数流程。

停止取数的流程则相反，在 RUNNING 状态时，ROS 收到 StopTrigger 命令就关闭触发源，进入 READY 状态；在 READY 状态收到 StopEB 命令后，会停止各工作任务，进入 CONFIGURED 状态；此时收到 Unconfig 命令后，ROS 会进行一些清理工作如关闭网络接口、释放缓冲区内存、释放信号量等等，做完这些工作以后，ROS 又恢复到最初的 INITIAL 状态。

5.2 电子学配置数据库

5.2.1 DAQ 配置数据库

在数据获取系统中，配置数据库（Configuration Database，ConfDB）被用来存储系统中种类繁多数量丰富的各种参数，包括硬软件结构参数，系统运行状态参数和探测器相关参数等等。DAQ 的配置数据库提供了存储、读出、修改系统参数的应用程序接口和图形化用户界面。底层数据以可扩展标记语言(Extensible Markup Language, XML)格式组织。这保证了它能够在多种环境中使用，并有丰富的工具可供灵活选用。

整个配置的实现基于 OKS 组件。OKS 的主要静态存储方式是 XML 文件，XML 文件分为两类：用于定义类的 schema 文件与用于存储对象的 data 文件。OKS 为浏览或修改 schema 和 data 分别提供了图形化的应用程序接口 OKS schema editor 和 OKS data editor。OKS 数据库的另一种静态存储方式是以存档为目的的 RDBMS（relational database）。OKS 还提供了对远程数据库 RDB（remote database）的访问。

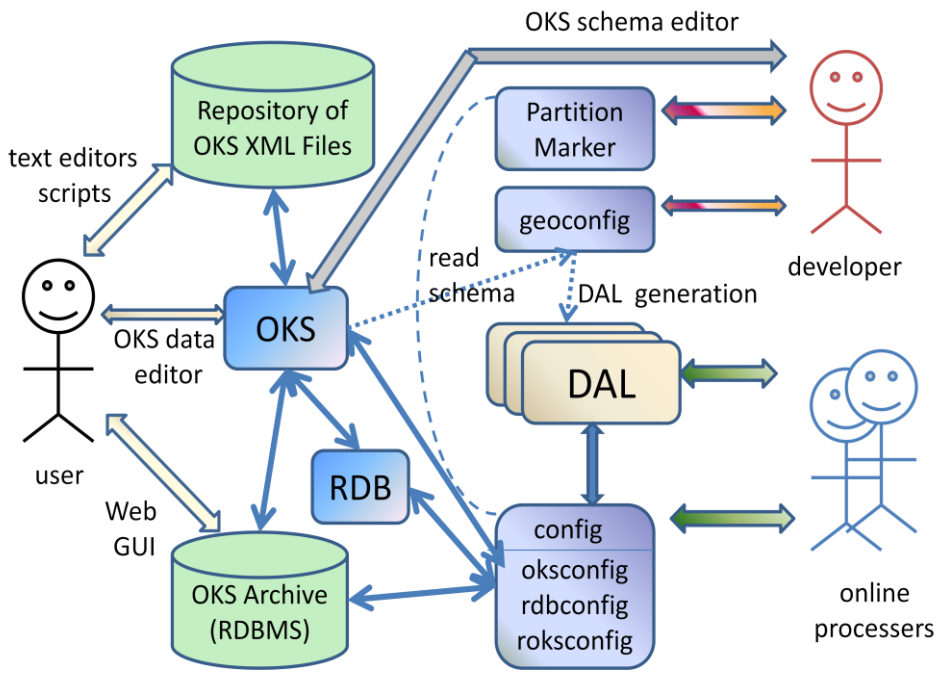


图 5-2 配置服务的用户接口

图 5-2 描述了数据库的接口与配置服务的用户之间的联系[42]，从图中可以看到用户（user）、软件开发人员（developer）与在线进程（online process）都具备多种访问数据库的接口，手段灵活，使用方便。

5.2.2 电子学配置数据库的设计

大亚湾实验中电子学插件的在线配置由数据获取读出系统实现，因此在读出系统中，除了自身的配置参数以外，针对探测器电子学相关的配置参数种类多，随着实验的需求经常需要修改，每次运行参数都需要存档备份等特点，需要特别设计电子学配置数据库（Electronics ConfDB）。

电子学配置数据库被用来存储探测器的信息与电子学插件的配置参数，如机箱编号、电子学插件种类、数目、位置、阈值信息等。电子学配置数据库同样采用如图 5-2 所示的框架结构，使用 XML 语言组织的电子学配置数据库使得对参数的改动灵活方便，能够适应系统的要求。在每次运行开始前，数据获取读出系统负责从电子学配置数据库里读到相应的配置信息来进行电子学插件的初始化与相关工作状态的设置。运行开始后，配置数据库里的内容都会被存储进在线数据库里。如图 5-3 所示便是当前的电子学配置数据库的结构方案（详细配置文件可参加附录 D）。

在 2.2.1.3 节中介绍了大亚湾实验的不同运行模式的需求，而不同运行模式下电子学插件的部分参数如阈值、触发模式等通常都是不相同的，因此电子学配置数据库的设计必须考虑到这一需求。如果在每次更改运行模式时都要修改配置数据库的参数，会给实验带来极大的不便，而且频繁改动参数也会增加因人为的不小心而出错的风险。因此，在电子学配置数据库里，针对那些与运行模式密切相关的参数，如阈值信息、触发模式等，应当为每种运行模式保留一份单独的配置参数。每次运行时，只需要根据当前的运行模式选取相应的参数进行配置，而省去了修改参数的不便与风险。

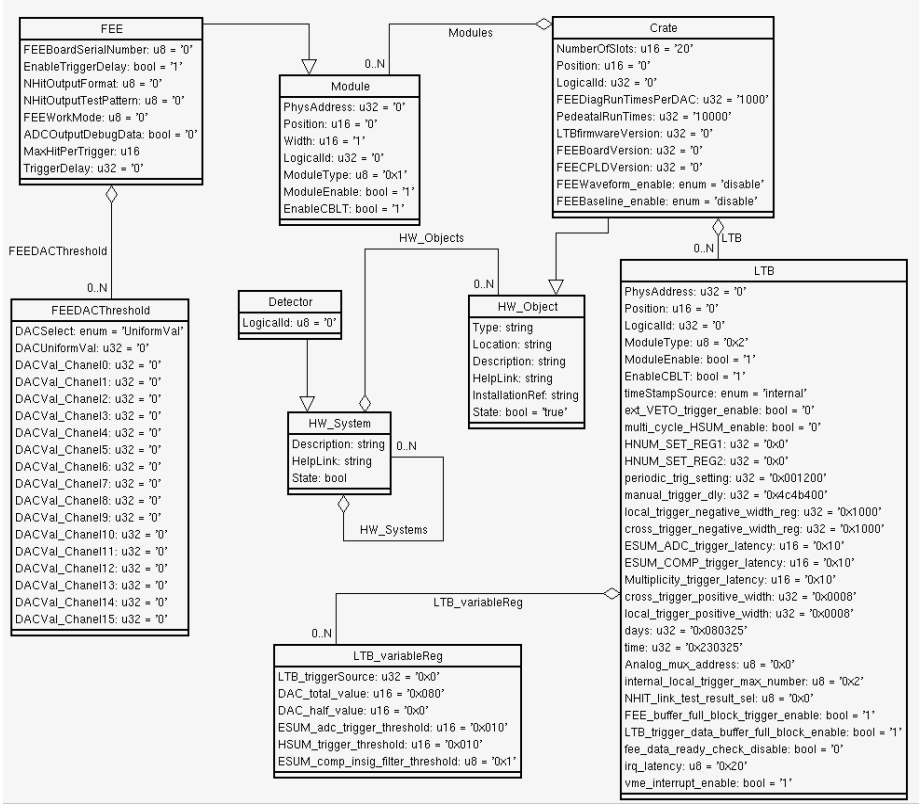
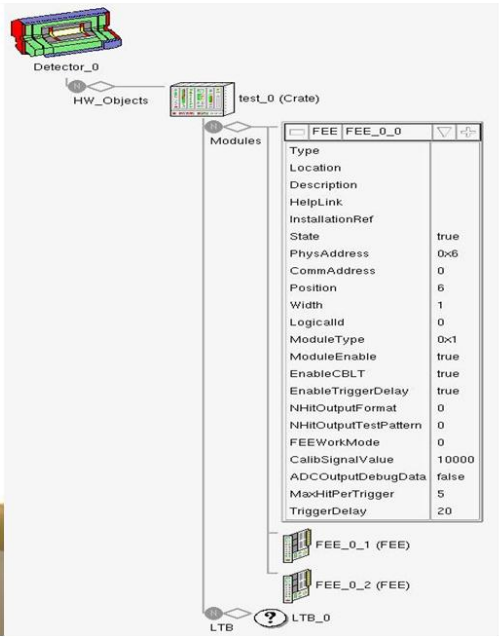


图 5-3 电子学配置数据库结构

5.2.3 电子学配置数据库的实现

TEST SYSTEM ENVIRONMENT

HARDWARE OBJECTS IN XML



XML DATA FILE

```
<obj class="Crate" id="test_0">
 <attr name="Type" type="string">""</attr>
 <attr name="Location" type="string">""</attr>
 <attr name="Description" type="string">""</attr>
 <attr name="HelpLink" type="string">""</attr>
 <attr name="InstallationRef" type="string">""</attr>
 <attr name="State" type="bool">1</attr>
 <attr name="NumberOfSlots" type="u16">0</attr>
 <attr name="Position" type="u16">0x1</attr>
 <attr name="LogicalId" type="u32">0x1</attr>
 <attr name="FEEDiagRunTimesPerDAC" type="u32">1000</attr>
 <attr name="PedeatalRunTimes" type="u32">1000000</attr>
 <rel name="Modules" num="3">
  "FEE" "FEE_0_0"
  "FEE" "FEE_0_1"
  "FEE" "FEE_0_2"
 </rel>
 <rel name="LTB" num="1">
  "LTB" "LTB_0"
 </rel>
 <rel name="ControlHost">"" ""</rel>
</obj>
```

图 5-4 电子学配置数据库示例

如图 5-4 所示，这是一个电子学配置数据库的具体实例。左上方是用于实验的电子学机箱照片，机箱中需要配置的电子学插件包括一块 LTB (Local Trigger Board)板和三块 FEE(FrontEnd Electronics)板。下方的图片是截取了电子学配置数据库 XML 文件的一段内容，这是一个描述机箱 test_0

的对象，这个 test_0 就对应于左边照片上的机箱，从文件内容中可以看到这个机箱中有三块 FEE 板，分别编号为 FEE_0_0，FEE_0_1 以及 FEE_0_2，还有一块编号为 LTB_0 的 LTB 板。右上方的图片是 DAQ 软件框架中的 OKS 组件通过读取配置文件生成的硬件对象图，从这张图片可以很直观的看到硬件的结构情况，并且如果展开相关的对象便可以看到隶属于这个对象的所有参数值的设置情况。图 5-5 所示，是运行结束后，通过访问在线数据库得到的每次运行保存的电子学配置信息。

RunNumber: 2344 ClassName: Crate

Row_No	OBJECTID	ValueVersion
1	test_0	1137
2	LTB_0	1154
3	PhysicsMode	1154
4	PedestalMode	1137
5	FEEDiagMode	1137
6	ADCalibMode	1137
7	WPCalibMode	1137
8	FEE_0_0	1154
9	PhysicsThreshold_0_0	1137
10	PedstalThreshold_0_0	1137
11	FEEDiagThreshold_0_0	1137
12	ADCalibThreshold_0_0	1137
13	WPCalibThreshold_0_0	1137
14	FEE_0_1	1154
15	PhysicsThreshold_0_1	1137
16	PedstalThreshold_0_1	1137
17	FEEDiagThreshold_0_1	1137
18	ADCalibThreshold_0_1	1137
19	WPCalibThreshold_0_1	1137
20	FEE_0_2	1154
21	PhysicsThreshold_0_2	1137
22	PedstalThreshold_0_2	1137
23	FEEDiagThreshold_0_2	1137
24	ADCalibThreshold_0_2	1137
25	WPCalibThreshold_0_2	1137

ObjectId: FEE_0_1

SCHEMAVERSION	DATAVERSION	CREATIONVERSION	CLASSNAME	OBJECTID	NAME	PARENTPOSITION	VALUE
11	1137	1137	FEE	FEE_0_1	ADCOutputDebugData	0	0
11	1137	1137	FEE	FEE_0_1	Description	0	
11	1137	1137	FEE	FEE_0_1	EnableCBLT	0	1
11	1137	1137	FEE	FEE_0_1	EnableTriggerDelay	0	1
11	1137	1137	FEE	FEE_0_1	FEEBoardSerialNumber	0	0
11	1137	1137	FEE	FEE_0_1	FEEWorkMode	0	0
11	1137	1137	FEE	FEE_0_1	HelpLink	0	
11	1137	1137	FEE	FEE_0_1	InstallationRef	0	
11	1137	1137	FEE	FEE_0_1	Location	0	
11	1137	1137	FEE	FEE_0_1	LogicalId	0	0
11	1137	1137	FEE	FEE_0_1	MaxHitPerTrigger	0	1
11	1137	1137	FEE	FEE_0_1	ModuleEnable	0	1
11	1137	1137	FEE	FEE_0_1	ModuleType	0	1
11	1137	1137	FEE	FEE_0_1	NHitOutputFormat	0	0
11	1137	1137	FEE	FEE_0_1	NHitOutputTestPattern	0	0
11	1137	1137	FEE	FEE_0_1	PhysAddress	0	0
11	1154(1137)	1154(1137)	FEE	FEE_0_1	Position	0	9(6)
11	1137	1137	FEE	FEE_0_1	State	0	1
11	1137	1137	FEE	FEE_0_1	TriggerDelay	0	0
11	1137	1137	FEE	FEE_0_1	Type	0	
11	1137	1137	FEE	FEE_0_1	Width	0	1

图 5-5 在线数据库中的电子学配置信息

5.2.4 电子学配置的实现

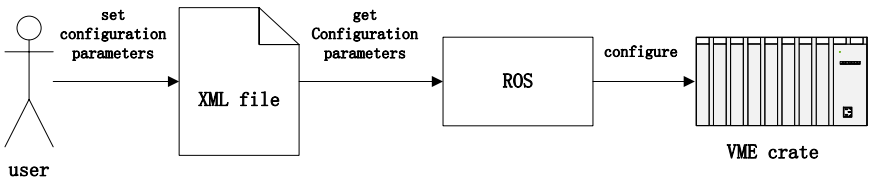


图 5-6 电子学配置的实现流程

读出系统中电子学配置的实现流程如图 5-6 所示。运行取数前，用户根据实验的需求配置好 XML 文件中的相关参数，例如阈值信息、触发类型等。运行时，读出系统从 XML 文件中提取配置参数信息，并按照严格的时序要求来配置各电子学插件的工作状态，具体的配置流程参见 6.3 节。正确的配置所有电子学插件的工作状态与相关参数，是整个系统能顺利运行，数据分析能够正常进行的必要前提条件。

5.3 多任务并行系统的实现机制

5.3.1 缓冲区保护

第 3 章已经介绍过，数据获取的读出系统设计为一个多任务并行的系统。为了实现这一设计，读出系统中的数据传递采用了生产者-消费者模式，如图 5-7 所示。缓冲区是生产者-消费者模型中的必要因素，对应于读出系统，在数据读出任务和事例组装任务之间，事例组装任务和网络发送任务之间，都需要设置数据缓冲区。增加了缓冲机制后，不仅可以支持任务的并行，同时使得系统稳定性增强，可以容忍一定程度的事例率抖动，而不至于由于一时抖动引起读数的死时间。

图 5-7 生产者-消费者模型

读出系统的两个数据缓存是通过构建两个环形缓冲区实现的。在通信程序中，经常使用环形缓冲区作为数据结构来存放通信中发送和接收的数据。环形缓冲区是一个先进先出的循环缓冲区，可以向通信程序提供对缓冲区的互斥访问。环形缓冲区通常有一个读指针和一个写指针，如图 5-8 所示。读指针指向环形缓冲区中可读的数据，写指针指向环形缓冲区中可写的数据。通过移动读指针和写指针就可以实现缓冲区的数据读取和写人。同时还必须添加互斥保护机制来确保数据的正确性。

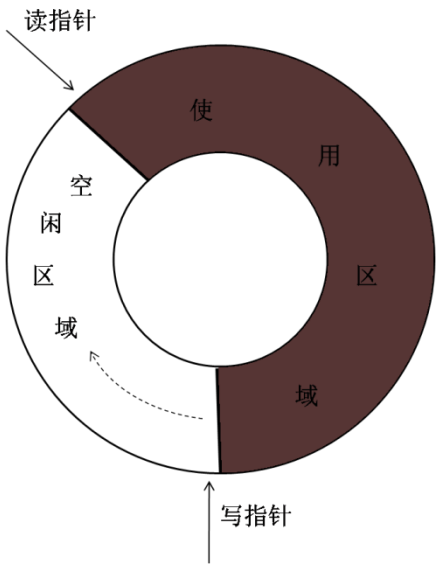


图 5-8 环形缓冲区示意图

5.3.2 线程间通信

5.3.2.1 LINUX 线程通信简介

读出系统在运行时有多个并发任务，需要实现多任务间的通信，而读出系统的这些并发任务间有大量资源需要共享，因此决定采用多线程的机制。在 Linux 操作系统下，线程仅仅被视为一个使用某些共享资源的进程。在内核中，它看起来就像是一个普通的进程，只是一个进程间共享资源的手段[43]。线程的共享资源的特点为系统的开发提供了方便，并节省了开销。

线程间的通信是数据获取读出系统的开发中一个重要环节。在读出系统中采用了消息队列和信号量两种方法来实现线程间的通信与同步。

5.3.2.2 信号量

信号量（也被称为信号灯）通常是在多个线程访问一个共同的但非共享的资源的情况下，用于同步各个线程之间的操作，通过信号量可以避免线程的饥饿和死锁[44]。在读出系统的环形缓冲区，以及多数需要通信的任务间都采用了信号量来协调线程的同步与互斥，达到对资源的保护和利用。

使用信号量时首先初始化，任务使用信号量时，如果信号量的值为非 0，则将信号量值减 1，允许任务的访问，如果信号量为 0，任务进入该信号量

的等待队列。当有任务释放信号量时，如果等待队伍不为空，则从等待队列中选出一个任务进入就绪态，信号量的值不变，选择算法可以基于优先级法或先进先出（FIFO）法。如果等待队列为空，则信号量加 1[20]。

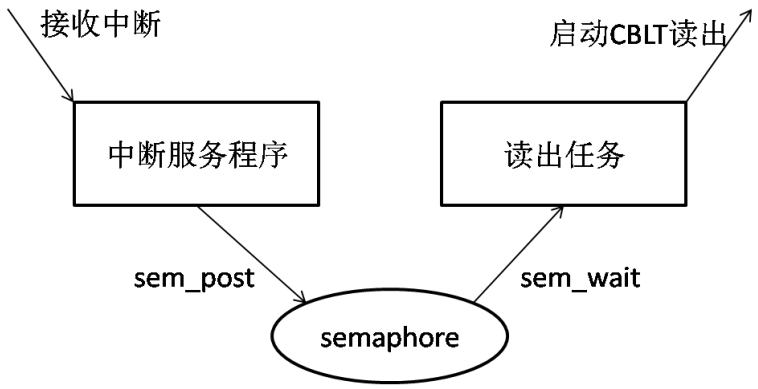


图 5-9 信号量使用示例

图 5-9 所示是读出系统中一个信号量使用的实例，中断服务程序一旦响应到中断，就会释放信号量，读出任务监测到信号量可用，说明数据已经准备好，便立即启动 CBLT 开始数据的读出。

5.3.2.3 消息队列

消息队列是将要进行通信的信息放置在一个预定义的消息结构中。线程生成的消息被放入一个由系统负责维护的消息队列中。访问消息队列的线程遵照 FIFO（先进先出）原则读取消息[44]。队列的访问具备锁机制。

消息队列在读出系统中的具体应用如图 5-10 所示：CBLT 读出任务每次将读出的一份原始数据片暂存在第一级环形缓冲区中，并在第一级消息队列中增加一个消息，这个消息的内容包含了原始数据片的长度。事例组装任务从该消息队列获得一个消息后，根据长度从第一级环形缓冲区中取出相应数据片进行组装，如果消息队列为空，则事例组装任务阻塞在该消息队列上。在这个过程中，CBLT 读出任务和事例组装任务是生产者-消费者关系。同样，事例组装任务每次将组装后的事例数据暂存在第二级环形缓冲区中，网络传输任务从中取出相应数据进行网络发送，在这个过程中，事例组装任务和网络传输任务是生产者-消费者关系[28]，任务间具体的时序实现如图 5-11 所示。

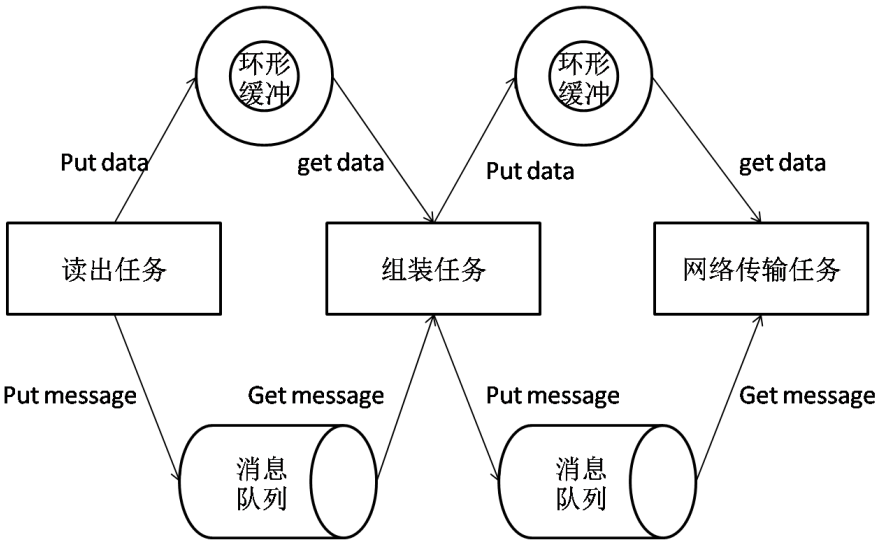


图 5-10 读出任务间数据通信示意图

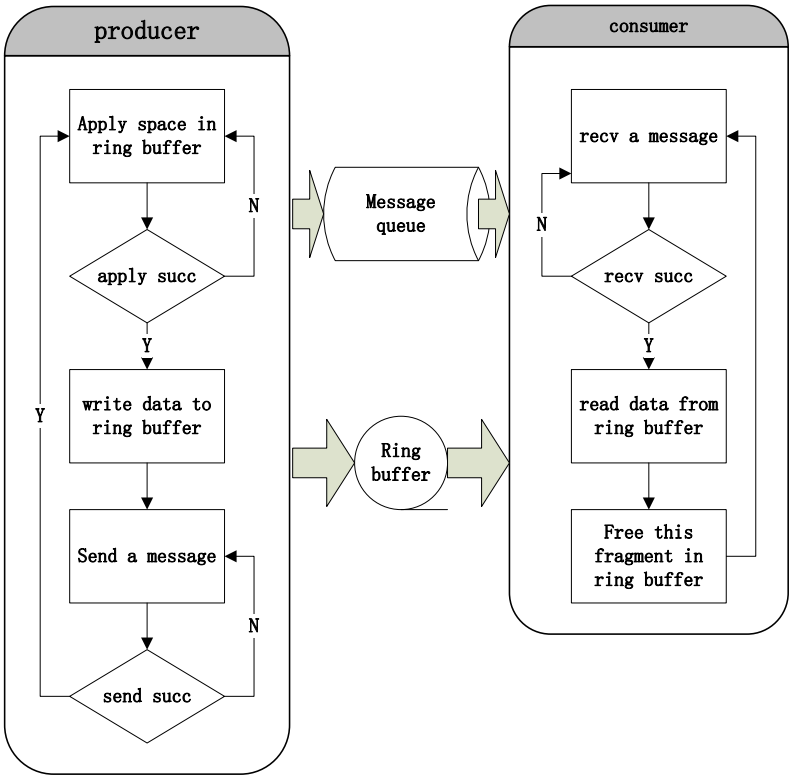


图 5-11 读出系统任务时序关系流程图

5.3.3 读出系统中各任务的具体实现

5.3.3.1 CBLT 读出任务

CBLT 读出任务的具体实现流程如图 5-12，首先要为存放 CBLT 读出

的数据在第一级环形缓冲区内申请一段内存空间，在接受到来自触发板的中断请求后，进行 CBLT 读取操作，并且将读取到的数据拷贝到第一级缓冲区内，之后向第一级消息队列里添加一个消息，以通知事例组装任务数据已经就绪。

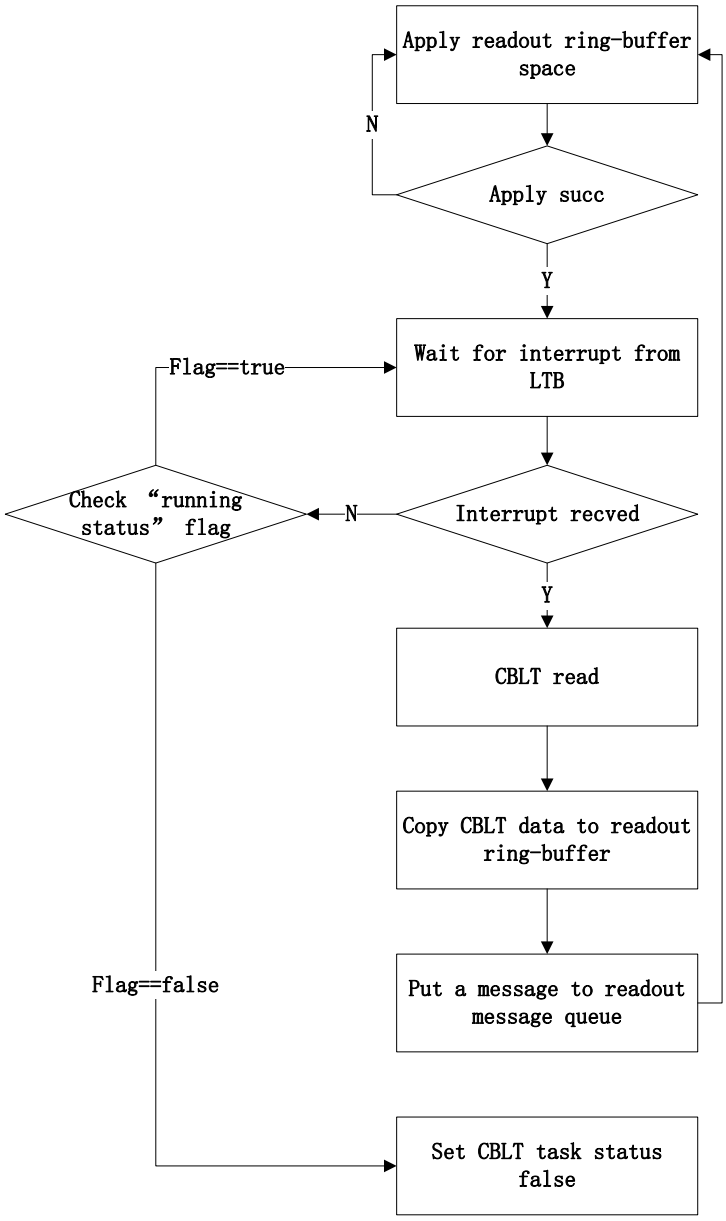


图 5-12 CBLT 读出任务流程图

5.3.3.2 事例组装任务

事例组装任务的实现流程如图 5-13 所示，组装任务从第一级消息队列得到数据准备好的通知，从第一级环形缓冲区内取到相应的数据进行检查

与组装，并且向第二级环形缓冲区申请一段内存空间用来存放组装好的事例数据，事例数据拷贝到第二级缓冲区以后，向第二级消息队列添加一个消息，以通知网络传输任务事例数据已经就绪。事例检查与组装的具体过程将在 5.4 节中进行介绍。

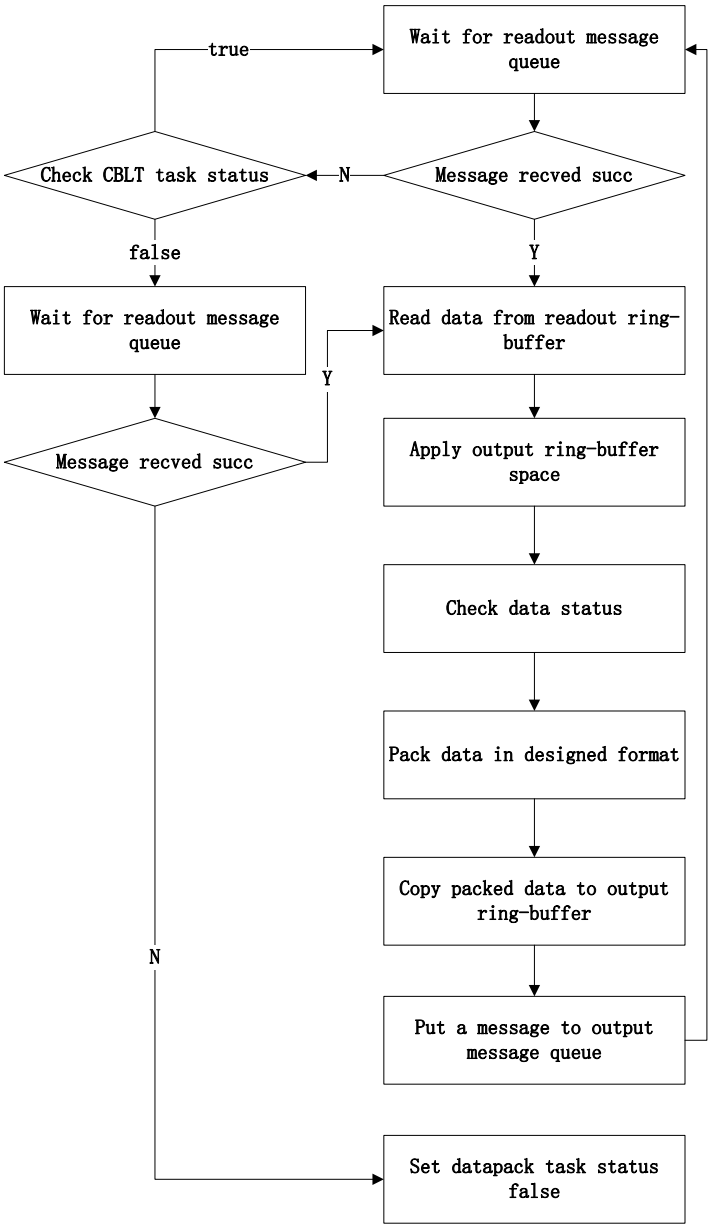


图 5-13 组装任务流程图

5.3.3.3 网络传输任务

网络传输任务的具体实现如图 5-14 所示，网络传输任务从第二级消息队列得到事例数据准备好的通知，从第二级环形缓冲区内取到相应的事例

数据，将按照 eformat 格式构造好的完整事例通过 EFIO 协议发送给 EFD。

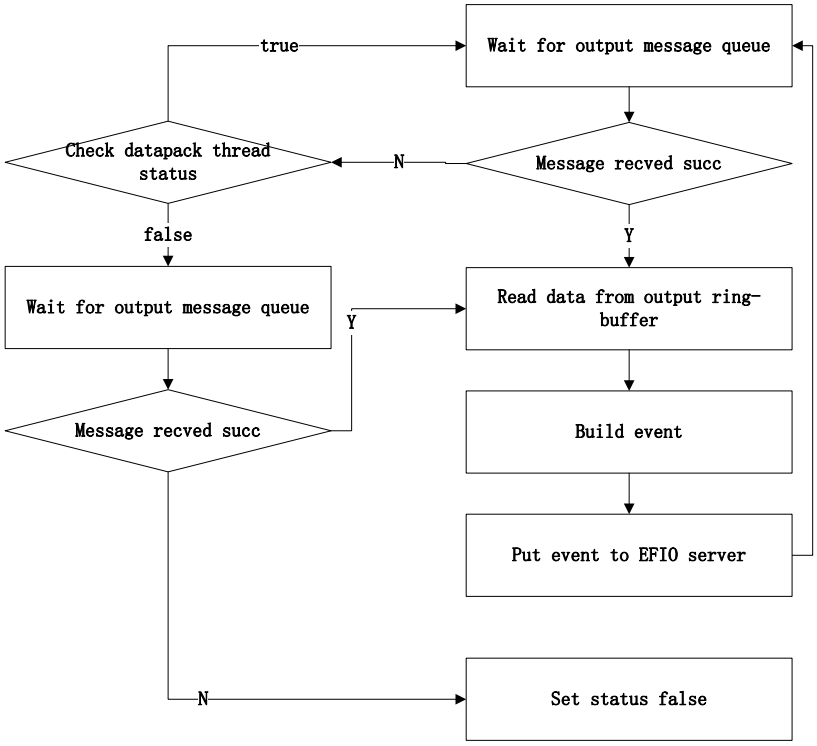


图 5-14 网络发送任务流程图

5.3.3.4 数据流监测任务

数据流监测任务主要流程如图 5-15 所示，首先在系统进入正式运行状态之前复位所有监测参数，当进入正式运行状态后，每经过系统指定的时间间隔（如 5 秒）更新所有监测参数的值，部分监测参数的值可以直接获得，例如已读取事例数、已组装事例数等；另有部分参数的值是接收到更新指令时通过计算获取，例如平均事例率，平均数据带宽等。数据流监测任务的应用将在 5.5 节中具体描述。

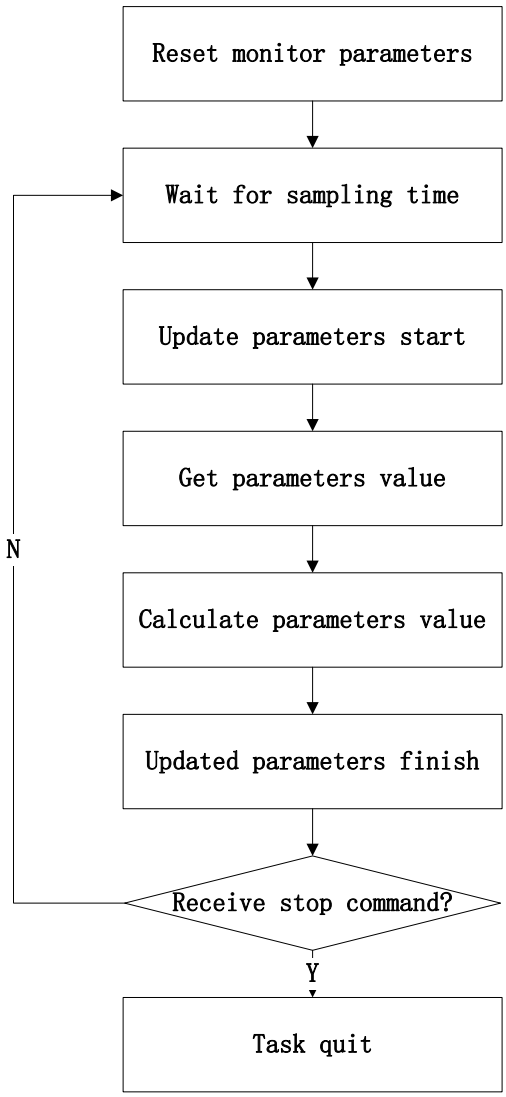


图 5-15 数据流监测任务流程图

5.4 数据检查与组装

5.4.1 DAQ 事例格式

DAQ 的事例格式需要满足以下需求[45]：

- 依据特定的数据获取配置，事例格式必须允许增大或者减小一个事例的尺寸
- 事例格式不能强制规定事例的最大和最小尺寸
- 事例格式必须提供足够的信息允许对事例进行一致性等健康检查

- 事例格式中的信息最好不超过完整事例尺寸的 20%
- 事例格式最好模块化，以提供足够的灵活性
- 事例格式中的基本单元必须是片段(fragment)。
- 事例片段的组织结构必须相同
- 事例格式最好提供一个事例头部（event header）
- 事例格式中的事例头部最好包含事例标志(event identifier)和触发类型(trigger type)
- 事例格式最好能提供一种手段表征数据是否在传输过程中受到损害
- 事例格式最好能提供一种手段表征数据是否由于硬件原因受到损害

大亚湾实验的数据获取系统定义的一个完整事例的格式如图 5-16 所示。一个完整的事例由事例头部以及参加取数的所有 ROM(ReadOut Module)的数据片段组成，每个 ROM 的数据片段也由包含了格式信息的数据头部和具体的数据内容组成。

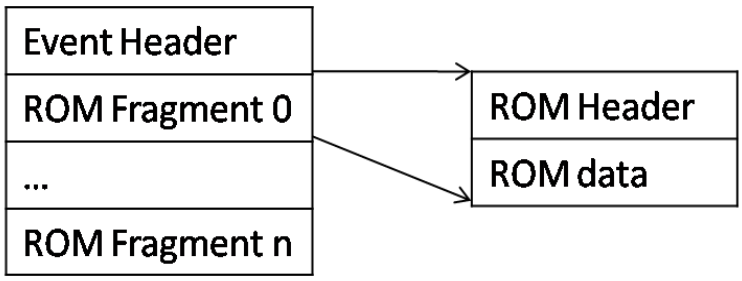


图 5-16 大亚湾实验 DAQ 事例格式

事例头部的格式如图 5-17 所示，主要包含了事例数据头部的标志、事例格式版本号、事例数据和事例头部信息的大小尺寸、数据源标识、run number、事例编号、数据类型与数据状态。其中数据类型里记录了取数模式的信息与相关参数信息，而数据状态里记录了数据流的重要信息，例如错误标志等。

Start of Header Marker
Format Version Number
Header Size / Total Size
Source Identifier
Run Number
Event ID
Detector Data Type
Data Status

图 5-17 事例头部格式

读出插件 ROM 头部格式如图 5-18 所示，主要由三部分组成，ROM 头部标志、数据段和头部信息的大小尺寸以及数据来源标识。

Start of Header Marker
Header Size / Total Size
Module Identifier

图 5-18 电子学读出插件数据头部格式

5.4.2 电子学数据格式

根据事例格式的要求以及为了数据检查的方便，制定了统一的电子学读出数据格式，如表 5-1 所示。所有电子学读出插件都应当在这个框架下组织读出数据的格式，在此基础上各不同类型的插件可以根据实际需要灵活使用其中的保留位。其中：

- GEO 是每个插件的地理地址，即该插件在 VME 机箱中的插槽号
- Module Type 是每个插件的插件类型，在大亚湾数据格式中为每种类型的电子学插件指定了各自对应的插件类型
- Status Check Info 用于比较 PMT 读出电子学中 FEE 与 LTB 板触发号的同步性
- Trigger Num 是触发号，是取真实触发号的低 8 位，即 L1ID&0xff
- Event Length 和 Total Length 所代表的长度均以字节为单位，并且

长度中都包含了所在结构的头尾两个整型的长度。

表 5-1 电子学读出数据格式

		31	30	29	28	27~23	22~20	19~16	15~8	7~0
Module Header		0	0	1	X	GEO		ModuleType		
Trigger 1	Header	0	1	0	X				Status Check Info	Trigger Num
	Data	0	0	0	X					
	Data	0	0	0	X					
	Footer	1	0	1	X				Event Length	
Trigger 2	Header	0	1	0	X				Status Check	Trigger Num
	Data	0	0	0	X					
	Data	0	0	0	X					
	Footer	1	0	1	X				Event Length	
... Trigger N										
Module Footer		1	1	0	X	GEO		Module package Total Length		

5.4.3 数据检查

数据获取读出系统需要对读出的数据进行初步的检查，目的是尽可能在第一时间捕捉到数据的异常，及时的将错误信息上报，相关人员可以尽早发现问题并解决，从而避免浪费宝贵的取数时间，有效提高取数的效率和数据的质量。但是在读出系统中的数据检验又不能够太过复杂，因为 PowerPC 的处理能力是有限的，不能因为数据检查占用过多 CPU 资源而影响取数的效率，因此读出系统不会对电子学读出数据的具体内容进行逐项的检查（这些内容会在 DQM(Data Quality Monitoring)系统中进行相对及时的处理），读出系统的数据检查内容主要是针对电子学读出数据的格式以及一些重要的读出信息而进行的。

根据表 5-1 所示电子学读出数据结构，读出系统中数据检查的内容至少需要有以下几个方面：

- 对数据头部、尾部以及真实数据的标志位的检查。通过标志位的检查，可以确保读出数据是有序组织的，根据指定格式读出的，从而确保读出逻辑的正确性。
- 校验各个读出插件数据中的地理地址位，与电子学配置数据库中的相应地理地址记录进行比较，必须确保一致，否则数据中一定发生了某项严重的错误，如丢失部分插件数据等。

- 对于 PMT 电子学系统的读出数据，需要检查各个电子学插件读出数据中的事例触发号是否一致，并且与读出软件自身记录的触发号是否一致。这是一项很重要的校验内容，是确保读出过程中电子学各插件保持同步的必要条件。
- 对于 FEE 插件，需要额外检查 check 信息位[46]。这一信息的内容可以通知读出系统 FEE 插件和触发板 LTB 之间的同步是否出现异常。
- 对于 LTB 插件，需要额外检查中断超时位[47]。正常情况下，FEE 读出数据准备好以后，会给 LTB 发送一个信号，当 LTB 自身数据准备好并且收到 FEE 发来的信号时，LTB 会给出中断通知读出系统取数。而如果 FEE 因为某种原因，没有给 LTB 就绪信号，LTB 如果死等会将整个系统陷入阻塞状态。因此 LTB 设置了超时机制，如果超过一定的预期时间没有收到 FEE 的就绪信号，即发出中断，但是在 LTB 的数据中置上中断超时位。一旦读出系统在 LTB 的数据中检查到被置上的中断超时位，即说明此时系统中已经由于某种原因导致了 FEE 与 LTB 数据的不同步。

5.4.4 数据组装

5.4.1 节和 5.4.2 节中已分别介绍了 DAQ 事例格式与电子学数据格式。电子学读出数据的格式必须经过一定的拆解与组装，才能够实现符合要求的 DAQ 事例格式，这一数据拆解与组装的过程，是在数据获取读出系统中完成的。

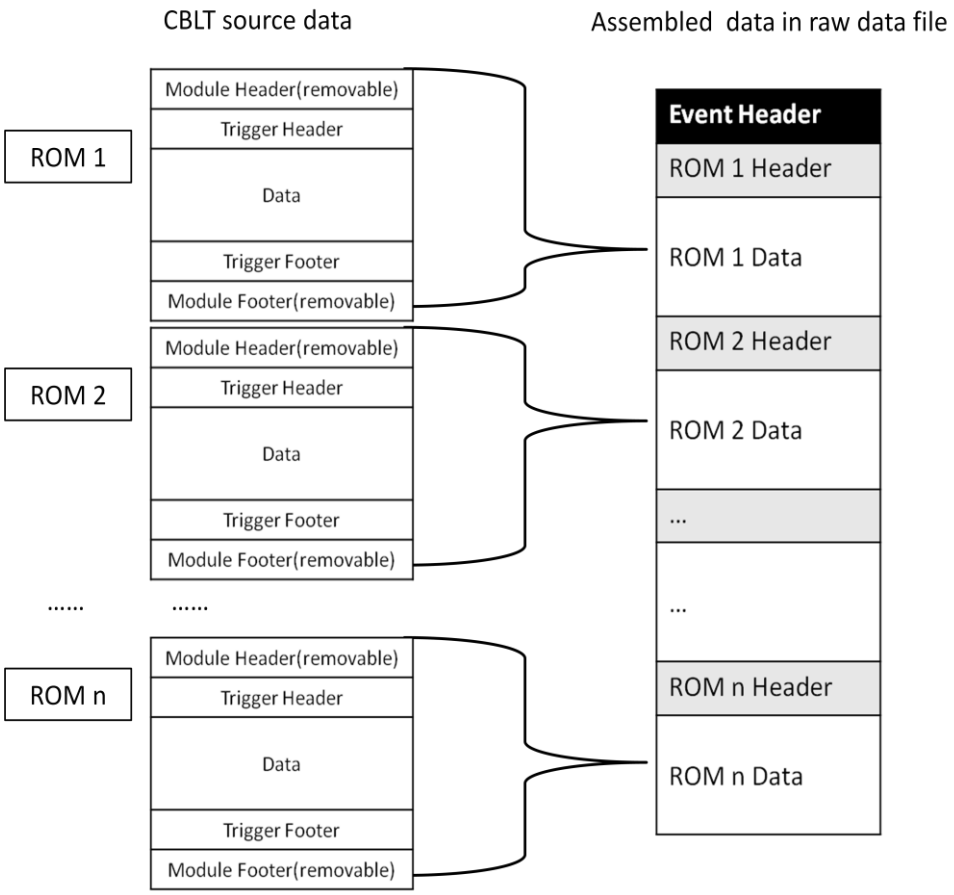


图 5-19 读出系统事例组装示意图

如图 5-19 所示，左边是 CBLT 读出的电子学源数据，右边是在读出系统中按照指定格式组装好的事例。读出系统获取到 CBLT 源数据以后，先将读出数据按照每个插件为一组拆解开，之后给每个插件的数据加上对应的 ROM 头部信息，组成 ROM 数据片段，再将 ROM 数据片段按照原来的插件顺序排列组合起来，最后在事例数据的最前面加上事例头部信息。这样就完成了符合要求的一个完整的事例数据。每一次读出的电子学数据在读出系统中都会经过这样的处理流程，这些组装好的事例数据会通过 EFD 组件送给 SFO 组件，由 SFO 加上文件分隔符与相应的文件信息后进行存盘处理。

对于多事例读出的情况，如图 5-20 所示，左边是 CBLT 读出的电子学源数据，每一个插件数据中包含了 N 次 trigger 的数据，一次 trigger 数据对应于最终的一个事例数据。在多事例读出的情况下，先要将读出数据按照

每个插件的每个 trigger 数据拆解开，再把每个事例按照事例格式进行组装，因此这样的一次数据读出经过拆解与组装最终会得到 N 个组装好的事例数据。这些数据同样被送往 SFO 进行存盘。

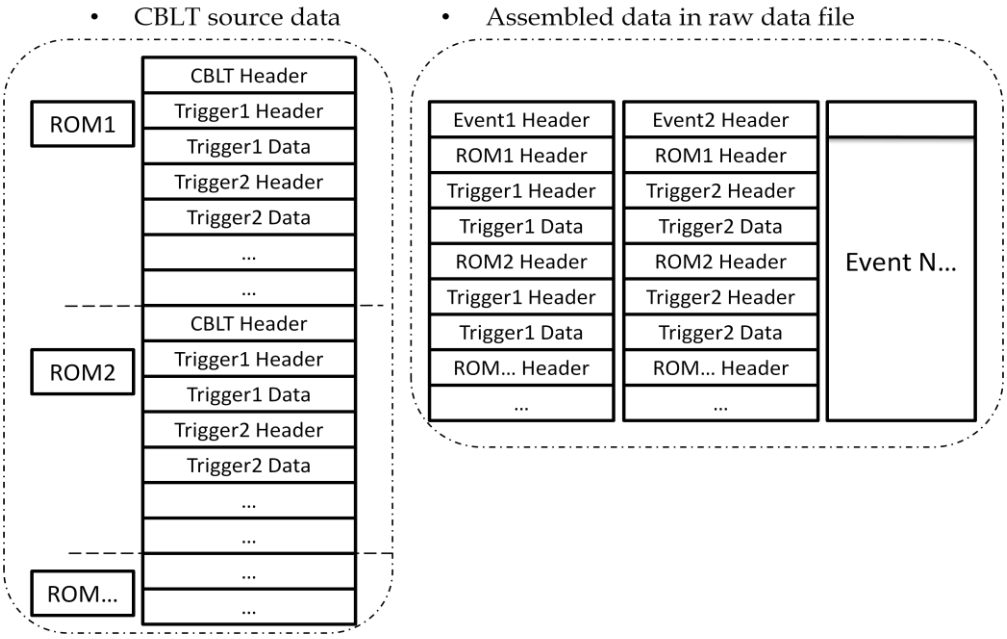
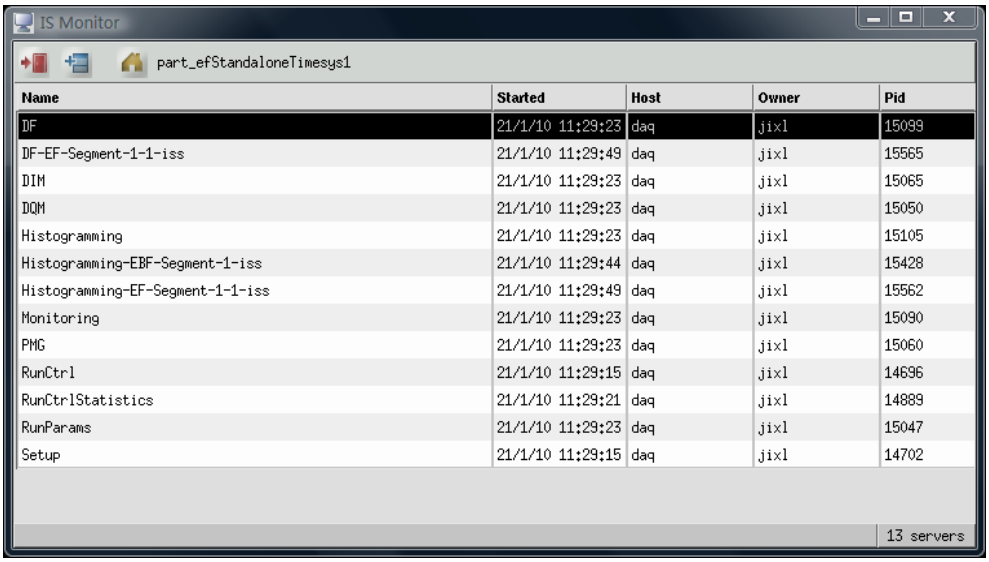


图 5-20 读出系统多事例读出组装示意图

5.5 数据流监测

5.5.1 is_monitor

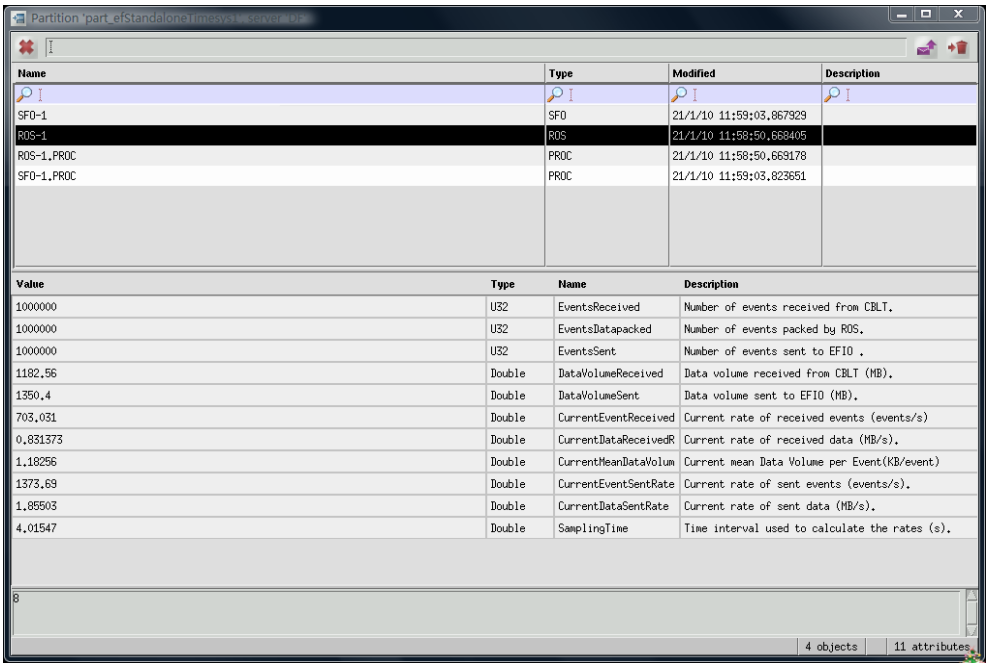
is_monitor 程序是在 XWindow 之上开发，通过 IS 服务的接口，可以查看所有 is_server 上暂存的信息，能够很好的帮助开发人员和维护人员调试系统。程序显示主界面如图 5-21 所示，界面中列出了所有的 is_server，不同 is_server 满足不同种类的信息共享。DF 保存数据流相关的信息，DF-EF 保存 EFD 相关信息，Histogramming 保存所有的完整直方图，Monitoring 保存有关事例监测服务的信息，RunCtrl 保存各个控制程序的状态信息，RunParams 保存 run 号、运行模式等运行参数，等等。



Name	Started	Host	Owner	Pid
DF	21/1/10 11:29:23	daq	jixl	15099
DF-EF-Segment-1-1-iss	21/1/10 11:29:49	daq	jixl	15565
DIM	21/1/10 11:29:23	daq	jixl	15065
DQM	21/1/10 11:29:23	daq	jixl	15050
Histogramming	21/1/10 11:29:23	daq	jixl	15105
Histogramming-EBF-Segment-1-iss	21/1/10 11:29:44	daq	jixl	15428
Histogramming-EF-Segment-1-1-iss	21/1/10 11:29:49	daq	jixl	15562
Monitoring	21/1/10 11:29:23	daq	jixl	15090
PMG	21/1/10 11:29:23	daq	jixl	15060
RunCtrl	21/1/10 11:29:15	daq	jixl	14696
RunCtrlStatistics	21/1/10 11:29:21	daq	jixl	14889
RunParams	21/1/10 11:29:23	daq	jixl	15047
Setup	21/1/10 11:29:15	daq	jixl	14702

图 5-21 is_monitor 主界面

双击某个 is_server 后就可以打开一个子窗口显示该 server 的内部信息。如 DF 的子窗口如图 5-22 所示，窗口上半部分显示了数据流相关的进程 SFO、ROS 的信息，下半部分显示出了该进程发布的信息内容，如处理的事例数目、采样时间等等。



Name	Type	Modified	Description
SFO-1	SFO	21/1/10 11:59:03.867929	
ROS-1	ROS	21/1/10 11:58:50.668405	
ROS-1.PROC	PROC	21/1/10 11:58:50.669178	
SFO-1.PROC	PROC	21/1/10 11:59:03.823651	

Value	Type	Name	Description
1000000	U32	EventsReceived	Number of events received from CBLT.
1000000	U32	EventsDatapacked	Number of events packed by ROS.
1000000	U32	EventsSent	Number of events sent to EFIO .
1182.56	Double	DataVolumeReceived	Data volume received from CBLT (MB).
1350.4	Double	DataVolumeSent	Data volume sent to EFIO (MB).
703.031	Double	CurrentEventReceived	Current rate of received events (events/s)
0.831373	Double	CurrentDataReceivedR	Current rate of received data (MB/s).
1.18256	Double	CurrentMeanDataVolum	Current mean Data Volume per Event(KB/event)
1373.69	Double	CurrentEventSentRate	Current rate of sent events (events/s).
1.85503	Double	CurrentDataSentRate	Current rate of sent data (MB/s).
4.01547	Double	SamplingTime	Time interval used to calculate the rates (s).

图 5-22 is_monitor 子界面

5.5.2 读出系统数据流监测

从图 5-22 所示的 is_monitor 子界面中可以看到，ROS 的数据流监测提供了接收到的事例数目、完成组装的事例数目、网络发送完成的事例数目、平均接收到的事例大小、平均网络发送的事例大小、在采样区间内接收数据的事例率与数据带宽、网络发送的事例率与数据带宽等等。这些参数的定义类型、名称与描述都在对应的 XML 文件中组织，运行时数值的更新在读出程序中都有相应的实现，读出系统经过每个指定的时间间隔计算相关数值并把结果提交给 is_monitor 进行显示。这些数值的监测对于开发人员观察系统工作状态，调试系统性能有着很大帮助。

同时，DAQ 的用户界面（IGUI）还提供了更直观的方便用户监测数据流情况的窗口，如图 5-23，它是基于 java 语言开发的。可以看到图 5-22 中所示的 ROS 数据流监测的参数在这个窗口的右上部分中都有显示，而它相对于 is_monitor 的特点是可以指定某个参数绘出一段时间内的变化曲线，方便用户（例如值班人员）随时查看运行信息。

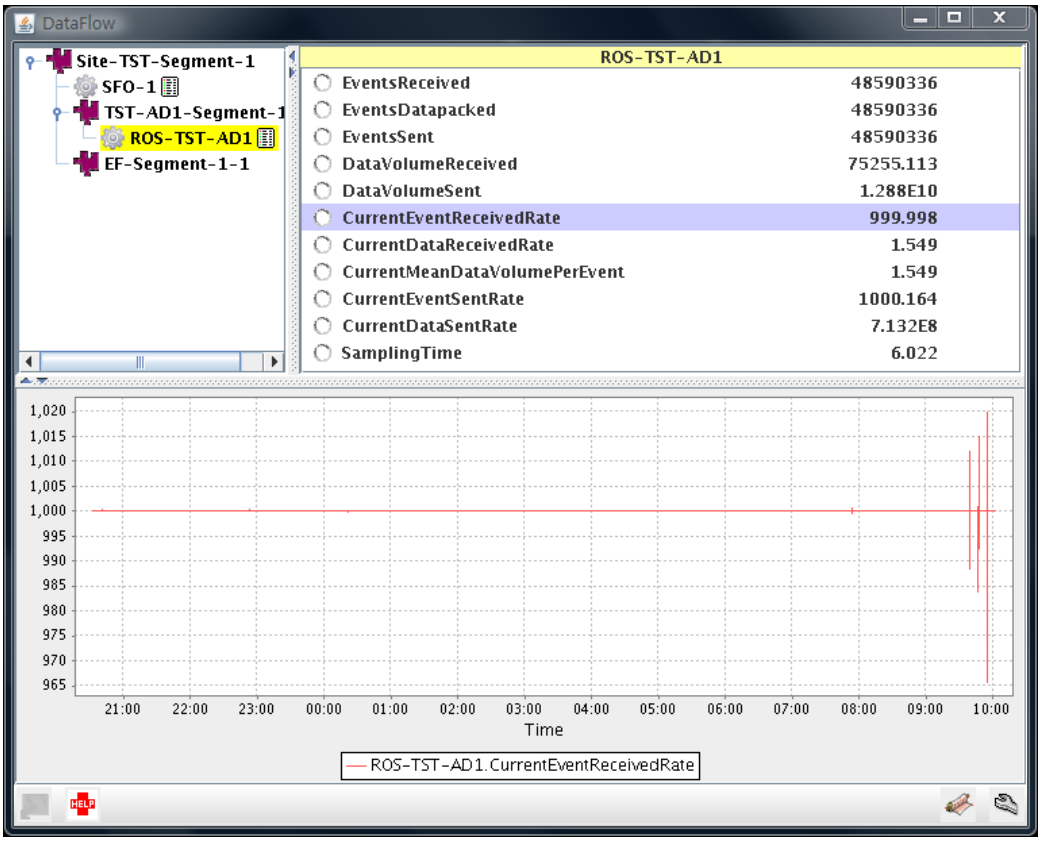


图 5-23 读出系统数据流监测界面

5.6 出错处理

5.6.1 消息报告显示程序

mrs_monitor 程序基于 java 开发，用来监视所有的错误消息，也可以集成在 DAQ 的主控制界面上。程序界面如图 5-24 所示，在界面中可以设置过滤选项，如消息来源，显示级别等，并且还可以设置消息格式，如消息长度、数量等。

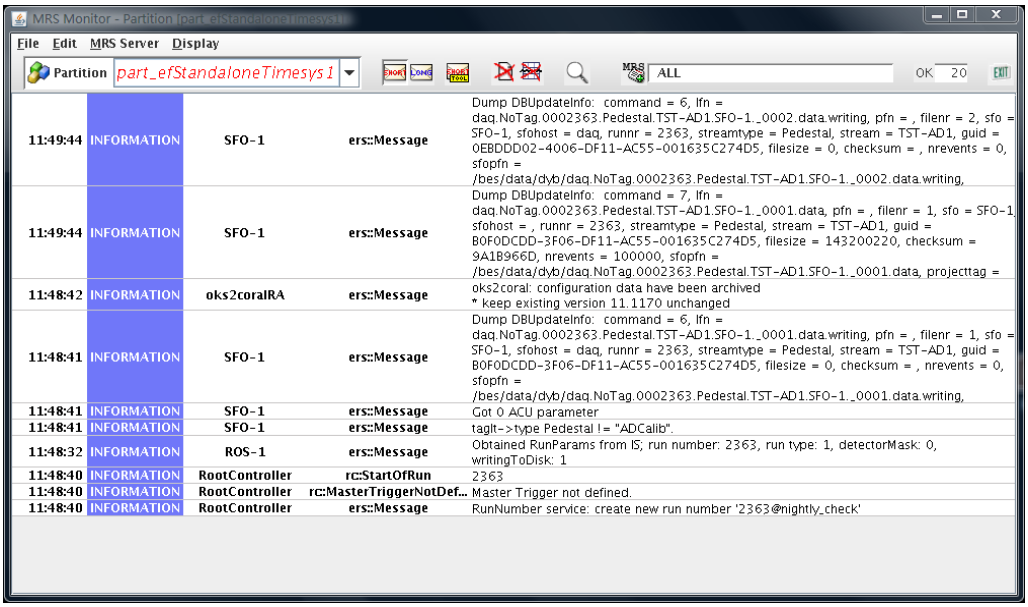


图 5-24 消息报告显示程序界面

5.6.2 读出系统的错误处理

在读出系统中可能发生的错误有来自读出系统自身的错误，如网络连接失败，缓冲区创建失败等；也有可能是来自前端电子学系统的错误，如配置插件寄存器出错等。读出系统自身的错误通常只会发生在调试过程中，一旦得到及时解决，在运行过程中出错的概率很小。而在系统的运行期间，发生概率最大的是来自前端电子学的错误，虽然严格的说，这不属于读出系统的错误，但是读出系统需要具有处理这种错误的能力。因为读出系统是数据获取系统与电子学系统的接口，在系统初始化与运行的过程中，电子学系统的异常必须通过读出系统进行捕获。读出系统能够检测到的电子

学异常通常有两大类型，分别是电子学硬件配置失败与电子学数据校验错误。

对于电子学配置过程失败导致的错误，读出系统必须向 MRS 发出错误消息，之后立即中止当前的操作。因为配置过程一旦失败，即便开始取数，取到的数据也会有各种不可预期的情况产生，这样的数据是完全没有意义的。所以发生配置错误时，必须及时地报告错误并且尽可能详细的报告错误的来源例如产生错误的插件编号、具体出错的寄存器地址等，等待相关人员找到问题根源并做出修复后再重新开始取数。

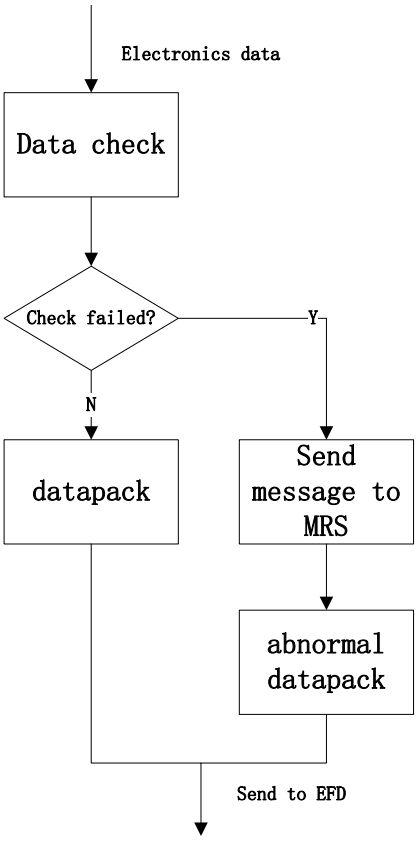


图 5-25 读出系统的数据检查结果处理示意图

而对于电子学数据的读取校验过程中产生的错误，通常有两种处理方法：直接丢弃异常数据或者是仍然保留这些出错数据。目前选择的处理方法是保留出错数据：即一旦检查出错，读出系统向 MRS 发出错误消息，之后对出错数据进行异常数据组装，组装之后的流程和正常情形是一样的，也会将数据通过 EFD 组件送给 SFO 进行存盘，如图 5-25 所示。所谓异常

数据组装，有别于5.4.4节中介绍的数据组装的格式，因为数据中存在异常，很可能无法按照每个ROM的格式分别进行组装，此种情形之下，如图 5-26所示，对已经发现异常的 CBLT 源数据不做任何的拆解处理，而仅仅在源数据之前加上每次事例的事例头部。需要注意的是在事例头部的 data status处置上标志位，表示该事例是个异常事例。保留原始数据的好处是，可以帮助相关人员快速定位出错的原因，提高解决问题的效率。同时这些异常数据中虽然存在某些错误或者异常，但是可能也会保留有一些有用的信息，仍然可以提供给相关人员进行分析。

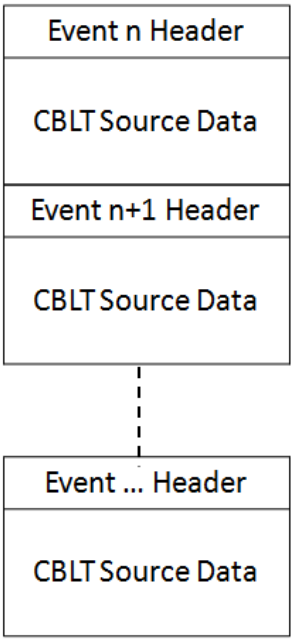


图 5-26 异常数据组装示意图

5.7 启动与退出的时序考虑

取数正式开始的标志是StartTrigger命令。通过用户界面上的start按钮，RC 组件负责传送 StartTrigger 命令给相应的 DAQ 组件。读出系统收到StartTrigger 后，立即通过写触发板寄存器的命令打开触发源，电子学系统进入取数状态。为了确保不丢失刚开始取数时的事例，读出系统的数据处理相关的任务（CBLT 读出任务、事例组装任务、网络传输任务）必须在StartTrigger命令到来之前就绪。相反，如果StartTrigger命令到来时，读出

系统中的数据处理任务尚未就绪，但触发源已经打开，电子学数据缓冲区很快会溢出，从而导致数据的丢失，这种现象是设计读出系统时必须考虑到要避免的。因此根据状态机的配置，在读出系统收到 PrepareForRun 的命令时，即启动所有数据处理的任务，因为此时触发源尚未打开，LTB 板也不会有中断请求送给读出系统，所有数据处理任务会处于阻塞状态。一旦读出系统收到 StartTrigger 的命令，触发源被打开，电子学系统随即进入取数状态，相应的读出系统中的数据处理任务也会被唤醒进入工作状态。这样的启动时序的设计能够确保取数开始时的每个数据都被 DAQ 系统记录下来而不会造成数据溢出或者丢失。

取数暂停或结束的标志是 StopTrigger 信号。通过用户界面上的 stop 按钮，RC 组件负责传送 StopTrigger 命令给相应的 DAQ 组件。读出系统收到 StopTrigger 命令后，通过写触发板寄存器的命令关闭 LTB 的触发源，使得电子学系统停止取数。而此时在前端电子学插件中可能还有缓存的事例，并且读出系统的本地内存中也可能还有尚未处理的事例数据，因此为了保证没有事例的丢失，必须严格按照如图 5-27 所示的时序退出，即关闭触发源以后，CBLT 任务必须继续运行，直到读空电子学插件的缓冲区内的数据后方可退出；而事例组装任务必须在检测到 CBLT 任务已经退出并且第一级环形缓冲区内的数据已经被读空后，方可退出；同样，网络传输任务也必须在检测到事例组装任务已经退出同时第二级环形缓冲区内的数据已经被读空后，才可以退出。经过实验检验，严格按照此时序退出的读出系统能够保证退出时没有事例的丢失。

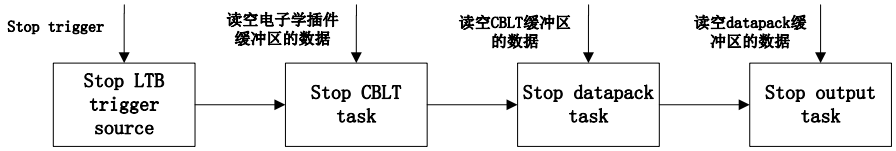


图 5-27 读出系统退出时序

5.8 多种运行模式的实现

2.2.1.3 节中介绍了大亚湾实验对于运行模式的需求。本节将详细分析

在读出系统中，针对不同运行模式的具体实现方案。由于截至目前为止水池刻度模式与矿物油监测模式的取数方案尚未确定，所以本节中不讨论这两种运行模式。

5.8.1 物理取数模式

物理取数是实验中最常用的取数模式，它的工作流程相对简单，读出系统从电子学配置数据库读取物理取数的相关配置参数，同时从配置数据库里获得相应的触发源类型配置触发源。只要接收 StartTrigger 命令就打开触发源开始取数，收到 StopTrigger 命令就关闭触发源而停止取数。物理取数模式下读出系统工作流程如图 5-28 所示。

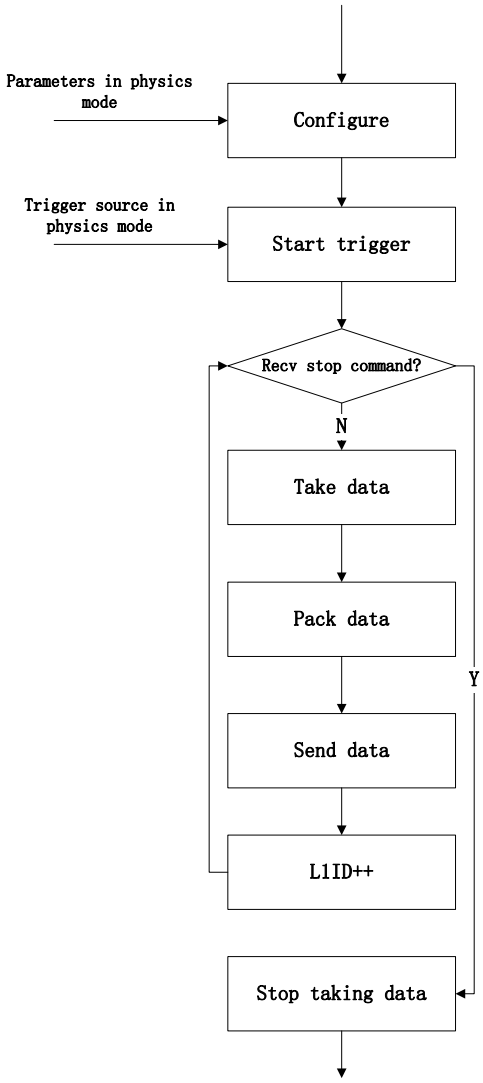


图 5-28 物理取数模式下读出系统工作流程图

5.8.2 FEE 电子学诊断模式

如图 5-29 所示，读出系统需要从配置数据库读取电子学诊断模式的相关参数对电子学进行配置，电子学诊断模式采用 manual trigger 模式。电子学诊断模式的工作流程比正常取数要略复杂一些，有多个不同的刻度 DAC 值，每设置一种刻度 DAC 值不变的情况下，循环取数一定次数，再换到下一个刻度 DAC 值，直到取完要求的所有 DAC 的值对应的数据为止。当读完预设的最后一个 DAC 值的最后一个事例后，读出系统停止取数。DAQ 开发了 supervisor 组件，作为基于 RC 开发的超级管理者，supervisor 通过与 IS 组件的信息沟通，确定读出系统已经完成取数后，会向整个 DAQ 系统发布 stop 命令，从而确保 DAQ 系统各组件都能够同步的退出运行状态。

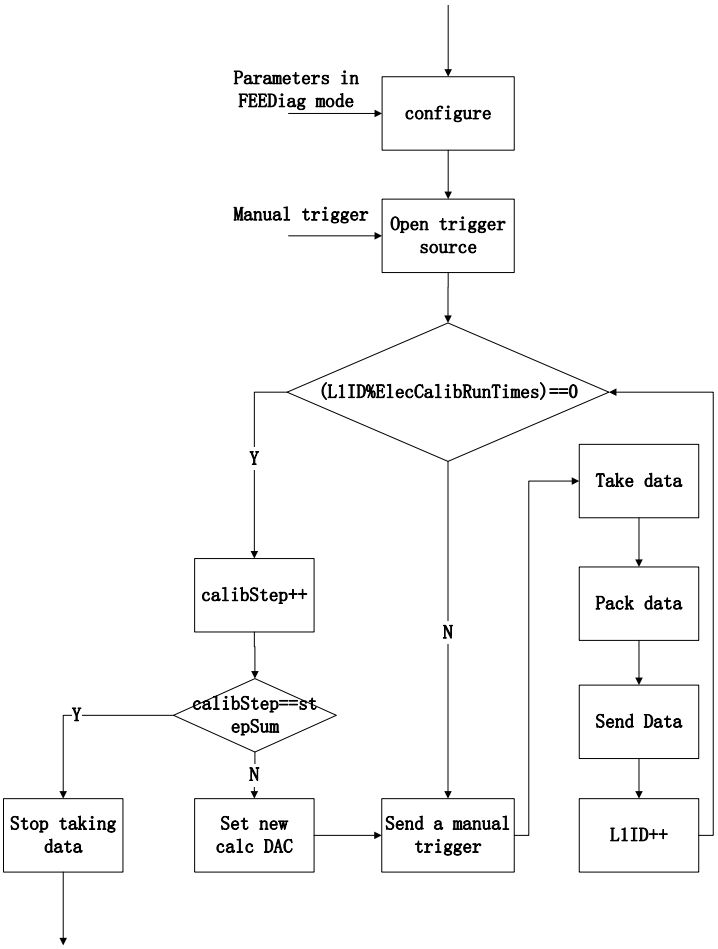


图 5-29 FEE 电子学诊断模式下读出系统工作流程图

5.8.3 台阶模式

台阶模式实现流程如图 5-30 所示，整个过程类似物理取数模式，它的配置参数也是独立的，台阶模式通常使用周期性触发，并且预先在 XML 文件中设置了每次运行的事例数要求。因此台阶模式下，读出系统自己统计事例数目，一旦达到预设值，则停止取数。与电子学诊断模式类似，同样是由 supervisor 组件负责整个系统的同步退出。

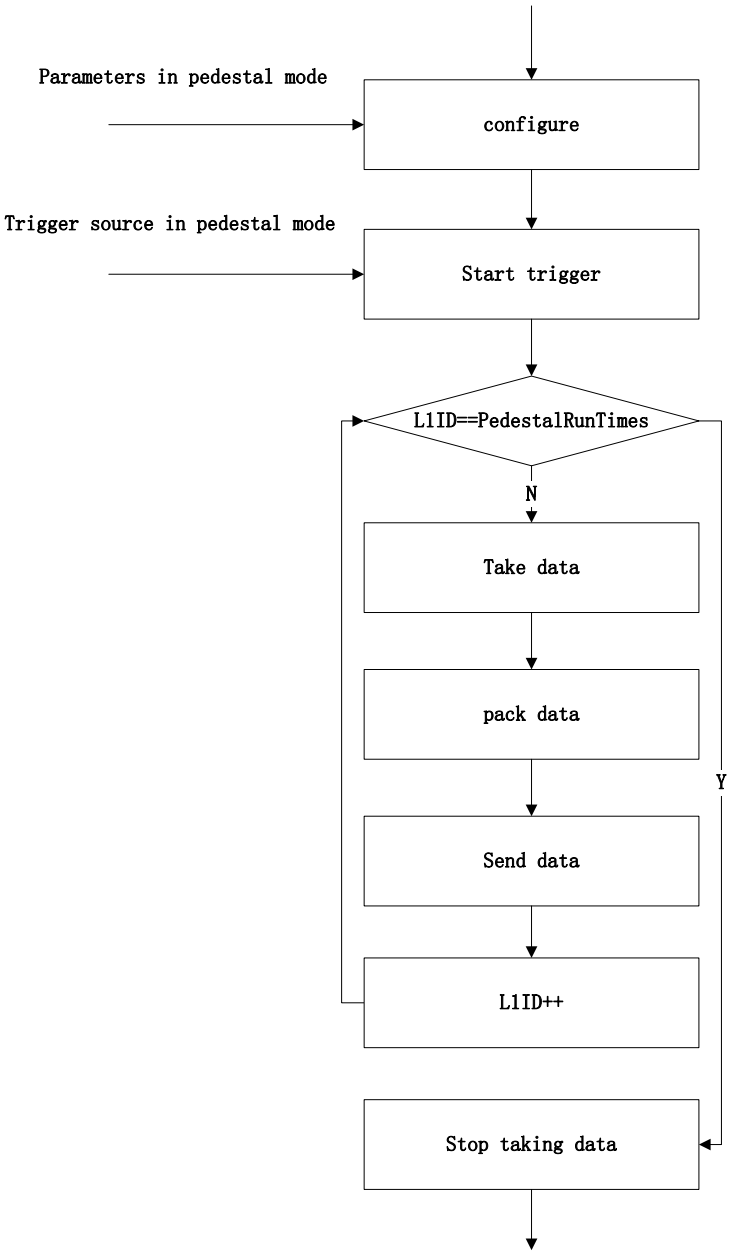


图 5-30 台阶取数模式下读出系统工作流程图

5.8.4 AD 刻度模式

AD 刻度模式的特点是，一次刻度的过程是由多个 RUN 组成。实现流程如图 5-31 所示，在运行开始前根据配置数据库的内容配置好电子学插件后，每次收到 Start 命令开始一个新 run 时，需要从 IS 获取当前 RUN 的触发模式信息，以此决定本次 RUN 的触发源类型。除此之外的基本实现过程与物理取数类似，只是每个 RUN 结束以后，如果收到 Start 命令则开始下一个 RUN，而如果收到 Unconfig 命令则说明本次刻度取数完毕。整个取数过程中的状态控制由 supervisor 组件完成，DAQ 系统通过 DIM[48]与刻度系统沟通，IS 组件则负责与 DIM 的信息交流，supervisor 组件以从 IS 获得的运行信息为判断依据作出整体的运行控制。

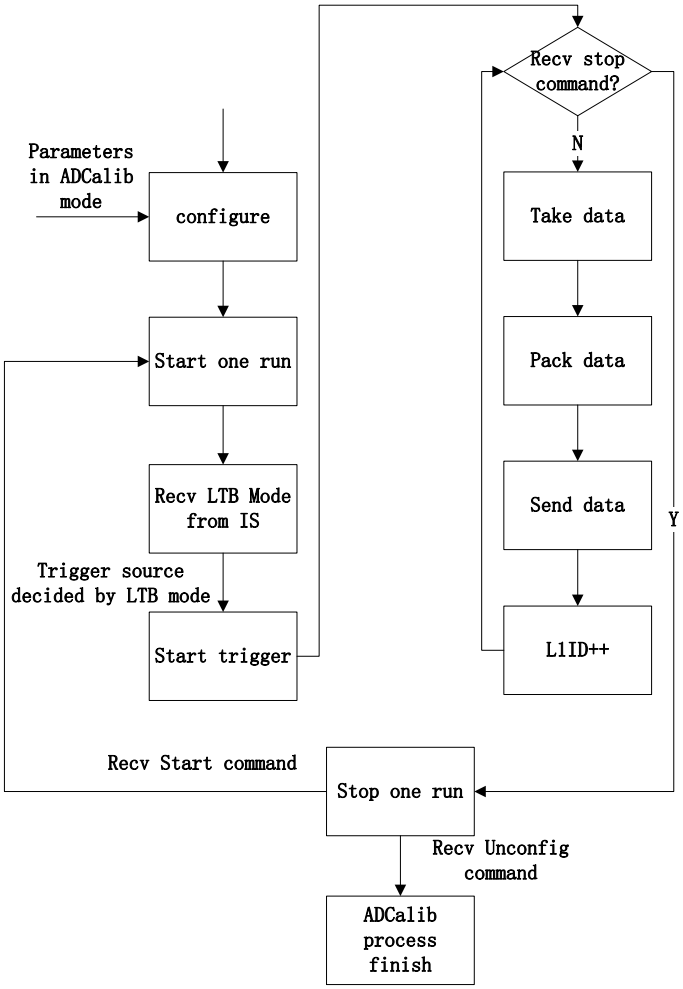


图 5-31 AD 刻度模式下读出系统工作流程图

5.8.5 小结

综上所述，针对不同的运行模式，读出系统的工作流程主要差别有：

- 从电子学配置数据库读取不同运行模式的参数，包括对应的触发类型、电子学阈值等
- 从 IS 组件获取不同的信息。例如，在刻度模式下，ROS 需要在每个 RUN 之前从 IS 获取触发源信息等相关刻度参数
- 对应不同的运行模式，ROS 配置电子学插件的步骤略有不同
- 对应于不同的运行模式，ROS 的取数流程也有一定的区别。例如在电子学诊断模式下，ROS 需要计数来判断是否需要改变输入 DAC 值或是是否该停止取数。
- 在数据格式中有相关的条目需要区别对待。例如在电子学诊断模式下，ROS 需要将每个点的输入 DAC 值填入数据头部中。
- 在不同的运行模式下，ROS 需要预先考虑到每种探测器类型对应于每种运行模式的不同表现。例如，在 AD 刻度模式下，水池系统需要运行物理取数模式，在确定需求的前提下，ROS 必须预先设定好对应关系。

5.9 本章小结

本章详细描述了大亚湾实验数据获取读出系统的具体实现方案，包括状态转换的实现、电子学配置数据库的实现、多任务并行系统的实现机制、数据检查与组装的实现过程、数据流监测的方案、错误处理机制、启动与退出时的时序考虑以及多种运行模式在读出系统中的实现流程等。在制定好系统的实现方案并解决了其中的技术细节后，在下一章中将以实例来分析数据获取系统的运行情况。

6 单 AD 机箱的数据获取系统

大亚湾中微子实验的设计方案要求各个探测器采用独立的前端电子学系统、触发系统以及数据读出系统。因此，用来读取反中微子探测器探测到的信号的 AD 机箱必须拥有完整的电子学、触发以及 DAQ 系统，并且接收来自时钟系统的时钟信号。大亚湾中微子实验的读出电子学有两大类，PMT 读出电子学与 RPC 读出电子学，AD 机箱的前端电子学系统采用 PMT 读出电子学的设计。

6.1 硬件组成

6.1.1 PMT 读出电子学简介

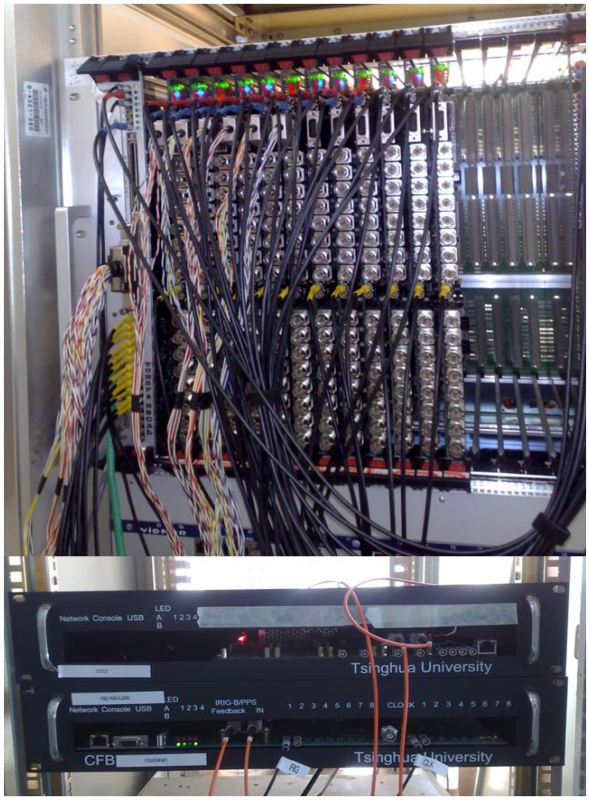


图 6-1 AD 电子学机箱硬件图

如图 6-1 所示，这是一个 AD 探测器的读出电子学机箱硬件图。整个硬件系统由一个 9U VME 机箱和两块独立的时钟插件组成。其中，VME 机箱内从左到右分布了：一块系统控制器 MVME5500，一块触发板 LTB（V2.0），12 块前端电子学读出插件 FEE（V1.3）以及最右边的一块扇出插件 FANOUT 板。两块时钟插件分别是 CCG（Central Clock Generator）以及 CFB（Clock Fanout Board）。

FEE 板的主要任务是处理反中微子探测器（AD）或者水切伦科夫探测器（WP）中光电倍增管（PMT）输出信号[61]。它最主要的功能包括：

- 测定每个光电倍增管输出的电荷量，以此来确定粒子在液闪中的能量沉积。这可以帮助我们区分中微子事例，分辨噪声本底，由此推出反中微子能谱。
- 提供光每个电倍增管信号到达的精确时间信息，此信息可以用于探测器内部反中微子事例反应顶点的位置重建，同时可以用来研究本底并去掉本底的影响。
- 为触发系统提供快信息（总电荷/能量和 PMT 击中数）。

大亚湾中微子实验的触发系统由 LTB（Local Trigger Board）板、MTB（Master Trigger Board）板以及时钟系统（Clock System）组成。每个 VME 读出机箱内都有一个 LTB 板，为每个探测器系统提供独立的触发判选，并且提供时间标签。而每个站点有一块 MTB 板，为实现同一个站点不同探测器间的交叉触发（cross trigger）提供可能[49]。时钟系统负责为整个实验提供精确的时间信息。时钟/时标广播系统用于向三个实验站传送高精度时钟信号和绝对时标信息，统一的时钟用以保证整个实验的三个实验站的电子学系统以相同的频率进行物理信号的测量，统一的绝对时标信息使得可以在三个实验站的物理事例之间建立时间关联[50]。

图 6-2 所示是 AD 机箱内部的 FEE 与 LTB 插件之间的信号传递情况[51]。其中从 LTB 传给 FEE 的信号有：由双芯 lemo 经过前面板传输的信号，包括了 40MHz 的时钟信号与触发信号，以及通过 VME 背板总线传递的信号，包括了 2bit 的触发类型信号与 1bit 的触发校验（check）信号。而从 FEE 板传向 LTB 板的信号有：通过前面板的电缆传递的击中数信息（Multiplicity）

和通过双芯 lemo 传递的能量和信息（Analog Sum），以及通过 VME 背板总线传递的 BufferFull 信号与 ReadRequest 信号。

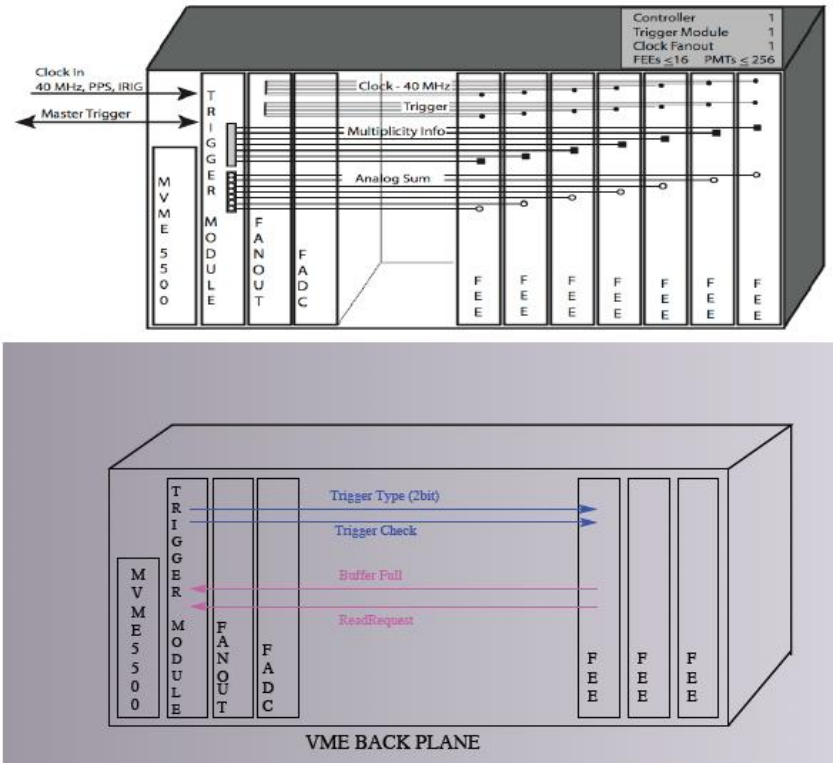


图 6-2 AD 机箱内部 FEE 与 LTB 之间的信号传递

6.1.2 DAQ 系统硬件配置

图 6-3 是单 AD 机箱 DAQ 系统的示意图，这里根据运行的硬件设备的不同粗略的将 DAQ 系统分为了读出系统与在线软件两个部分，读出系统运行在 MVME5500 平台上，而在线软件运行在 PC 机上。读出系统通过 VME 总线与电子学系统连接，同时通过以太网与在线软件进行通信。MVME5500 的软硬件配置在第 3 章中已经有过详细介绍。在线软件运行的控制 PC 的硬件配置为 Intel(R) Xeon(TM) CPU 3.20GHz 的 CPU，4G 的内存以及千兆网卡，控制 PC 安装 SLC（Scientific Linux CERN） release 4.6，内核版本 2.6.9，GCC 版本 3.4.6。

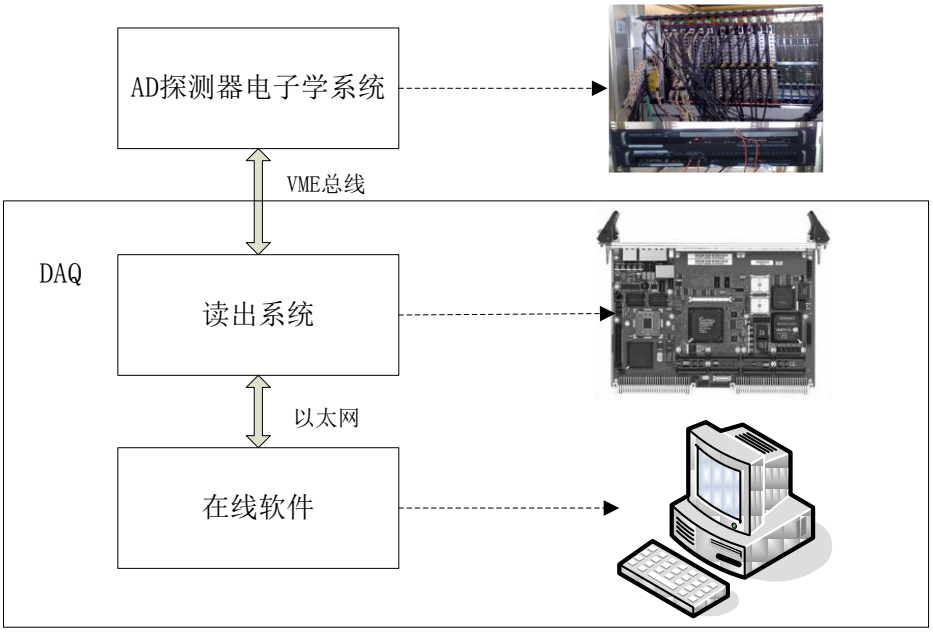


图 6-3 单 AD 机箱的 DAQ 系统示意图

6.2 软件环境

开发大亚湾 DAQ 系统的软件是一个庞大的软件工程，大型的软件项目开发对于软件版本管理和代码安全性的要求都格外高，因此需要考虑选择一种适合 DAQ 系统使用的软件版本管理和代码备份管理工具。CVS（Concurrent Versions System 并行版本系统）[52]是一个开放源码的免费软件，功能非常强大，在全球有很多用户，包括很多国内外著名的高能物理实验（如 Atlas、BESIII）都使用它来管理代码，有关这部分内容的详细情况可以参见[53]。大亚湾实验的 DAQ 软件也通过 CVS 进行管理，图 6-4 所示是大亚湾实验的 DAQ 软件功能随版本升级的情况。

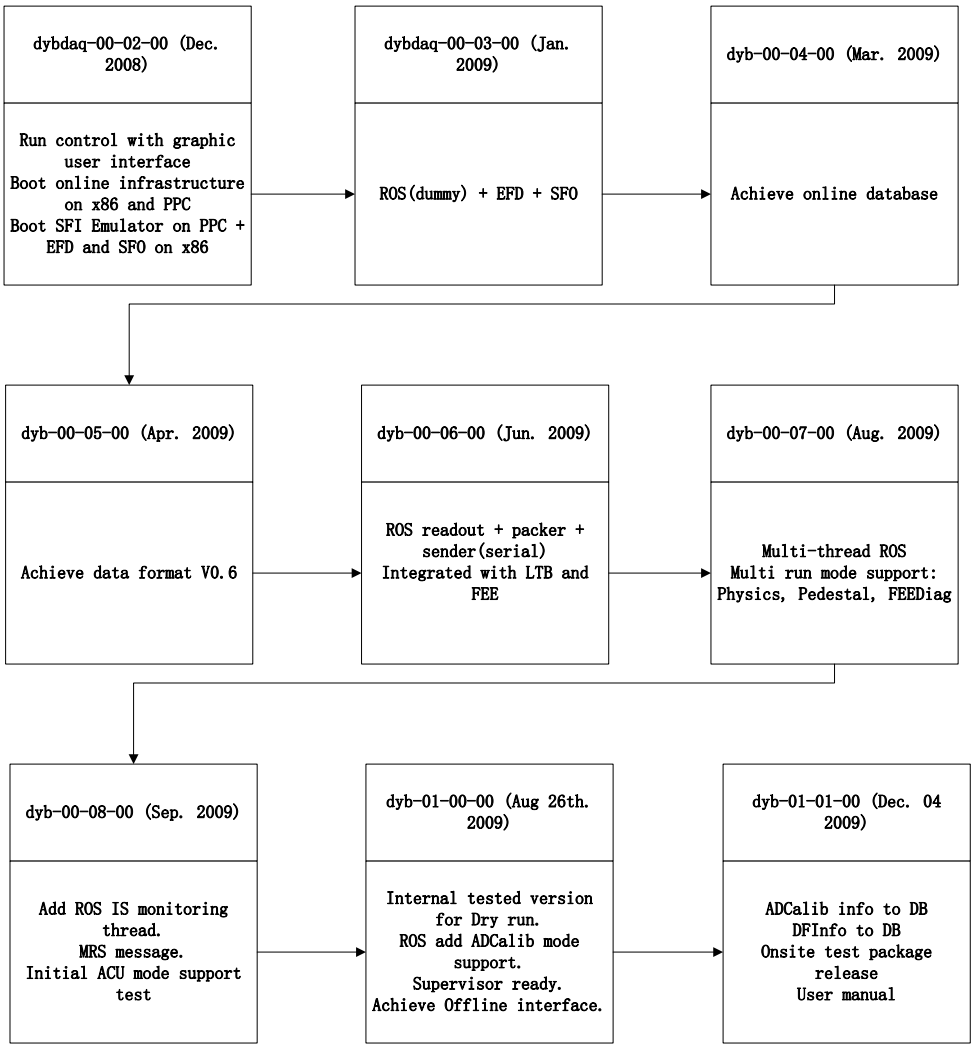


图 6-4 大亚湾 DAQ 软件版本升级情况

在单机箱的 AD 电子学读出系统搭建完成时，大亚湾数据获取系统的软件版本已经升级到 dybdaq-01-01-00，该版本软件功能已经较为完善，基本实现实验设计时的功能需求（除了在线直方图显示功能尚缺），能够完成电子学插件的配置，电子学数据的读出、组装以及存储，数据库功能基本完成，具有初步的报错与数据流监测体系，具备友好的用户界面。同时，该版本的数据获取软件的整体性能情况也满足大亚湾实验的设计指标。

6.3 DAQ 与电子学系统的联调

6.3.1 测试软件包的开发与维护

为了方便与电子学系统的联调，开发了独立于整个 DAQ 软件框架的测试软件包。这是一套简易的数据获取程序，只保留了 DAQ 软件中最基本的配置电子学插件、通过 VME 总线读取电子学数据以及检查、数据存盘等功能。程序中大部分代码重用了 DAQ 框架软件中的读出系统。

测试程序包的代码清晰直观，修改方便，操作简单灵活，因此能够第一时间诊断电子学插件，迅速定位问题来源。测试程序的代码依赖于电子学系统的软硬件需求，因此需要随整个电子学系统一起更新升级。在 DAQ 与电子学系统联调的过程中，测试软件包的存在起到了重要作用。

6.3.2 电子学配置流程

为确保电子学系统有序工作，首先必须明确电子学系统的配置流程，保证电子学各系统间的时序同步。AD 探测器的读出使用的是 PMT 读出电子学，因此本节只介绍与 PMT 读出电子学相适应的配置流程。

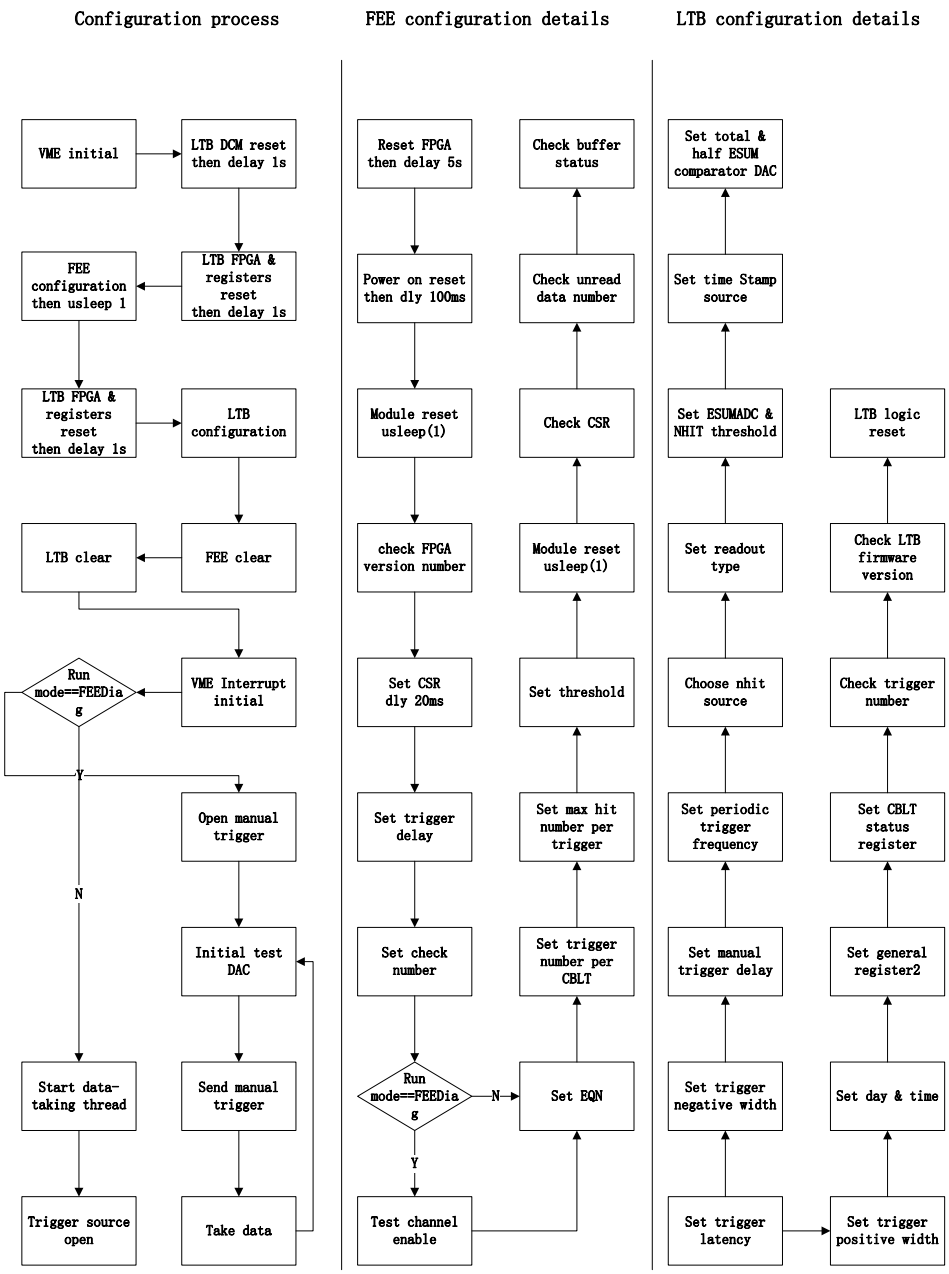


图 6-5 PMT 读出电子学插件配置流程

在 PMT 读出电子学系统的 VME 机箱中，主要有两种需要配置的电子学插件：FEE 和 LTB，它们详细的配置流程如图 6-5 所示[54]。为了方便分析，可以从逻辑上将配置流程简化为如图 6-6 所示，当读出系统收到 Config 命令时，进行 LTB 与 FEE 的复位操作以及所有相关寄存器的配置；在收到 PrepareForRun 命令后，LTB 板与 FEE 板进行 clear 操作，即清空数据缓冲区并且复位触发号；最后在收到 StartTrigger 命令以后，随即打开触发源，

开始进入取数流程。在 2.6.1 节中讲述系统状态转换时曾经提到，如果运行模式不改变，系统可以在 Stop 以后直接 Start，而不需要重新进行配置。电子学的配置也需要适应这一要求，因此在每次 PrepareForRun 时，仅仅对数据缓冲区和触发号进行复位（clear 操作），而保留板上其他寄存器的已有配置，这就节省了再次配置寄存器的时间，可以使系统迅速的投入再次的运行中。

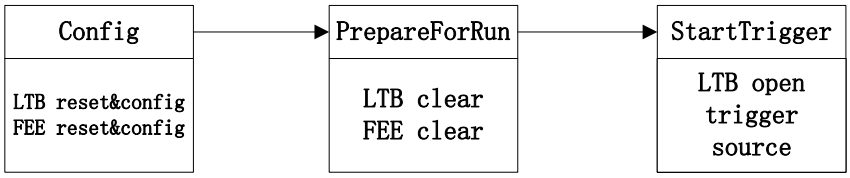


图 6-6 简化的电子学配置流程

6.3.3 电子学系统功能的完善

在系统联调的过程中，DAQ 系统作为电子学系统的用户，需要协助电子学系统进行功能的完善，以更好的满足实验的需求。下面只列举了一些相对比较重要的功能，详细内容请参见参考文献[47]、[55]。

- LTB 板中断功能的实现
- LTB 与 FEE 板 CBLT 读出模式的实现
- FEE 板与 LTB 板对于板上数据缓冲区的保护功能的实现
- LTB 送给 FEE 板的触发号校验信号 check 的添加
- FEE 送给 LTB 板的 EQN 信号（表示数据准备好）的添加
- FEE 板与 LTB 板调试手段的完善，主要体现在板上寄存器功能的完善

6.4 DAQ 系统工作情况

基于 AD 机箱的数据获取系统的取数性能（包括事例率与读出带宽）见表 6-1，表中的数据是根据调整阈值来设定不同读出数据量大小的情况下，DAQ 系统所能够达到的最大事例率与带宽（单事例读出模式）。当 AD 机箱中 192 个 PMT 电子学通道全击中，并且每个通道只有 1 个 hit 的情况

下，DAQ 系统能够达到的最大事例率为 1.7kHz，对应的数据读出带宽为 2.9Mbytes/s，这一性能完全满足大亚湾实验 PMT 电子学机箱的读出性能指标。如果在今后的实验中对读出性能指标提出更高的要求，DAQ 系统会考虑采用 4.5.1 节中介绍的多事例读出的模式来提高系统的读出性能。对于当前单事例读出模式下系统性能情况的具体分析，将在下两节中展开详细的讨论。

表 6-1 大亚湾实验单 AD 机箱整体取数性能

Fired Channel (full crate: 16chn*12FEE)	48	96	192	192	192
Hit Number per Channel	1	1	1	2	5
Event Size (KB)	0.57	0.94	1.7	3.0	8.1
Event Rate (kHz)	2.5	2.0	1.7	1.6	1.1
Readout Bandwidth (Mbytes/s)	1.4	1.9	2.9	4.8	9.2

图 6-7 展示了 DAQ 系统长期稳定性实验的情况，DAQ 系统工作在 1kHz，台阶取噪声的模式下，系统经过二十个小时的运行，工作正常，性能稳定，取数过程中的相关软硬件信息都及时通过数据库进行了存储，如图 6-8 所示。

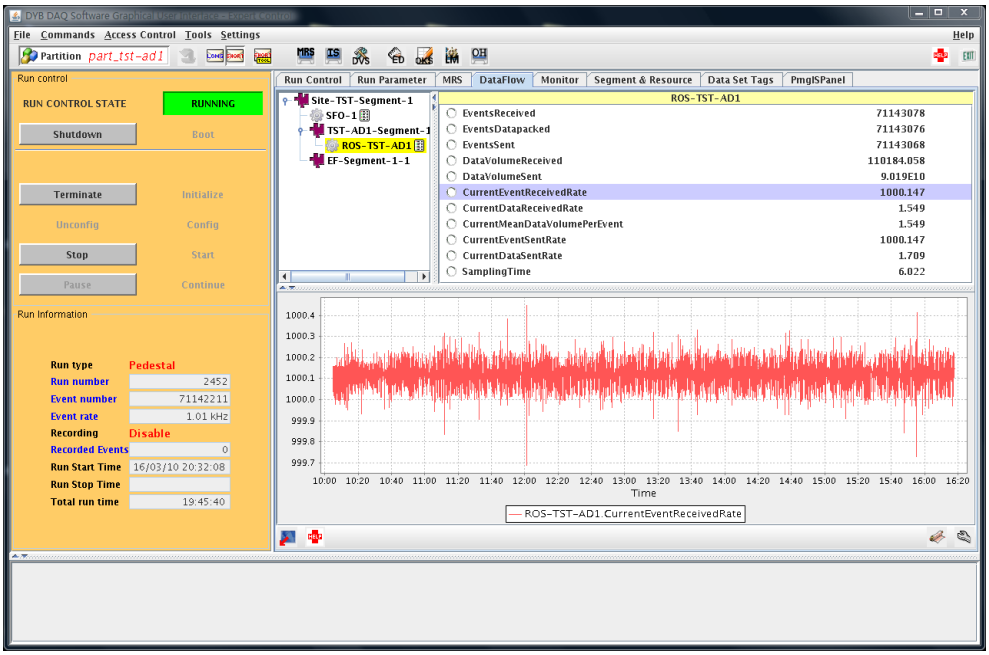
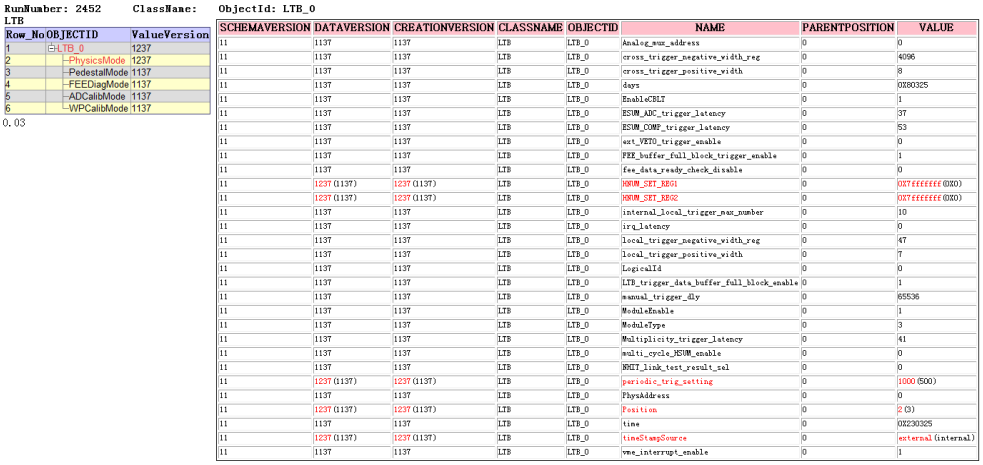


图 6-7 DAQ 长期稳定性实验



RunNumber: 2452 ClassName: ObjectId: LTB_0
LTB

Row_No	OBJECTID	ValueVersion
1	LTB_0	1237
2	PhysicsMode	1237
3	PedestalMode	1137
4	FEEDiagMode	1137
5	ADCalibMode	1137
6	WFCalibMode	1137

0.03

SCHEMAVERSION	DATAVERSION	CREATIONVERSION	CLASSNAME	OBJECTID	NAME	PARENTPOSITION	VALUE
11	1137	1137	LTB	LTB_0	Analog_mux_address	0	0
11	1137	1137	LTB	LTB_0	cross_trigger_negative_width_reg	0	4096
11	1137	1137	LTB	LTB_0	cross_trigger_positive_width	0	8
11	1137	1137	LTB	LTB_0	days	0	0X80325
11	1137	1137	LTB	LTB_0	EnableCBLT	0	1
11	1137	1137	LTB	LTB_0	ESUM_ADC_trigger_latency	0	37
11	1137	1137	LTB	LTB_0	ESUM_COMP_trigger_latency	0	53
11	1137	1137	LTB	LTB_0	ext_VETO_trigger_enable	0	0
11	1137	1137	LTB	LTB_0	FEE_buffer_full_block_trigger_enable	0	1
11	1137	1137	LTB	LTB_0	fee_data_ready_check_disable	0	0
11	1237 (1137)	1237 (1137)	LTB	LTB_0	KSUM_SET_REG1	0	0X7fffffff (0X0)
11	1237 (1137)	1237 (1137)	LTB	LTB_0	KSUM_SET_REG2	0	0X7fffffff (0X0)
11	1137	1137	LTB	LTB_0	internal_local_trigger_max_number	0	10
11	1137	1137	LTB	LTB_0	irq_latency	0	0
11	1137	1137	LTB	LTB_0	local_trigger_negative_width_reg	0	47
11	1137	1137	LTB	LTB_0	local_trigger_positive_width	0	7
11	1137	1137	LTB	LTB_0	LogicalId	0	0
11	1137	1137	LTB	LTB_0	LTB_trigger_data_buffer_full_block_enable	0	1
11	1137	1137	LTB	LTB_0	manual_trigger_dly	0	65536
11	1137	1137	LTB	LTB_0	ModuleEnable	0	1
11	1137	1137	LTB	LTB_0	ModuleType	0	3
11	1137	1137	LTB	LTB_0	Multiplicity_trigger_latency	0	41
11	1137	1137	LTB	LTB_0	multi_cycle_MSUM_enable	0	0
11	1137	1137	LTB	LTB_0	NHIT_link_test_result_sel	0	0
11	1237 (1137)	1237 (1137)	LTB	LTB_0	periodic_trig_setting	0	1000 (500)
11	1137	1137	LTB	LTB_0	PhysAddress	0	0
11	1237 (1137)	1237 (1137)	LTB	LTB_0	Position	0	2 (3)
11	1137	1137	LTB	LTB_0	time	0	0X230325
11	1237 (1137)	1237 (1137)	LTB	LTB_0	timeStampSource	0	external (internal)
11	1137	1137	LTB	LTB_0	vme_interrupt_enable	0	1

图 6-8 数据库查询功能示意

6.5 结合 AD 机箱的数据获取性能分析

整个 DAQ 系统的数据流可以简化为图 6-9 所示，要分析 DAQ 的性能也必须从数据流的走向着手。为了方便的讨论这个问题，将图 6-9 中参与数据传递的几个重要元素切割成两个组合，上半部分是数据从 VME 机箱到 ROS 读出系统，性能主要取决于读出系统的数据通过能力；而下半部分数

据流从 EFD 接受网络传来的事例数据开始，一直到存盘结束，性能主要取决于 EFD 的数据通过能力以及 SFO 的数据存盘的能力。本节将根据这样的切分来具体的研究 AD 机箱中的 DAQ 系统性能。

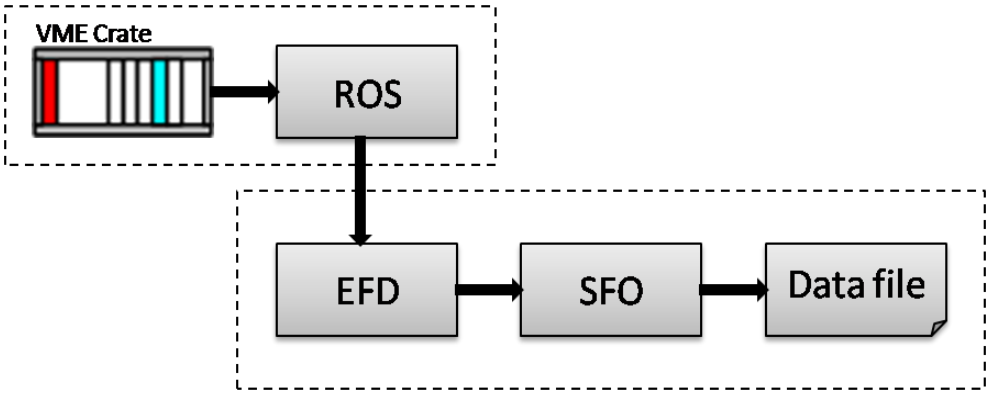


图 6-9 DAQ 数据流简化示意图

6.5.1 读出系统的数据通过能力

首先研究读出系统的数据通过能力。在第 4 章中对基于 Linux 操作系统的 DAQ 读出系统的整体性能作出了仔细的分析，在本节中将会结合大亚湾 AD 机箱运行时的硬件与软件的实际配置情况，给出具体的性能参数。读出系统的数据通过能力取决于 CBLT 读出任务、数据组装任务、网络传输任务这三个主要任务的性能。

6.5.1.1 AD 机箱 CBLT 读出能力

为了单独测试读出系统中 CBLT 读出任务的能力，关闭数据组装任务与网络传输任务，通过监测任务给出的参数观察 CBLT 读出性能，如图 6-10 所示。

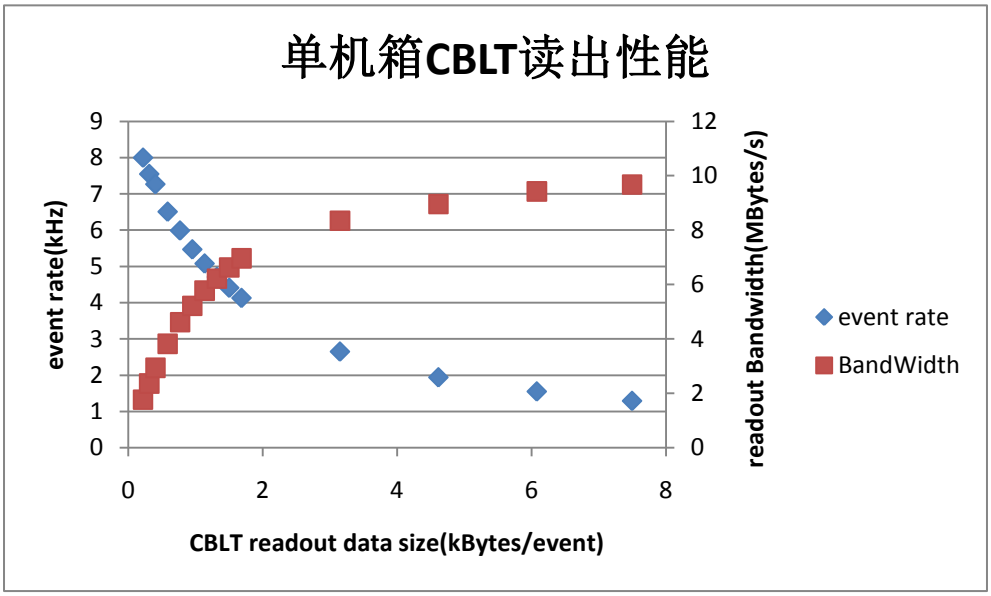


图 6-10 单机箱 CBLT 读出性能情况

表 6-2 单机箱 CBLT 读出性能

Over threshold channel	Event Size(KB)	Event Rate(KHz)	Bandwidth(MB/s)
0	0.221	8	1.768
12	0.313	7.55	2.363
24	0.404	7.27	2.937
48	0.587	6.51	3.821
72	0.771	5.99	4.618
96	0.954	5.47	5.218
120	1.14	5.08	5.776
144	1.32	4.71	6.217
168	1.50	4.41	6.628
1hit*192	1.69	4.13	6.963
2hit*192	3.15	2.65	8.350
3hit*192	4.62	1.94	8.955
4hit*192	6.08	1.55	9.424
5hit*192	7.50	1.29	9.672

通过上述性能列表可以看出 AD 机箱在全通道（192PMT）击中并且每个通道只有 1 个 hit 的情况下，单机箱读出数据长度约为 1.6Kbytes，此时读出事例率约为 4kHz，数据传输带宽约为 7Mbytes/s。这一性能完全满足系统的设计要求，但是与 4.2.3 节中测到的 DMA 的传输性能有较大差距，这是因为 VME 的传输速率并不是一个固定的指标。通常说来，32 位 VME 总线的数据传输的极限带宽是 40Mbytes/s，然而这只是一个理论上的数据，并且只是瞬时传输可能达到的最高速率而并非能够持续达到的平均传输速率[56]。事实上 VME 总线的传输速率取决于主从设备的具体硬件设计指标，并且和软件的机制也是密不可分的。从硬件设计上来说，AD 机箱所能达到的最大传输带宽为 12Mbytes/s[57]。从软件机制上来说，读出系统的数据处理是受事件驱动的，电子学插件在准备好事例数据时发出中断请求，读出系统响应中断才会开始 CBLT 传输，因此这里测到的实际 CBLT 传输速率必然还要因为与中断相关的处理而比理想值有所下降。综上所述便解释了为何 AD 机箱中的实际 CBLT 传输速率要低于 VME 总线传输的理论指标。

6.5.1.2 事例组装的运算能力

事例组装任务主要负责数据的检查与打包，在此过程中涉及大量的位操作、比较运算以及内存拷贝的操作，组装的性能取决于多方面的因素，包括采用的算法、约定的数据结构、数据包大小以及处理机的运算速度等，此外组装任务还需要处理数据中可能出现的异常，因此，给出准确的读出系统的事例组装的运算能力是比较困难的。通过表 6-3 的实验数据可以对事例组装任务对读出性能的影响有一个大致的概念。表中第一列是电子学数据包的大小，第二列是关闭数据组装任务与网络传输任务后，只进行 CBLT 任务所观察到的事例率情况，而表中第三列是关闭网络传输任务，打开 CBLT 任务与事例组装时，通过监测任务所观察到的事例率情况。从表中二三列的数据比对可以看出，增加数据组装任务以后，对于读出性能的影响是很微小的，读出事例率相对于增加组装任务之前只有 2%~5%的下降。这主要是由于读出系统被设计为多任务并行的系统，在 CBLT 传输过程中，CPU 可以空闲下来去处理事例组装运算，因此有效的利用了 CPU，节省了

事例处理的时间，相对于普通的串行式设计能够大大提高系统的工作性能。

表 6-3 事例组装任务对读出性能的影响

Event Size(byte)	Event Rate(kHz) CBLT task only	Event Rate(kHz) CBLT + Datapack task
206	11.00	10.56
694	6.51	6.40
1304	4.41	4.29
4234	1.66	1.60

6.5.1.3 读出系统的网络传输机制

读出系统中组装好的事例会通过网络发送给后端 DAQ 组件进行其他处理，例如直方图显示、数据文件存盘等等。如果事例数据不能及时通过网络发送走，那么读出系统的各级缓冲区便会由后向前逐个堵塞，一旦堵塞电子学插件中的数据缓冲区，便会造成电子学数据无法及时读出，最终导致系统出现死时间。4.2.6 节中已经给出了基于 MVME5500 的 Linux 操作系统下的网络传输性能参数，但是因为测试采用了最为简单的测量机制，因此得到的是一个理想的参数，可以说是同样配置下的极限性能参数。而结合读出系统采用的网络传输机制，实际的数据传输能力将在下一节得到进一步的分析。

6.5.2 DAQ 后端组件的数据通过能力

所谓的后端组件，这里主要指图 6-9 中的 EFD 以及 SFO 这两个参与数据流传递的组件。为了不受到前端电子学系统的影响，脱离读出系统而单独测试 EFD 与 SFO 的性能，在 DAQ 的软件框架中设计了 dummyROS（模拟 ROS）的组件。dummyROS 组件运行在 MVME5500 的 CPU 上，它的功能仅仅是自动生成一段数据来模拟 ROS 组装完成后的事例，并且将这段模拟数据通过网络发送给 EFD，最终通过 SFO 进行存盘。利用 dummyROS 取数的 DAQ 性能结果如图 6-11 所示，当模拟数据量的大小约为单 AD 机箱全通道击中的数据量（1.6Kbytes）时，此时后端组件的整体数据通过的

事例率约为 2KHz，数据带宽约为 3Mbytes/s。

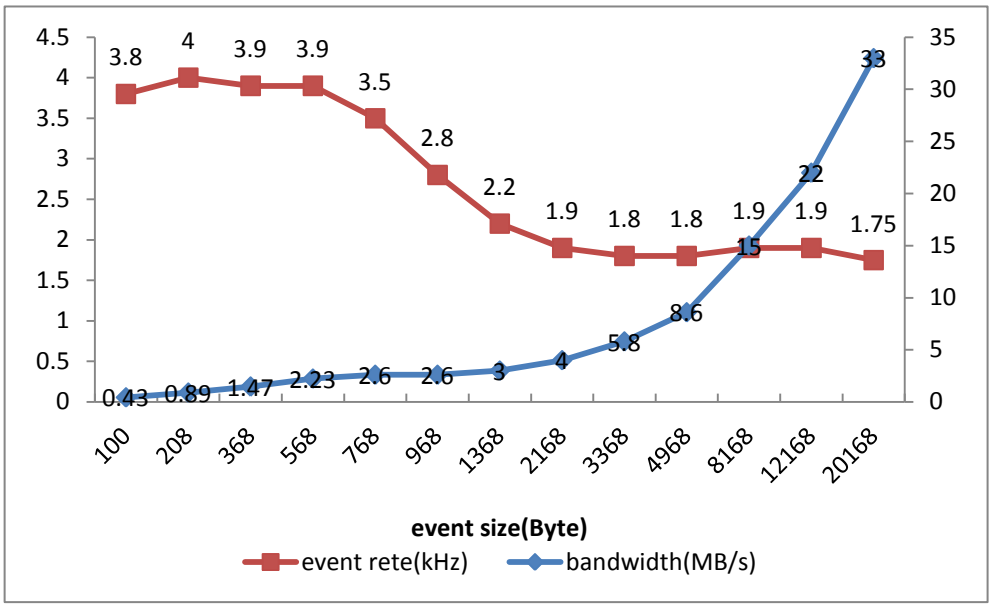


图 6-11 使用 dummy ROS 的 DAQ 读出性能

从 2.4.3 节中对 EFD 的简介可以了解到，ROS（或 dummyROS）将事例数据通过网络发送给 EFD，数据在从 ROS 向 EFD 传输的过程中使用了握手的机制。当 EFD 准备好接收数据时，会向 ROS 发出事例请求（Event Request），ROS 在准备好一个事例时便会响应这一请求，将事例数据发送给 EFD。EFD 成功接收事例后，会给出确认应答（Acknowledgement），并且准备好启动下一次的事例请求。ROS 响应 EFD 的事例请求是没有时间限制的，因为并不能确定下一个事例的到达时间，因此可以说，事例的传递是完全的数据驱动的。ROS 会将已经发送出去的事例保留在内存中直到接收到来自 EFD 的确认应答。而如果在 ROS 接收到确认应答之前发生了网络连接失败的情况，ROS 会将最后一次发送的事例重新放回队列中去，当新的连接建立，下一次的事例请求到来后将这个事例再次发送。类似的，EFD 与 SFO 之间的传输也有相应的应答机制。

这样的网络传输机制能够确保在事例的传输过程中，不会发生由于网络连接失败而造成数据丢失的情况，但随之带来的负面影响是数据传输的性能必然会因此降低，因此在目前的 DAQ 系统中，EFD 的传输机制是制约系统性能的主要因素。

6.6 系统稳定性分析

根据上一节对 DAQ 工作性能的分析可以了解到，在目前大亚湾 DAQ 系统的设计框架下，系统的平均读出性能（包括事例率以及数据带宽）可以满足实验的设计指标。而通过之前几章的介绍已经知道，DAQ 系统的内部存在着复杂的结构，有密集的上下文切换、有多级的数据传递，包括任务间的数据传递以及通过网络的数据发送等等，类似的一系列因素最终导致了系统性能必然会存在一定程度的抖动性。因此在 DAQ 平均性能满足指标的前提下，仍然需要通过实验检验是否会存在因为 DAQ 系统的性能抖动所造成的读出死时间。

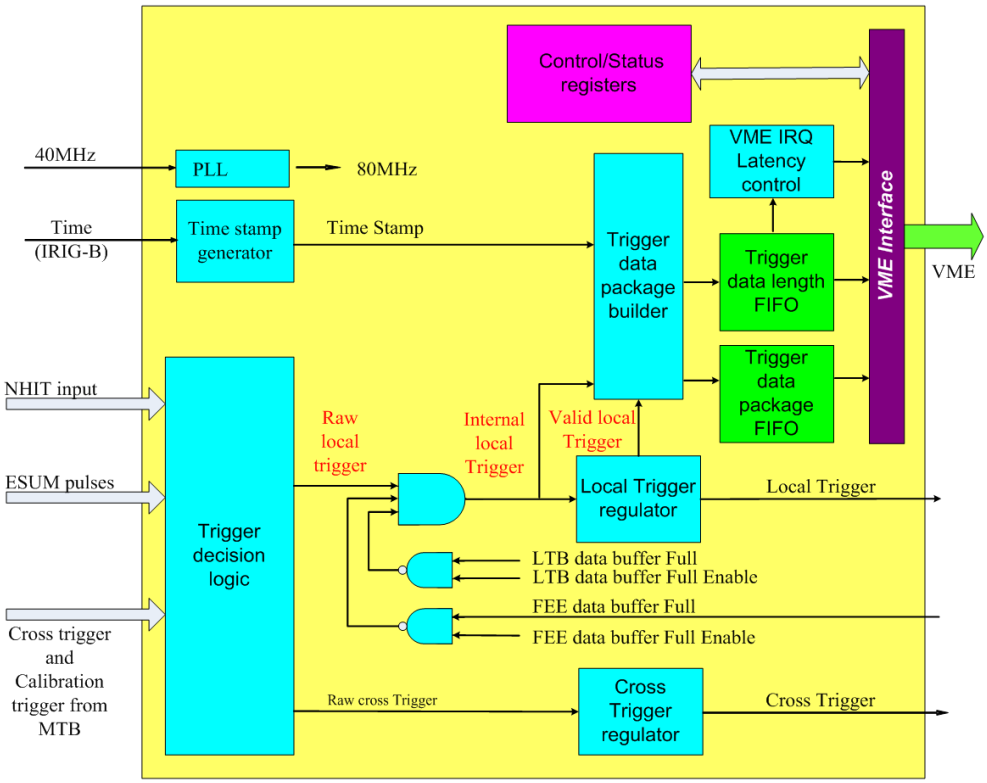


图 6-12 LTB 逻辑框图

LTB 板的触发判选逻辑如图 6-12 所示，从图中可以看到，当各触发模块给出 trigger 时，这个 trigger 被称作 raw local trigger，而此时如果电子学插件（包括 FEE 与 LTB）的缓冲区都尚未满，则 raw local trigger 会形成一个 internal local trigger，再经过与触发板预先设置的触发类型的符合后，满

足要求的 internal local trigger 才会成为最终的 local trigger 发出。

如果 DAQ 系统不能够对数据进行及时的读出处理，那么 LTB 或者 FEE 插件的数据缓冲区就会被堵塞，此时触发模块虽然能够继续产生 raw local trigger，但是无法形成 internal local trigger，即这个 raw local trigger 只能被丢弃。换句话说，一旦电子学插件的缓冲区出现了堵塞的现象，LTB 板内部所记录的 raw local trigger 的数目便会大于 internal local trigger 的数目。根据以上分析，可以用实验数据检验 DAQ 系统的工作情况。

大亚湾实验要求数据获取系统需要设计到能够承受最大触发率为 1kHz。因此这里采用 LTB 板的周期性触发模式，触发频率设置为 1kHz，并且将 FEE 的阈值设为 0，使得 FEE 板可以达到全通道读出的工作条件。在上述取数条件的设置下，DAQ 系统工作超过 12 小时，取到超过四千万个事例，平均带宽为约 1.8Mbytes/s。在停止触发后，检查触发板的寄存器 Register17：Raw Local trigger number register 以及 Register18：Internal Local trigger number register[47]，发现 raw local trigger 的数目和 internal local trigger 的数目是完全相等的，说明在取数的过程中，电子学插件没有出现过堵塞的情况。通过这个实例便验证了当触发率稳定在 1kHz，并且数据读出的带宽也足够满足设计要求的条件下，DAQ 系统读出性能稳定，没有给系统造成任何的死时间。

6.7 数据文件的回读分析

1.3.2 节中在介绍高能物理实验中的 DAQ 系统的任务时，指出了 DAQ 系统应当具备存盘文件的回读与处理的功能。通过对存盘的数据文件的读取与分析，既可以通过从数据中提取相关的性能指标检验电子学系统的工作状态，同时也可以逆向证明 DAQ 系统工作的正确性与稳定性。

6.7.1 检验 DAQ 系统的工作情况

通过对存盘数据文件的回读分析，至少可以从以下几点来检验数据获取系统的工作逻辑：

- 通过对数据文件中电子学通道工作状态的分析可以检验DAQ系统对电子学插件的配置以及相关寄存器的设置是否正确
- 通过对数据文件格式的校验可以确认DAQ的事例组装的正确性以及数据文件的格式正确性
- 通过对数据文件中事例时间信息的分析可以校验DAQ系统工作的稳定性，是否在取数过程中引入死时间等（仅限于固定触发频率的取数模式）

6.7.2 PMT 电子学通道性能检验

检验电子学各通道的工作情况以及性能表现，在读出机箱未连接探测器的条件下，有两种主要的取数模式作为检验手段：即电子学诊断模式与台阶模式。电子学诊断模式是让电子学插件自己产生标准输入信号，通过分析输出信号与输入信号的关系来检验电子学通道是否正常工作；台阶模式用来观测没有信号输入的情况下，电子学系统自身的基线与噪声水平。目前在线直方图功能尚缺，因此当前的做法是对存盘的数据文件进行回读，根据要求对数据进行相关的处理后，通过 ROOT[58]软件填注直方图，给出电子学通道性能的检验结果，并且及时将其中的重要信息提供给电子学人员进行分析与处理，以确保电子学插件始终工作在正常范围内。

6.7.2.1 电子学诊断模式

对于电子学诊断模式，就是按照平均分布原则选择一系列的标准输入信号值，经过各电子学插件的处理后得到相应的输出信号值。为了使测量准确，需要对每个标准信号的测量重复上千次再取平均得到一组输入输出对。针对这些输入输出点对，用线性拟合来得到两组信号值函数关系曲线。通过观察拟合直线的斜率与截距，可以检验电子学各通道是否正常工作，是否存在死道或者冒道等异常通道。

电子学诊断模式中用到的算法主要分两类：一类是数据筛选算法，还有一类是数据拟合算法。如图 6-13 所示为 FEE 某一个通道的输入输出对在坐标轴上的位置，两条拟合直线对应与高低两个量程的数据。另外，对

每个标准信号的测量重复上千次得到的数据会受到各种因素的影响，所以在必要时需要考虑使用一定的规则来对数据进行过滤筛选，例如使用 3σ 规则或去掉前后 10%的数据等[59]。

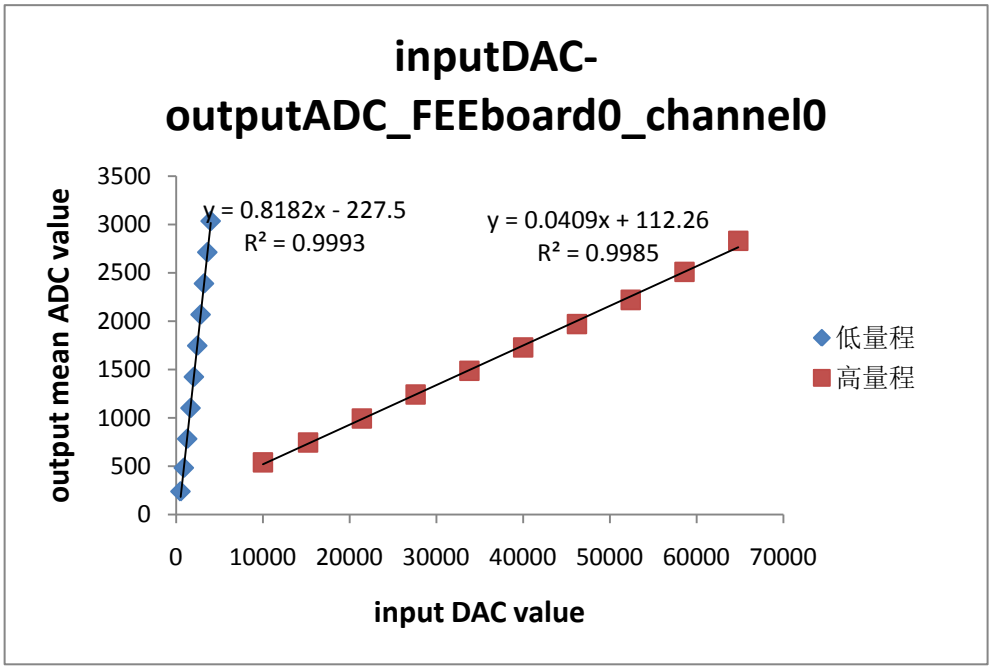


图 6-13 FEE 单通道输入 DAC 与输出 ADC 线性关系示意

原先的 n 个点对通过数据筛选算法后，就得到了 m（$m \le n$）个点对 $\left(\overline{Value_i}, Point_i\right)$，通过解如下多项式方程：

$$\begin{bmatrix} m & \sum_{i=1}^{m}\overline{Value_i} \\ \sum_{i=1}^{m}\overline{Value_i} & \sum_{i=1}^{m}\overline{Value_i}^2 \end{bmatrix}\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{m} Point_i \\ \sum_{i=1}^{m} Point_i\,\overline{Value_i} \end{bmatrix}$$

简化为：

$$\begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix}\begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \end{bmatrix}$$

即可得到输入信号和输出信号之间线性关系的参数：

$$a_0 = \frac{\begin{vmatrix} s_1 & r_{12} \\ s_2 & r_{22} \end{vmatrix}}{\begin{vmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{vmatrix}}, a_1 = \frac{\begin{vmatrix} r_{11} & s_1 \\ r_{21} & s_2 \end{vmatrix}}{\begin{vmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{vmatrix}}, rms = \sqrt{\sum_{i=1}^{m}\overline{value_i}^2 - 2a_1s_2 - 2a_0s_1 + a_1^2.r_{22} + 2a_0a_1r_{12} + a_0^2.r_{11}}$$

利用以上的公式，可以算出拟合直线的斜率 a0、截距 a1 以及标准偏差值 rms。

根据电子学组给出的通道判选标准 -400<QL_A0<400，0.5031<QL_A1<0.9431，0<QH_A0<200，0.03294<QH_A1<0.04942[60]，可以判定是否存在死道冒道等异常电子学通道。图 6-14 所示是单机箱 12 块 FEE 插件的电子学诊断结果图，根据上述判选标准，机箱内 12 块 FEE 插件的 192 个电子学通道均工作在正常区域内。作为比较，图 6-15 是机箱中所有通道的高量程的放大倍数，从图中可以明显看到存在一个死道。通过电子学诊断的工作模式以及对数据及时的分析处理，可以帮助第一时间发现电子学通道的异常。

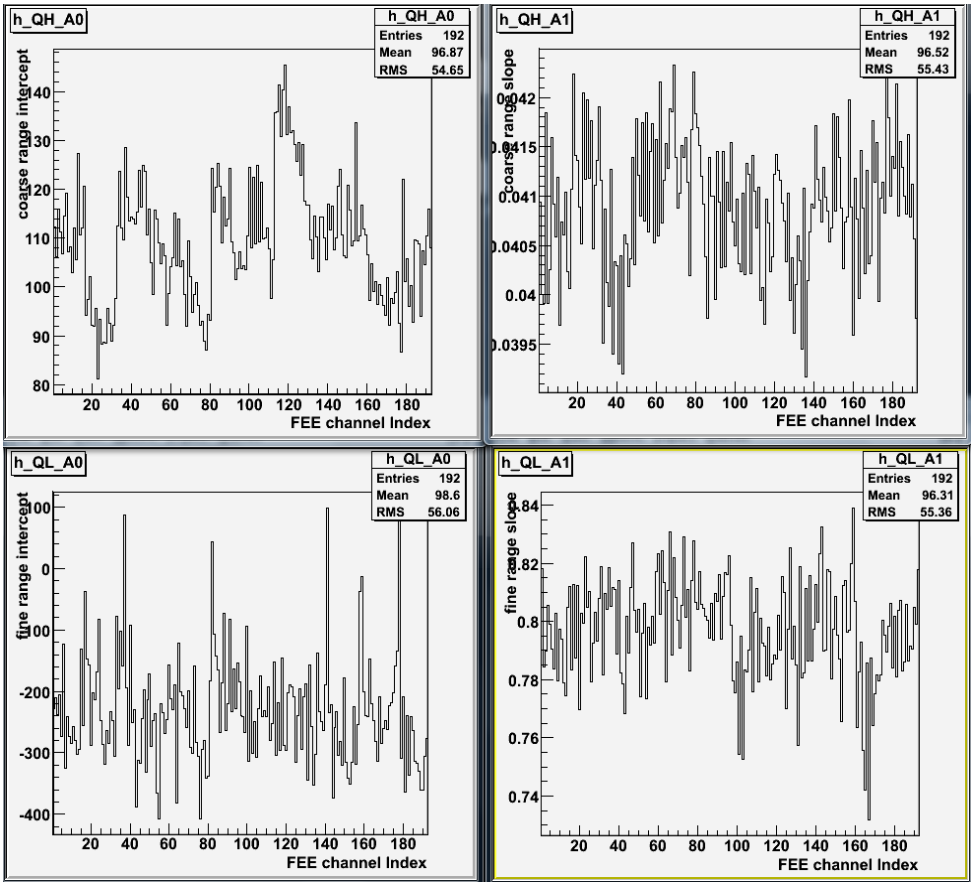


图 6-14 FEE 电子学诊断结果图

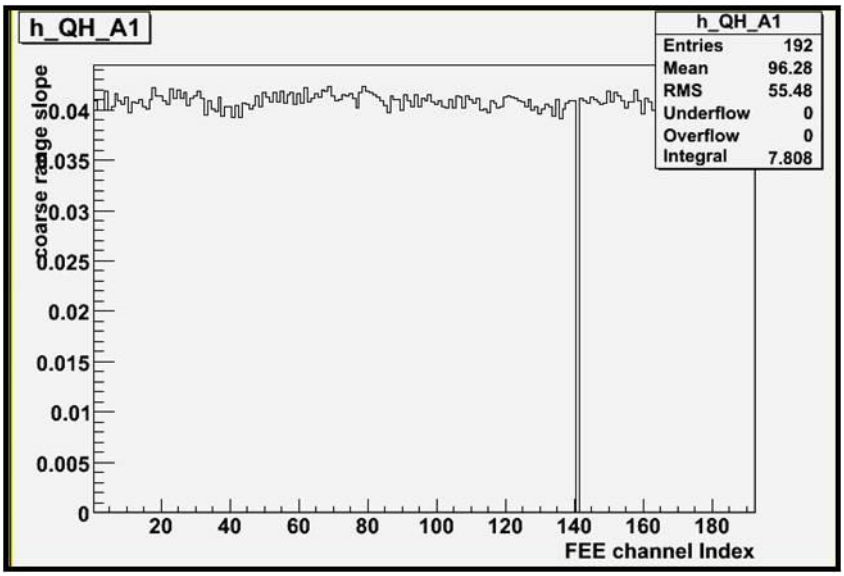


图 6-15 FEE 电子学诊断结果（有死道的情况）

6.7.2.2 台阶模式

台阶模式是在阈值为 0 的情况下，采用随机触发的方式读取电子学数据以观测各通道的基线值以及噪声水平。在电子学通道没有连接探测器的情况下，噪声主要来源于电子学插件自身。台阶模式采用周期性触发，采到一定数量的数据后，可以统计出每个通道采到电荷的平均值 ADCMean 与标准偏差值 ADCRMS，对应每个电子学通道的基线值以及噪声水平。

图 6-16 所示是某个单通道的基线以及噪声分布情况，该通道 ADC 基线值为 79.63，噪声 RMS 为 1.55 道 ADC。利用示波器测量输入信号与输出 ADC 值的关系，可以大概估算出 FEE 输出电荷信号 40 道 ADC 值对应电荷量约为 1p.e.（假定 PMT 增益为 $2*10^7$），PMT 电子学设计要求电子学噪声必须小于 0.1p.e.[61]，即 4ADC bin，因此该通道的噪声水平是符合设计要求的。

图 6-17 所示是机箱内 12 块 FEE 插件的所有 192 个通道的基线与噪声水平的情况。可以看出，每块插件的 16 个通道的基线水平呈现阶梯下降的趋势，这是因为在 VME 机箱中，每块插件内部从下到上温度升高（风扇在最下部），当温度升高时，电子学通道的噪声也随之有升高的趋势。

台阶模式中得到的基线值是今后进行数据分析的重要依据，例如实验

中通过单光电子谱得到的峰值必须扣去基线值才是对应真实的单光电子产生的电量。电子学通道的噪声水平也是实验中的重要指标，决定了如何选取合适的阈值从而尽可能的压低噪声、保留好事例。

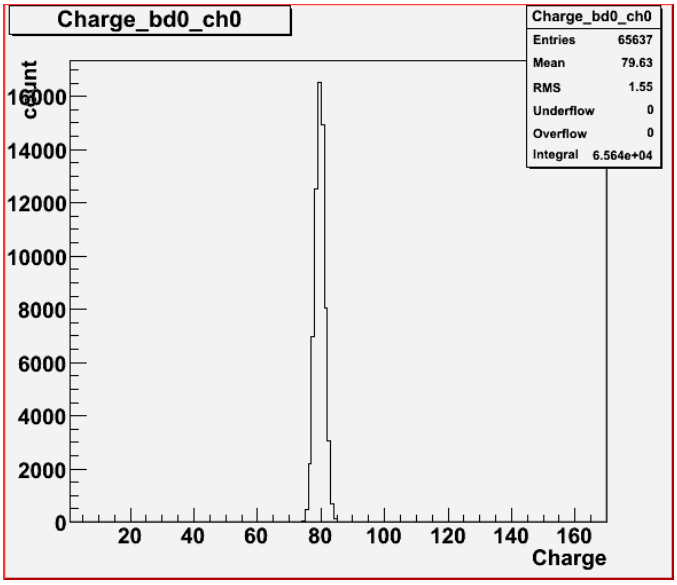


图 6-16 FEE 单通道噪声谱

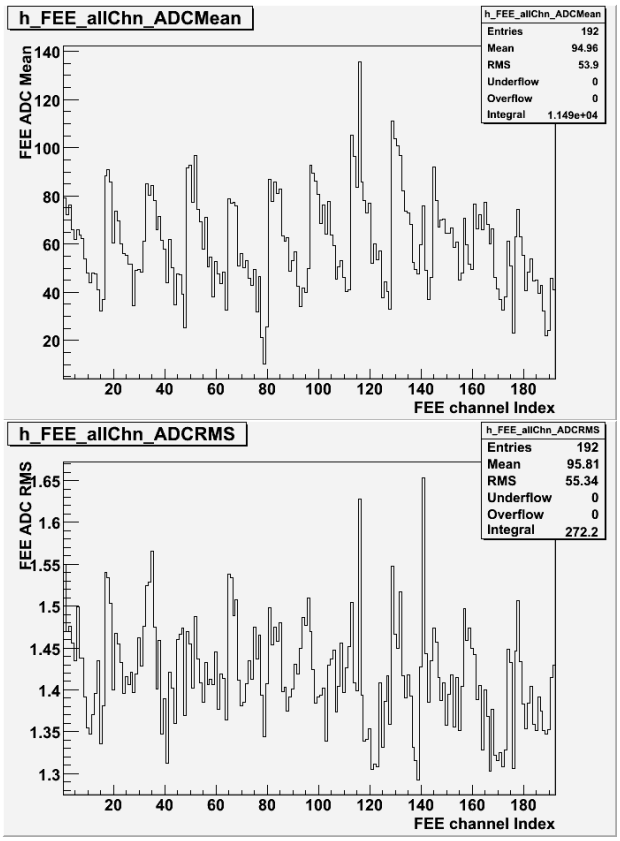


图 6-17 台阶模式得到电子学噪声水平

6.8 本章小结

本章利用大亚湾实验中的反中微子探测器的读出 VME 机箱搭建了完整的大亚湾数据获取系统读出平台，通过实例检验了大亚湾数据获取系统的整体工作状态以及性能和稳定性的情况。并且通过对数据文件的回读与分析，检验了电子学通道的工作情况与性能指标，也验证了 DAQ 系统从取数、事例组装到文件存盘这一完整工作流程的正确性与稳定性。

7 多机箱数据获取系统的设计

上一章节中以单 AD 机箱为平台，介绍了数据获取系统的工作情况，本章将介绍为适应大亚湾实验多探测器的设计，DAQ 系统所采取的设计方案。

7.1 RPC 电子学的 DAQ 系统

大亚湾实验的前端读出电子学系统分为两类，即 PMT 电子学系统与 RPC 电子学系统。数据获取系统与电子学系统有着非常密切的联系，保证两者之间接口逻辑的正确性，对整个实验都有着重要的影响。在本章之前所涉及到的与电子学系统相关的讨论，都是针对 PMT 电子学读出系统的。而 RPC 读出电子学的设计与 PMT 电子学是完全不同的，因此针对 RPC 电子学的读出逻辑[62]（如图 7-1），数据获取的读出系统也需要做出与之相适应的设计。

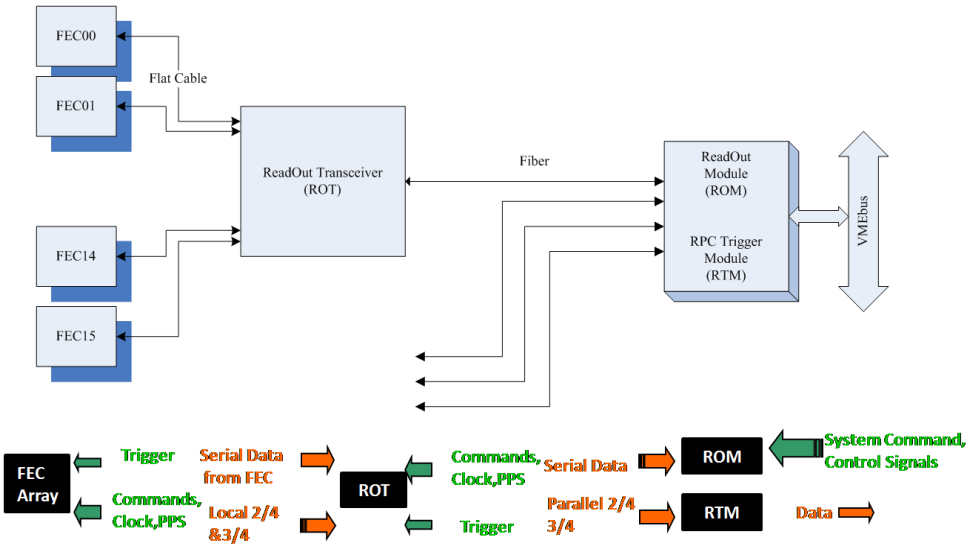


图 7-1 RPC 电子学读出逻辑图

RPC 电子学的读出插件 Readout Module（ROM）通过 VME 总线向数

据获取系统提供来自探测器的电子学数据。每个站点有一个负责 RPC 读出的 VME 机箱，一个机箱中有且仅有一块 ROM 插件，这块 ROM 插件通过光纤连接了 6 块 ReadOut Transceiver（ROT），而每块 ROT 通过电缆最多可以连接 16 块 Front-End Card (FEC)，每块 FEC 有 32 个读出通道。另外，每个 VME 机箱中还有一块负责触发但不参与取数的 RPC Trigger Module（RTM）。

对于数据获取系统来说，RPC 电子学插件的读出与 PMT 电子学最大的区别在于，RPC 电子学数据中没有触发号的信息，读出数据的时序只能通过数据里的时间标签来获得。并且每次 DAQ 系统通过 VME 总线读取到的数据包中，包含了来源于不同 trigger 的的 FEC 的数据。因此 DAQ 系统需要根据读出数据块中的每个 FEC 数据中的时间标签，来寻找与之时间匹配的 FEC 数据。

目前 DAQ 系统针对 RPC 读出的设计考虑了两种排序的方案。第一种方案为：在一次或者多次的读出数据包中找到时间标签完全一致的 FEC 数据，将其组装为一个完整事例，这种方案的优点是数据文件中直接以事例格式存储，但是因为 RPC 电子学读出系统每个事例本身的数据量可能比较小，相对的就会因为事例组装时添加的事例头部信息而占用过多存储空间。第二种方案是，DAQ 系统在指定的时间范围内将乱序的读出数据做一个初步排序，举例说就是，如果以某块 FEC 的某个时间标签 T 为基准，那么 DAQ 系统将把时间标签在（T，T+Δ T）范围内的所有 FEC 数据组装到一起。与第一种方案相比，这种方案的优点就是节省了存储空间，缺点也很明显，因为文件中的数据只是经过了初步的排序，离线人员在进行数据分析时还需要对数据进行进一步的处理才能得到每个事例的信息。

DAQ 系统对 RPC 读出数据的排序工作计划是在读出系统内完成，但是因为读出系统工作在单板计算机上，CPU 能力有限，如果触发率太高可能会因为占用过多 CPU 而影响整体取数性能。因此本节中所提出的方案还要在今后的实际应用中进行检验，才能确定最终的读出模式。

7.2 多探测器取数的同步控制

第 2 章中介绍大亚湾实验的数据获取系统的功能需求时，对于系统的运行控制有这样的要求：可以允许对每个探测器的独立操作，也可以允许对任意探测器组集的统一控制。因此对于多探测器的数据获取系统，对于同步控制就提出了更高的要求。从配置的时序关系，到各探测器的取数状态的控制，都需要做出统一的协调。

从电子学配置上考虑，在多探测器取数的模式下，会加入时钟系统和 MTB 插件等与其他各独立的探测器系统协调工作，各电子学系统间的时序配合可能会比目前提出更复杂的要求，因此数据获取系统现有的状态机的设计也可能会需要进一步的扩展。

从系统的取数状态考虑，在 2.6.2 节中介绍了数据获取系统中 Partition 的概念，如果有一组或者多组探测器的组集需要独立取数，那么就需要为每组独立取数的探测器以及电子学系统建立对应的 Partition，每个 Partition 内部的所有探测器系统通过 RC 组件的统一控制管理以确保各系统间的同步运转。同时，针对非物理取数模式（physics run mode 以外的其他模式），数据获取系统还开发了 supervisor 组件来进行系统的同步控制，supervisor 基于 RC 组件与 IS 组件开发，能够利用与 IS 组件交换运行参数以及相关信息捕捉系统当前的运行状态，并且在满足某种预设的条件时，通过 RC 组件向其他各组件分发状态控制命令。每个 partition 中都包含了独立的控制组件以实现一个 partition 内部所有探测器电子学与 DAQ 系统的同步控制。

7.3 多探测器取数模式下的读出系统

每个探测器的 VME 机箱都对应了一套完整的读出系统，读出系统的设计应当是满足所有探测器的需求的。而每个探测器的电子学系统会有一份独立的电子学配置数据库，因此每个 VME 机箱的数据获取读出系统会从当前探测器对应的电子学配置数据库中读取相应的配置参数进行配置。

在多探测器模式下的读出系统的设计中，需要考虑到取数模式与探测

器的具体需求，例如在 AD 系统进行刻度的同时，希望水池与 RPC 系统同步进行物理取数。这就要求读出系统不仅要接受来自 IS 组件的运行模式的信息，还要能根据运行模式与对应的探测器类型做出判断，以满足实验的要求。

7.4 数据流合并

每个探测器都有一个读出 VME 机箱，每个机箱的数据可以称作是一个独立的数据流。而对于多探测器的数据获取系统，不可避免的需要考虑到数据流合并的问题。

7.4.1 数据流合并的内容

首先需要考虑的是合并哪些数据流，目前的考虑是来自同一个站点的所有探测器的数据流将被合并在一起，在以后实际的取数过程中，根据实验的具体需要，这个条件也可能会有所变化，因此 DAQ 系统的设计方案是将所有数据流是否参与合并设置为一个可以灵活配置的参数，这样便可根据实验的需求对需要合并的数据流随时做出调整。

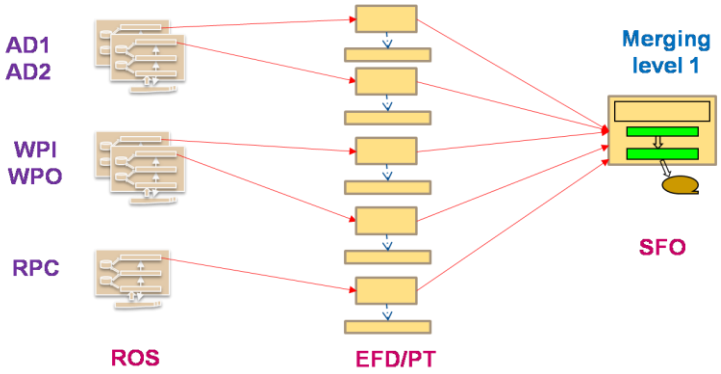
7.4.2 数据流合并的原则

对于数据流合并的原则，DAQ 系统将采取时间排序的方案，即提取出参加合并的所有数据流中事例数据的时间标签，按照时间进行排序，从而将一组数据流合并为单一数据流。

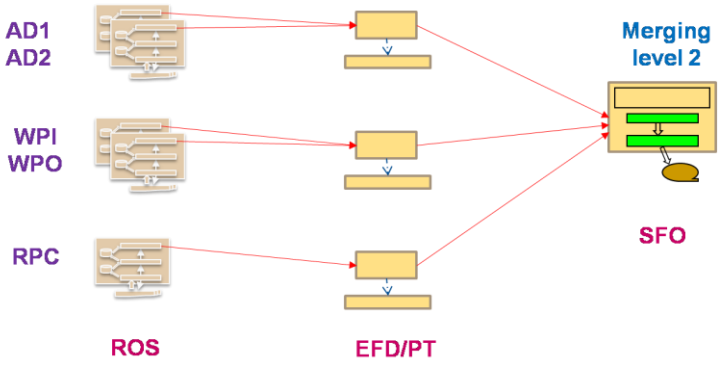
7.4.3 数据流合并的位置

如图 7-2 所示，目前考虑了两种数据流合并的方案。第一种方案是，每个独立数据流的走向与单探测器的情形基本相同，即 ROS 将数据通过 EFD 发送给 SFO，一个 ROS 对应于一个 EFD，不同的是数据在 SFO 级进行合并，形成单一数据流后再进行存盘。这种方案的优点是，只需要进行一级合并，设计实现比较简单，缺点是可能给 SFO 造成较大的负载压力。第二种方案是，一个站点分布三个 EFD，分别接受来自 AD、WP、RPC 探

测器的数据，即在 EFD 就进行了第一级的数据流合并，将来自同一类探测器的不同机箱的数据流进行合并，之后 EFD 再将合并后的三个数据流送往 SFO，由 SFO 进行再一次的合并后，形成最终的单一数据流进行存盘，这个方案的优点是可以通过 EFD 来分担 SFO 的负载压力，但是进行两级的数据流合并在实现上会相对复杂。因此最终采取哪种方案，还需要根据实验的具体情况以及系统的硬件配置等综合考虑决定最佳的方案。



（a）只在 SFO 进行一级合并



（b）在 EFD 进行第一级合并，SFO 进行第二级合并

图 7-2 数据流合并的两种方案

7.5 本章小结

本章简要介绍了大亚湾实验数据获取系统对于多探测器取数的扩展设计，包含了针对 RPC 电子学系统的数据获取系统的设计考虑，多探测器取数模式下的同步控制的设计，多探测器取数模式下读出系统的设计思想，并且针对多探测器取数的情形，讨论了数据流合并的设计方案。

8 大亚湾 miniDryRun 实验

8.1 miniDryRun 实验介绍

2009 年 12 月 21 号到 2010 年 1 月 12 号在大亚湾实验站进行了基于单 AD 探测器的系统联调实验，称为 miniDryRun 实验。实验中只使用了一个 AD 探测器系统，并且探测器的组件尚未完全安装到位。图 8-1 所示是 miniDryRun 实验中的 AD 探测器的俯瞰图[63]。实验中使用了位于 ladder #7 的 24 个 PMT，以及一个位于 AD 底端靠近 ladder #7 的 2” calibration PMT。

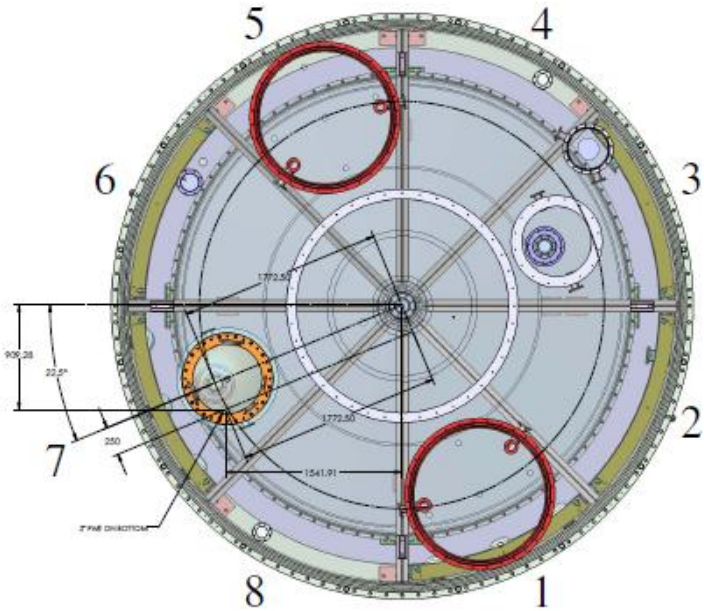


图 8-1 miniDryRun 实验中的 AD 俯瞰图

miniDryRun 实验中的探测器电子学系统位于地面装配大厅（surface assembly building，SAB）。实验的电子学系统包括：一个 48 通道的高压机箱，一个 48 通道的 PMT decoupler 模块，一个包含了系统控制器、LTB 板、扇出板以及 3 块 16 通道读出 FEE 板的 VME 机箱，如图 8-2 所示。

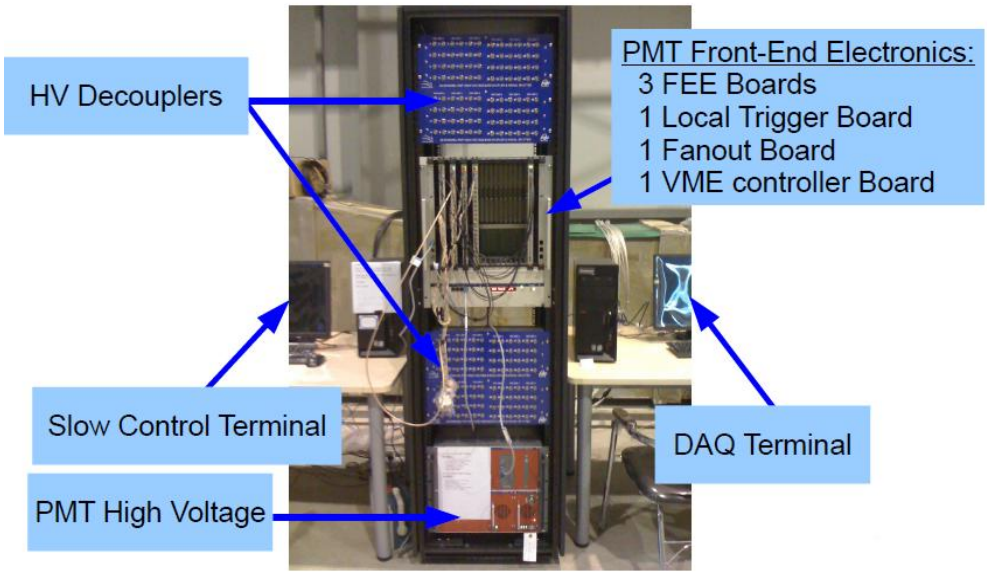


图 8-2 miniDryRun 实验中的电子学系统

实验中的 25 个 PMT 通过电缆连接到 FEE 电子学板的 25 个读出通道，它们之间的连接关系如表 8-1 所示。

表 8-1 miniDryRun 实验中 PMT 与 FEE 连接关系表

536937985	SABAD1-board06-connector01	0	Fee-6	536936723	SABAD1-ring01-column19	0	Pmt8inch-19
536937986	SABAD1-board06-connector02	0	Fee-6	536936979	SABAD1-ring02-column19	0	Pmt8inch-43
536937987	SABAD1-board06-connector03	0	Fee-6	536937235	SABAD1-ring03-column19	0	Pmt8inch-67
536937988	SABAD1-board06-connector04	0	Fee-6	536937491	SABAD1-ring04-column19	0	Pmt8inch-91
536937989	SABAD1-board06-connector05	0	Fee-6	536937747	SABAD1-ring05-column19	0	Pmt8inch-115
536937990	SABAD1-board06-connector06	0	Fee-6	536938003	SABAD1-ring06-column19	0	Pmt8inch-139
536937991	SABAD1-board06-connector07	0	Fee-6	536938259	SABAD1-ring07-column19	0	Pmt8inch-163
536937992	SABAD1-board06-connector08	0	Fee-6	536938515	SABAD1-ring08-column19	0	Pmt8inch-187
536937993	SABAD1-board06-connector09	0	Fee-6	536936724	SABAD1-ring01-column20	0	Pmt8inch-20
536937994	SABAD1-board06-connector10	0	Fee-6	536936980	SABAD1-ring02-column20	0	Pmt8inch-44
536937995	SABAD1-board06-connector11	0	Fee-6	536937236	SABAD1-ring03-column20	0	Pmt8inch-68
536937996	SABAD1-board06-connector12	0	Fee-6	536937492	SABAD1-ring04-column20	0	Pmt8inch-92
536937997	SABAD1-board06-connector13	0	Fee-6	536937748	SABAD1-ring05-column20	0	Pmt8inch-116
536937998	SABAD1-board06-connector14	0	Fee-6	536938004	SABAD1-ring06-column20	0	Pmt8inch-140
536937999	SABAD1-board06-connector15	0	Fee-6	536938260	SABAD1-ring07-column20	0	Pmt8inch-164
536938000	SABAD1-board06-connector16	0	Fee-6	536938516	SABAD1-ring08-column20	0	Pmt8inch-188
536938497	SABAD1-board08-connector01	0	Fee-8	536936725	SABAD1-ring01-column21	0	Pmt8inch-21
536938498	SABAD1-board08-connector02	0	Fee-8	536936981	SABAD1-ring02-column21	0	Pmt8inch-45
536938499	SABAD1-board08-connector03	0	Fee-8	536937237	SABAD1-ring03-column21	0	Pmt8inch-69
536938500	SABAD1-board08-connector04	0	Fee-8	536937493	SABAD1-ring04-column21	0	Pmt8inch-93
536938501	SABAD1-board08-connector05	0	Fee-8	536937749	SABAD1-ring05-column21	0	Pmt8inch-117
536938502	SABAD1-board08-connector06	0	Fee-8	536938005	SABAD1-ring06-column21	0	Pmt8inch-141
536938503	SABAD1-board08-connector07	0	Fee-8	536938261	SABAD1-ring07-column21	0	Pmt8inch-165
536938505	SABAD1-board08-connector09	0	Fee-8	536938517	SABAD1-ring08-column21	0	Pmt8inch-189
536938504	SABAD1-board08-connector08	0	Fee-8	536936449	SABAD1-ring00-column01	0	Pmt2inch-1

数据获取系统和慢控系统的控制终端位于 SAB，实验中记录下的数据文件同时自动传输往高能物理所与 LBNL（Lawrence Berkeley National Laboratory）。

miniDryRun 实验中，包括 PMT、电子学、DAQ、ACU LED、离线等相关的各系统进行了第一次较为完整的集成调试。在实验中测试了各个系统的工作状况以及系统之间的衔接情况，并且利用 LED 对探测器的性能进行了初步的研究。本章接下来的一节内容将主要介绍利用 LED 对探测器的

触发效率进行的分析研究。

8.2 基于 miniDryRun 实验的触发效率的研究

8.2.1 研究意义

大亚湾中微子实验以测量θ_{13}为目的，实验中的中微子实际事例数=探测到的中微子事例数/（触发效率*事例选择效率*其它各种效率），因此触发效率的大小会直接影响θ_{13}的测量结果。同时，因为触发阈值的设置会直接影响触发的效率，例如随着 ESum(即总能量)阈值的增大，或 Nhit(即有信号的 PMT 数目)阈值的增大，触发效率会受到影响而降低，因此如果给出触发效率随阈值变化的情况，便能够对触发阈值的设置起到一定的指导作用，使得触发阈值的设置既能够尽可能的排除本底，又不影响好事例的正确记录。

8.2.2 实验方法

在 miniDryRun 实验中，触发效率的研究是针对 LED 事例进行的。数据获取的过程如下：驱动位于 AD 探测器中 PMT ladder #7 正前方的 LED（ao0）发光，位于 ladder #7 的 24 个 PMT 与 2” calibration PMT 接收到相应的光电子数并放大后，将输出电量送入对应 FEE 通道，FEE 将相关的电量信息 ESum 以及击中数信息 Nhit 送给 LTB 板后，LTB 板根据触发类型与触发阈值的设置给出相应的触发判选，最终符合条件的有效触发将驱动 FEE 以及 LTB 板的数据读出，DAQ 系统会将获取到的事例进行存盘。本章节用到的所有分析触发效率的 run 中，LED 的发光频率均设为 400Hz，并且 FEE 各通道的阈值均设置为 750（~2.5mV~0.25p.e. in 2e7 PMT gain）。

根据上述实验过程取到的数据便可用来研究 LED 事例的触发效率。触发效率=引起触发的 LED 事例的数目/LED 的真实发光次数。这里首先假设 LED 在工作时间内每次都做到有效发光（实际上 miniDryRun 的实验中探测到 LED 大约有百分之几的几率没有正常发光[64]），因此 LED 的发光次数=设置的 LED 发光频率*整个取数过程中允许触发的时间。综上所述，计算

触发效率的关键，就在于得到引起触发的 LED 事例数，即如何从数据文件中尽可能的排除本底事例的干扰，提取真实的 LED 事例的数目。

要从数据中提取 LED 事例，首先必须了解 LED 事例的特征。LTB 板上记录了每次触发的时间信息，精确到 LTB 板的时钟周期即 12.5ns。由于 LED 的发光是周期性的，因此可以预计到，由 LED 事例产生的触发之间的时间间隔应该也能呈现出周期性的分布。如图 8-3 所示，相邻触发的时间间隔集中分布在 2.5ms，以及 2.5ms 的整数倍的位置，当然也有少数呈现出不规则的分布，这是因为除了 LED 事例外，还会有一些本底事例（主要来自 muon）造成的触发。在 2.5ms 处的集中分布刚好符合了 LED 发光的周期，而相应的在 2.5ms*N（N 为正整数）处的分布则意味着这组相邻的事例之间丢失了 N-1 个 LED 事例。而导致 LED 事例丢失的原因主要有三点：第一，LED 没有成功发光；第二，LTB 板收集到的信息（电量和或是击中数）没有过阈，因此无法给出触发，从而导致事例数据没有被成功记录；第三，本底事例的存在间隔了原本符合时间间隔特征的 LED 事例。

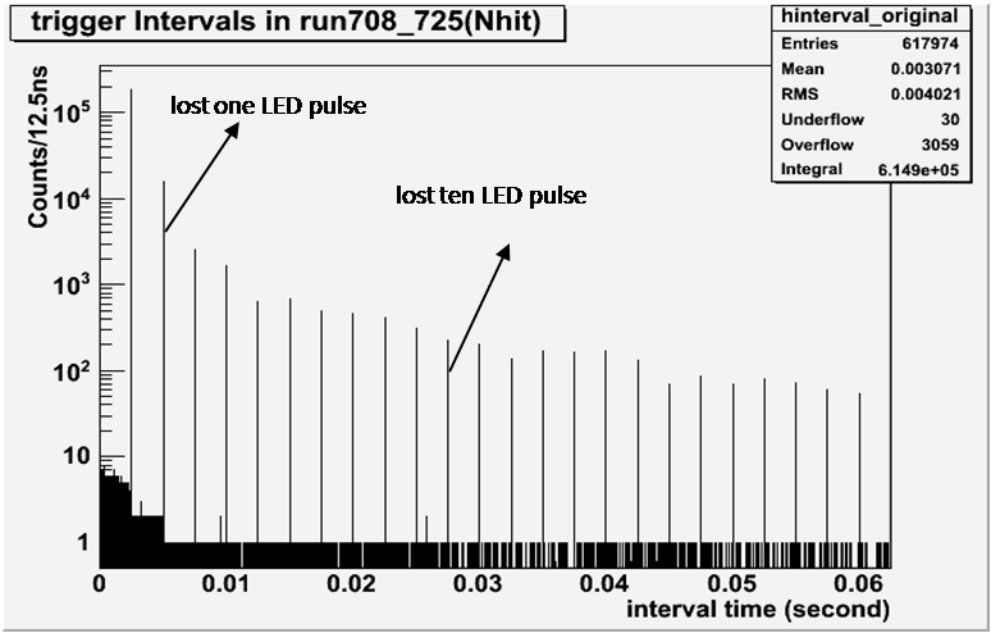
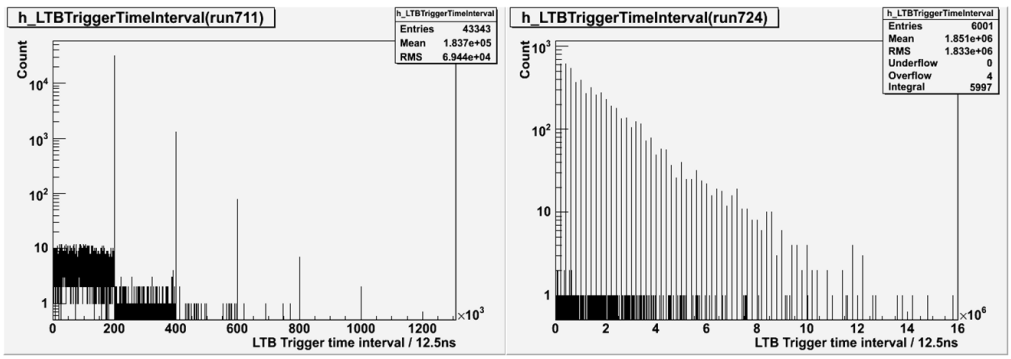


图 8-3 trigger time interval



(a)run711 触发阈值为 nhit>=5　　(b)run724 触发阈值为 nhit>=17

图 8-4 不同触发阈值条件下的 trigger interval

图 8-4 中以两组不同触发阈值设置下的数据为例来说明触发时间间隔分布的特征。左边的 a 图中采用了 run711 的数据，Nhit 触发类型，阈值设置为$\mathrm{Nhit} \geq 5$，因为触发阈值较低，真实的 LED 事例没有过阈而丢失的可能性较小，然而过阈噪声偏多，因此触发率较高（高于 LED 设置的发光频率 400Hz），run711 的实际触发率约为 432Hz。右边的 b 图采用了 run724 的数据，同样是 Nhit 触发类型，阈值设置为$\mathrm{Nhit} \geq 17$，因为触发阈值较高，导致过阈的噪声明显减少，而 LED 事例丢失很大比例，因此触发率明显降低，run724 的实际触发率约为 44Hz。

8.2.3 计算触发效率的第一种方法

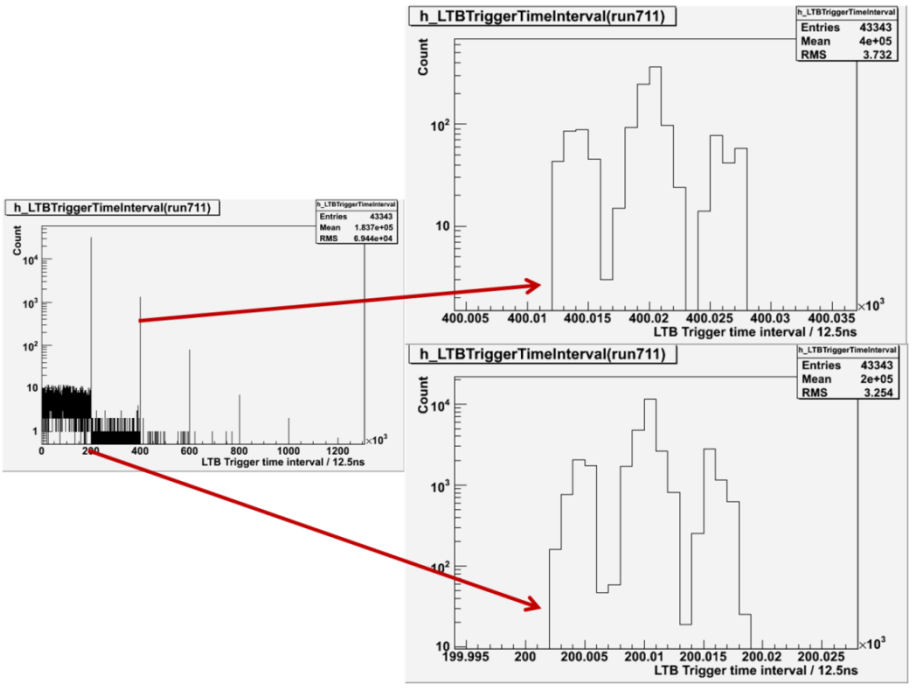


图 8-5 触发时间间隔分布情况

基于上一节中关于 LED 事例的触发时间间隔特征的讨论，首先考虑从数据中挑选出触发时间间隔符合 LED 周期的事例对作为真实事例的评判依据，即一组 LED 事例对的时间间隔应当满足$\mathrm{triggerInterval} = \mathrm{NT} \pm \Delta\mathrm{T}$，公式中 T 为 LED 发光周期 2.5ms，而$\Delta\mathrm{T}$为可以允许的时间晃动范围。从图 8-5 中放大后的触发时间间隔分布可以看到，每个符合周期特征的时间分布大约有 250ns 的宽度，因此可以粗略的将$\Delta\mathrm{T}$的最大允许波动范围设置为 250ns。在确定了 T 与$\Delta\mathrm{T}$后，根据$\mathrm{triggerInterval} = \mathrm{N} \times 2.5\mathrm{ms} \pm 250\mathrm{ns}$，从数据文件中挑选符合条件的 LED 事例对，得到的一组 run 的触发效率如图 8-6 所示，这里采用的是 run708 到 run725 的一组数据，触发类型为 Nhit，触发阈值从$\mathrm{Nhit} \geq 2$一直到$\mathrm{Nhit} \geq 18$，LED 驱动电压为-7.2V。

要理解图 8-6 中的效率曲线，不可避免的需要考虑本底事例对计算结果的影响：首先，在两个 LED 事例之间如果出现本底事例，会将一个本来符合 LED 周期的触发时间间隔分隔为两个无效的时间间隔；其次，如果两个本底事例的时间间隔偶然符合了 LED 的周期，也会伪造 LED 事例而影

响结果。

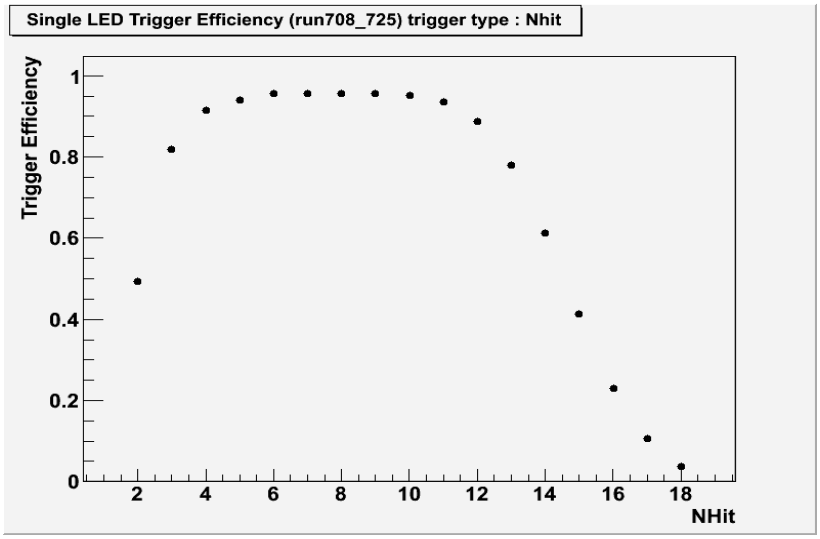


图 8-6 挑选 triggerInterval 方法得到的效率曲线

因此，为了尽可能的排除本底事例造成的影响，希望从数据中能够先将本底事例挑除。如图 8-7 所示，这是这组数据的 FEE ADC SUM 分布情况，由每次触发中所有有击中的 PMT 对应的 FEE 通道记录的 ADC 值相加得到一次触发的 ADC SUM，将这一组 run 的所有 ADC SUM 的值填成直方图，可以看到图中存在明显的分布，右边的峰值对应真实的 LED 事例。如图 8-7 中所示，设置 ADC SUM 的阈值，阈值以下的事例被判别为本底事例。

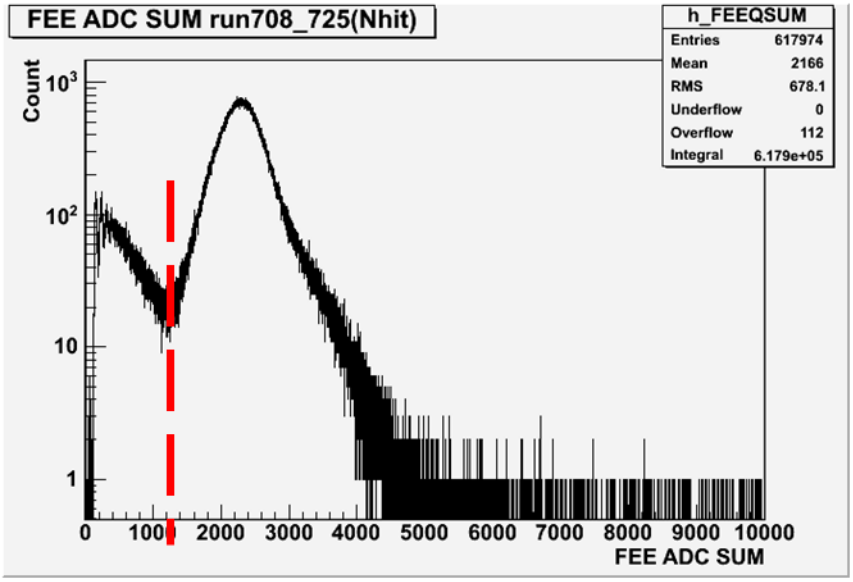


图 8-7 QSum 阈值的设置

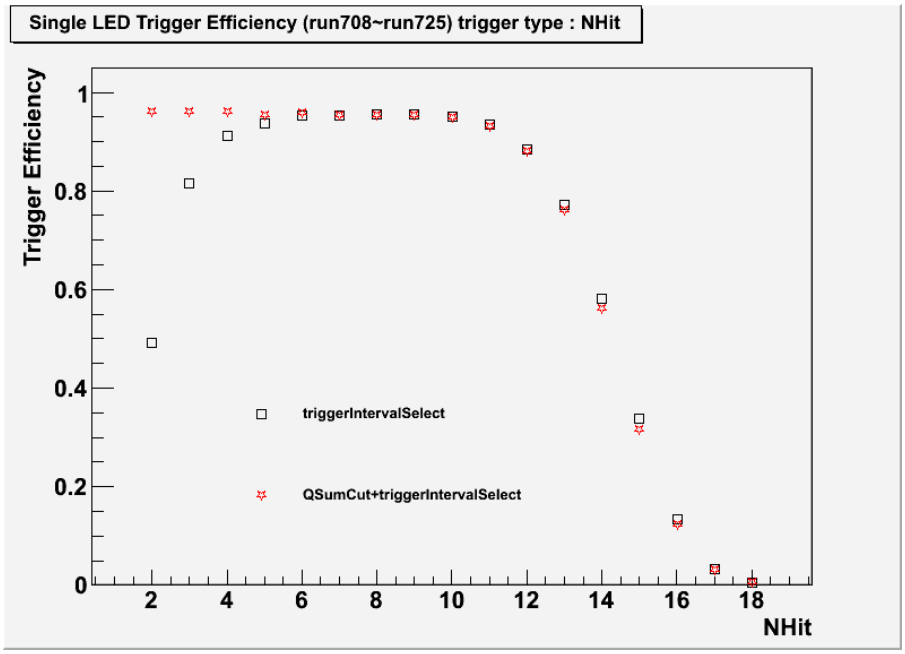


图 8-8 设置 QSumCut 对触发效率结果的影响

根据上述分析，先将本底事例从数据中剔除后，再挑选满足条件 $\mathrm{triggerInterval} = \mathrm{N} \times 2.5\mathrm{ms} \pm 250\mathrm{ns}$的事例对，重新计算得到的触发效率如图 8-8 所示。可以看到，加上 QSumCut 之后再挑选触发时间间隔的办法，比起直接挑选触发时间间隔的方法，得到的两条触发效率曲线主要的区别在于触发阈值设置较低时的几个触发效率值。这正验证了之前的分析：因为阈值较低时存在较多的本底事例，将满足时间间隔特征的 LED 事例对打断。采用 QSumCut 方法之后，剩下的事例为相对比较干净的 LED 事例，从而能够得到更准确的触发效率值。

表 8-2 实验中 24 个 PMT 的刻度参数与对应 FEE 的台阶值

Channel ID	pedestal	SPE
SABAD1-board06-connector01	74.352593675	5.73653391432
SABAD1-board06-connector02	74.1864093871	7.16432236858
SABAD1-board06-connector03	67.4980427363	7.95389964849
SABAD1-board06-connector04	60.675840558	6.14424574067
SABAD1-board06-connector05	60.8350072011	7.96962205523
SABAD1-board06-connector06	68.5505433145	6.27628843138
SABAD1-board06-connector07	63.6473879777	7.9197664092
SABAD1-board06-connector08	66.3495681848	7.60184796723
SABAD1-board06-connector09	53.33387661	5.18794198093
SABAD1-board06-connector10	66.7034575227	7.4124438732
SABAD1-board06-connector11	62.0385541798	7.97912285355
SABAD1-board06-connector12	69.6745341113	8.36557046422
SABAD1-board06-connector13	53.4118972136	7.05303939139
SABAD1-board06-connector14	65.3352162945	8.08454553096
SABAD1-board06-connector15	65.4077315628	8.14465436201
SABAD1-board06-connector16	54.8914115776	7.08866171348
SABAD1-board08-connector01	80.5734405583	9.02272903134
SABAD1-board08-connector02	60.0916908598	7.99276107917
SABAD1-board08-connector03	69.2768474175	9.14424283247
SABAD1-board08-connector04	85.728783545	8.96597076618
SABAD1-board08-connector05	68.4499258541	7.83835815653
SABAD1-board08-connector06	64.5937803305	7.1261264041
SABAD1-board08-connector07	70.7323290299	9.15872440628
SABAD1-board08-connector09	57.3862006821	8.3754298844

而事实上，由于各个 PMT 以及对应电子学通道具有一定程度的不一致性，为了得到更好的分析结果，这里将对 QSumCut 的方法做进一步的完善。

如表 8-2 所示，miniDryRun 中的其他实验已经提供了 24 个 PMT 的刻度参数以及对应的 FEE 通道的台阶值，因此可以将每个 FEE 通道数据的 ADC 值转化成对应的 PMT 接收到的光电子数目，即 $n_{p.e.}/PMT = (ADC - pedestal)/(SPE)$，所有有击中的 PMT 接收到的光电子数相加即得到每次触发中所有 PMT 接收到的光电子总数 peSum，peSum 的分布如图 8-9 所示，位于 200p.e.处的峰值则对应于驱动电压设置为-7.2V 时所有 PMT 探测到的 LED 总电量。

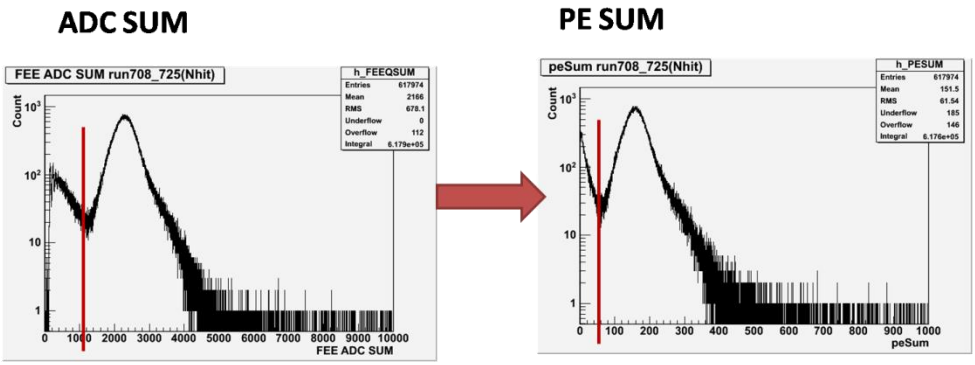


图 8-9 对 QSum 与 peSum 设置阈值条件进行 Cut

用 peSumCut 的方法替代之前较为粗略的 QSumCut 的方法，两种 Cut 条件的设置下得到的触发效率曲线如图 8-10 所示，可以看到这两种 Cut 条件下得到的触发效率是几乎完全一致的，原因将在 8.2.5 节中进行讨论。

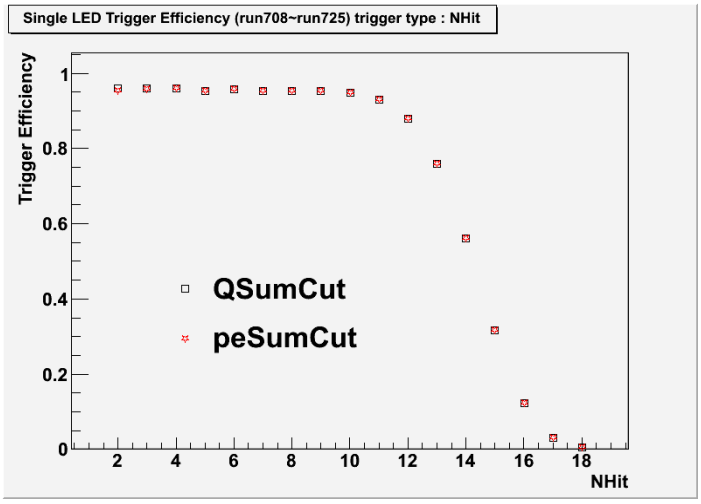


图 8-10 采用 QSumCut 与 peSumCut 得到的触发效率曲线对比

8.2.4 计算触发效率的第二种方法

为了对上一节中给出的触发效率的结果进行验证，本节将通过另一种方法对触发效率进行分析与计算。分析的过程如下：在每个 run 中，假定第一个记录到的事例为真实的 LED 事例，通过 LTB 板获取到的触发时间记为T_1。依据之前介绍的 LED 事例的时间间隔特征，在接下来的事例中寻找第二个 LED 事例，假定该事例的触发时间为T_2，则符合条件$T_2 - T_1 = N \times T \pm \Delta T$（其中 T 为 LED 发光周期，$\Delta T$为允许的时间波动范围，N 为非 0 整数）的第一个事例即为第二个 LED 事例。同样的方法去寻找第三个事例，符合条件$T_3 - T_2 = N \times T \pm \Delta T$的第一个事例即为第三个 LED 事例。以此类推，根据前一个 LED 事例的触发时间去寻找下一个事例，寻找完整个 run 的数据后，得到所有的 LED 事例总数记为$Count_1$。而事实上，数据文件中第一个记录下的事例并不一定是 LED 事例，因此需要改变起始事例的位置，重新假设第二个事例为第一次出现的 LED 事例，以此类推，得到在这样的假设下的 LED 事例总数$Count_2$。重复此过程 n 次（$n \geq 10$），得到的最大$Count_n$的值对应的起始事例应当被认为是数据中第一次出现的真实的 LED 事例。如图 8-11 所示，在该 run 的搜索过程中，数据文件中的第三个事例即 eventId=2 的事例是最终确定下的第一个 LED 事例的位置，而以该事例为首事例，依次找到的文件中满足时间间隔特征的总 LED 事例数为 21273。

```
start LED event id = 0;  number of subsequent LED event = 1
start LED event id = 1;  number of subsequent LED event = 4
start LED event id = 2;  number of subsequent LED event = 21272
start LED event id = 3;  number of subsequent LED event = 21271
start LED event id = 4;  number of subsequent LED event = 21270
start LED event id = 5;  number of subsequent LED event = 21269
start LED event id = 6;  number of subsequent LED event = 8
start LED event id = 7;  number of subsequent LED event = 6
start LED event id = 8;  number of subsequent LED event = 21268
start LED event id = 9;  number of subsequent LED event = 21267
start LED event id = 10;  number of subsequent LED event = 2
start LED event id = 11;  number of subsequent LED event = 5
start LED event id = 12;  number of subsequent LED event = 21266
start LED event id = 13;  number of subsequent LED event = 3
start LED event id = 14;  number of subsequent LED event = 21265
start LED event id = 15;  number of subsequent LED event = 4
start LED event id = 16;  number of subsequent LED event = 21264
start LED event id = 17;  number of subsequent LED event = 7
start LED event id = 18;  number of subsequent LED event = 1
start LED event id = 19;  number of subsequent LED event = 6
start LED event id = 20;  number of subsequent LED event = 7
start LED event id = 21;  number of subsequent LED event = 21263
start LED event id = 22;  number of subsequent LED event = 2
start LED event id = 23;  number of subsequent LED event = 21262
start LED event id = 24;  number of subsequent LED event = 4
start LED event id = 25;  number of subsequent LED event = 3
start LED event id = 26;  number of subsequent LED event = 3
start LED event id = 27;  number of subsequent LED event = 21261
start LED event id = 28;  number of subsequent LED event = 21260
start LED event id = 29;  number of subsequent LED event = 4
```

图 8-11 寻找有效 LED 事例示例

根据以上的分析方法，同样对 run708~725 的数据进行分析计算，得到的效率结果如图 8-12 所示。这里取$T = 2.5\text{ms}$，图中对应的三条效率曲线分别取$\Delta T = 2.5\text{us}$、$\Delta T = 1\text{us}$、$\Delta T = 250\text{ns}$，从图中看出这三种设置下的结果都不是很理想。结合分析的过程可以想到，此方法简单的说就是逐点寻找下一个 LED 事例，而如果在确定每个 LED 事例的过程中有一个事例的位置出现偏差（将本底事例错误判定为 LED 事例），便有可能导致之后的事例定位失败，因此对于此方法，T 与ΔT的准确设定就显得相当关键。

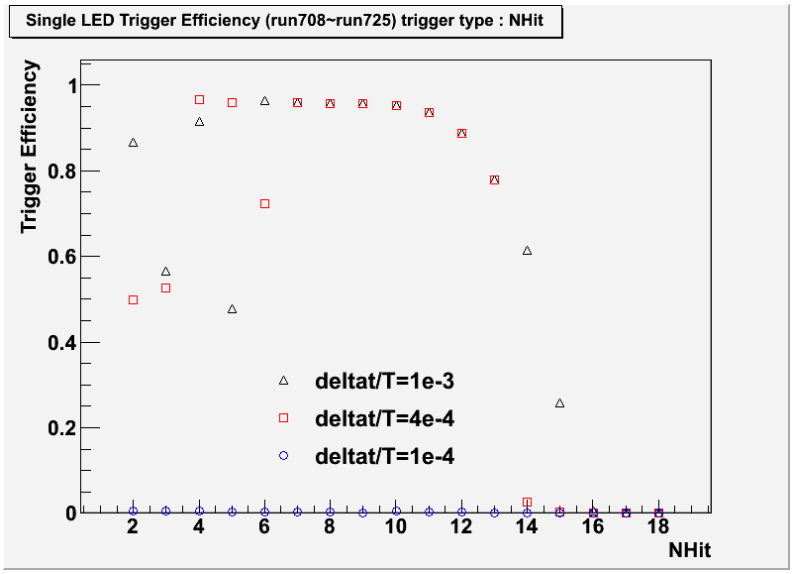


图 8-12 周期 T=2.5ms 时改变ΔT得到的触发效率情况

图 8-13 所示是放大后的触发时间间隔位于单倍 T 以及两倍 T 处的分布示意图。可以计算出，左边图中分布的平均值为 0.002500123s，而右边图中的平均值位于 0.005000247s，左右端距离中心值的偏移约为 150ns。由此可以得出结论，虽然 LED 的发光周期设置为 0.0025s，但是实际的发光周期与设置值有着细微的不一致，由此导致了图 8-12 中触发效率的计算结果出现了严重偏差。

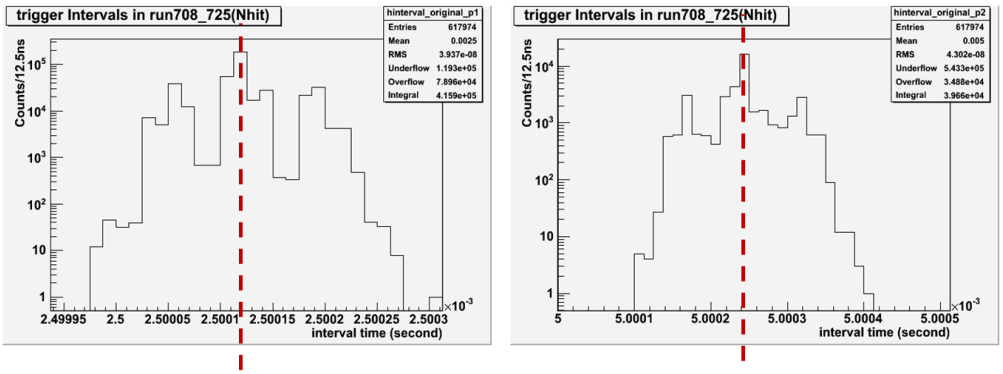


图 8-13 对应单倍与双倍 LED 周期的触发时间间隔分布

因此，根据图 8-13，修正 LED 发光时间的偏差，取$T = 2.500123\text{ms}$，$\Delta T = 250\text{ns}$，重新计算得到的触发效率曲线如图 8-14 所示。

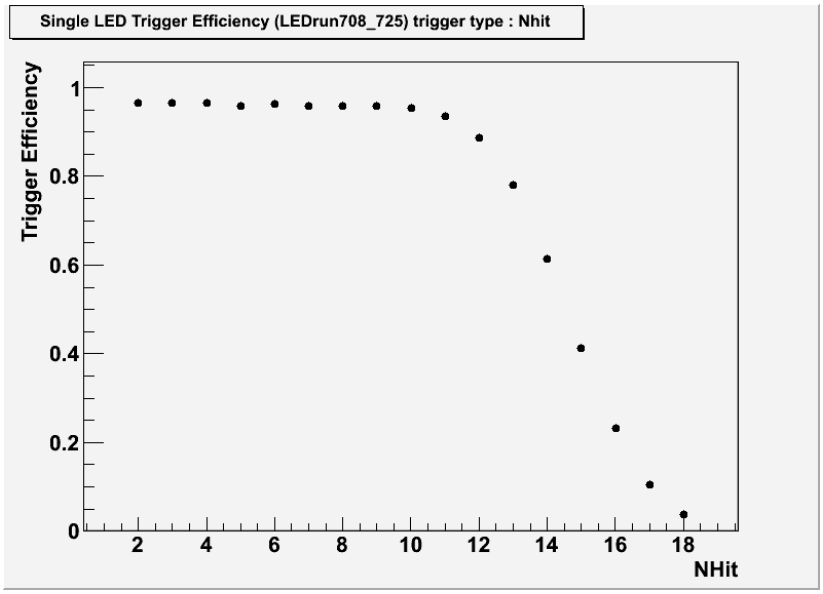


图 8-14 修正 LED 发光周期后得到的触发效率曲线

8.2.5 两种方法的比较

如图 8-15 所示，两条效率曲线符合的非常好，说明前面两节中介绍的两种不同的分析计算触发效率的方法所得到的结果是非常一致的。

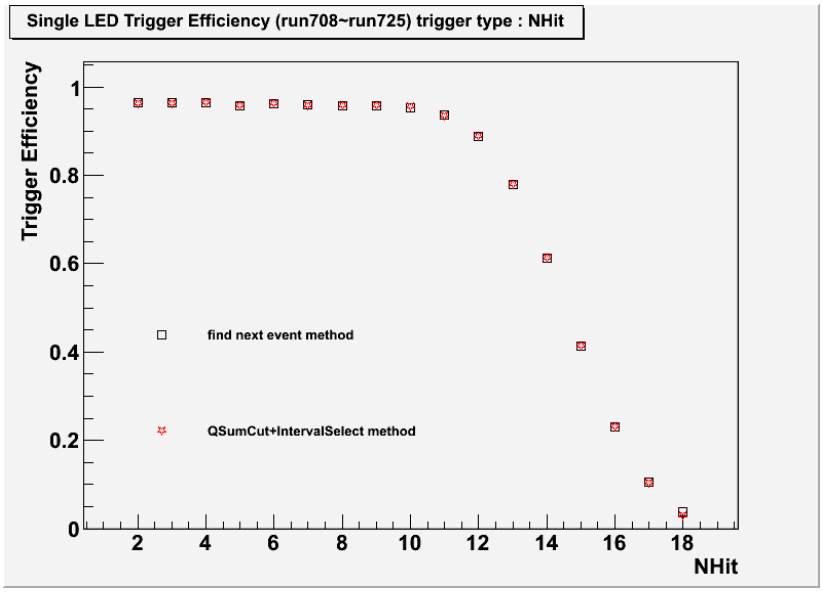


图 8-15 两种分析方法得到的效率曲线的比较

本节将重点讨论以上两种分析计算的方法的异同以及各自的优缺点。

首先，两种方法的相同点在于都利用了 LED 发光的周期性的时间特性；

而两种方法的区别在于，前一种方法不仅需要从 LTB 板获取触发时间的信息，还需要从 FEE 板获取通道电量 ADC 的值，以及各通道的台阶与刻度参数的值，而第二种方法仅仅需要获取 LTB 板的触发时间信息，因此从信息量来考虑，后一种方法较为简单易行。

第一种方法中选取 TriggerInterval 的时候，如果噪声偶然符合这个时间间隔，也会被选取进来，并且从图 8-7 以及图 8-9 中可以看出，在设置 Cut 阈值的时候，既不能完全排除本底，也有可能将 LED 事例排除在外。但是，虽然这种方法里的两个选择条件都不够精确，却可以起到互补的效果，因此该方法的优势在于，对于 T 和 ΔT 的选择不敏感，同时对于 Cut 阈值的选取也不敏感，因此这也导致了 peSumCut 与 QSumCut 得到的结果也相差无几。

第二种方法相对于前一种方法，由于综合考虑了事例的绝对时间与事例之间的时间间隔，能够更好的排除本底事例。然而通过分析计算的过程可以看到，这种方法对于 T 与ΔT的选取要求较高的精度，尤其需要修正 LED 的发光周期 T，否则将会使结果产生严重的偏差。

8.2.6 影响触发效率的因素

利用之前介绍的分析触发效率的方法，本节中将对 miniDryRun 实验中各种条件下的 LED 数据进行分析比对，从而给出实验中影响触发效率的因素。

8.2.6.1 触发阈值的设置对触发效率的影响

图 8-16 中采用了一组 ESum 触发的数据 run558~575，LED 驱动电压设置为-7.4V，PMT 增益为 2e7。图中的触发效率曲线反应了触发效率随着 ESum 触发阈值的变化情况，当 ESum 阈值的设置低于 160（~240p.e.~2MeV）时，触发效率基本能够稳定在 96%左右的范围内，当阈值大于 160 时，随着阈值的增加，触发效率便出现明显的下降趋势。

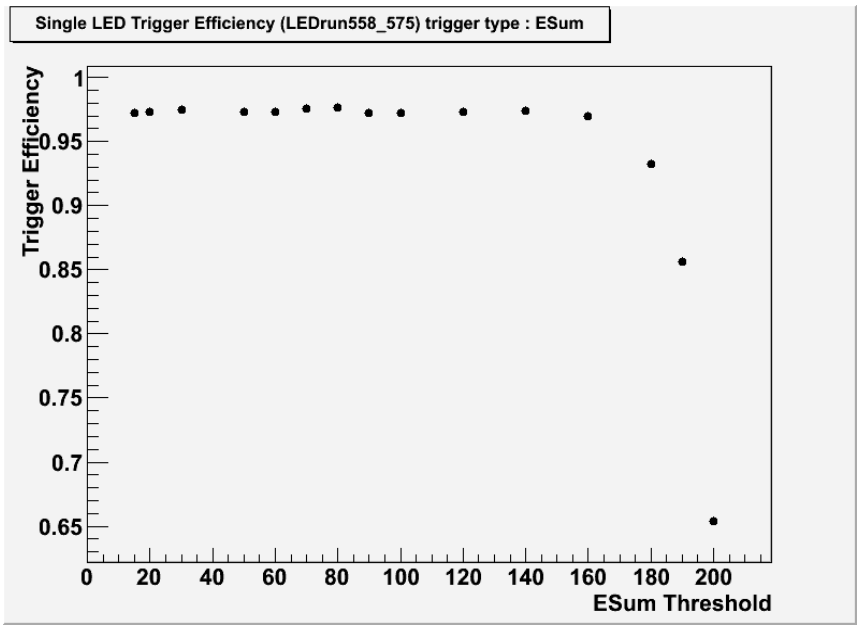


图 8-16 ESum 触发模式下阈值对触发效率的影响

而图 8-17 中展示了一组 Nhit 触发的数据 run542~555 得到的效率曲线，其中 LED 驱动电压为-7.4V，PMT 增益为 2e7，从图中可以看到 Nhit 阈值的设置对触发效率的影响。综合两种触发模式下的效率情况可以得出结论，对于不同的取数条件，触发阈值有一个相对应的合理的设置区间，在此区间内触发效率基本保持不变，而一旦阈值的设置过高，触发效率将随着阈值的增加而出现明显的下降。

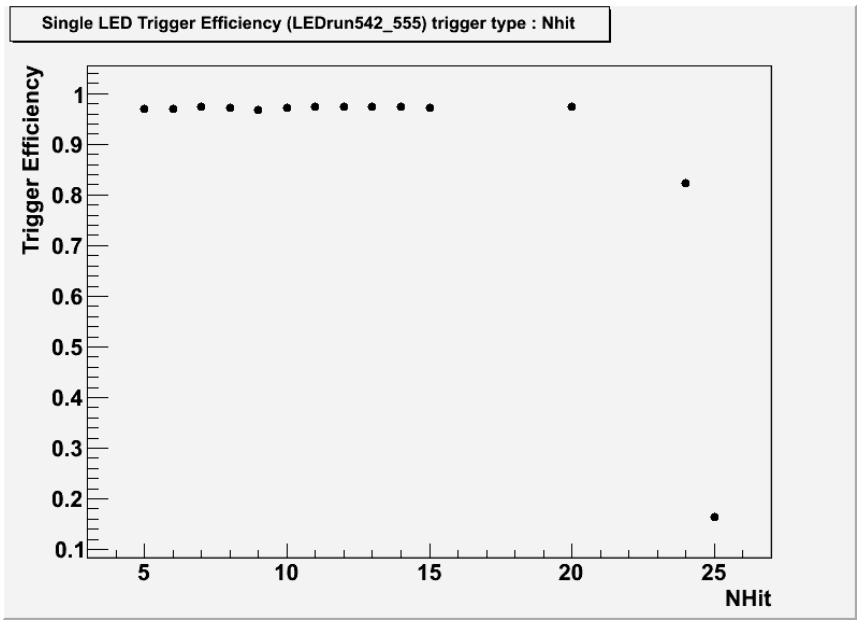


图 8-17 Nhit 触发模式下阈值对触发效率的影响

8.2.6.2 LED 光强对触发效率的影响

图 8-18 是由一组触发类型为 ESum 的数据 run605~615 计算得到的触发效率随 LED 驱动电压的变化曲线。触发阈值设置为ESum > 40（~60p.e.~0.5MeV），LED 驱动电压从-7.1V 增加到-9V，PMT 增益为 2e7。从图中的触发效率变化情况可以看到，当 LED 驱动电压设置为-7.1V 时，触发效率几乎为 0，而随着驱动电压的增加，触发效率迅速增长并且稳定在 96%左右的水平。

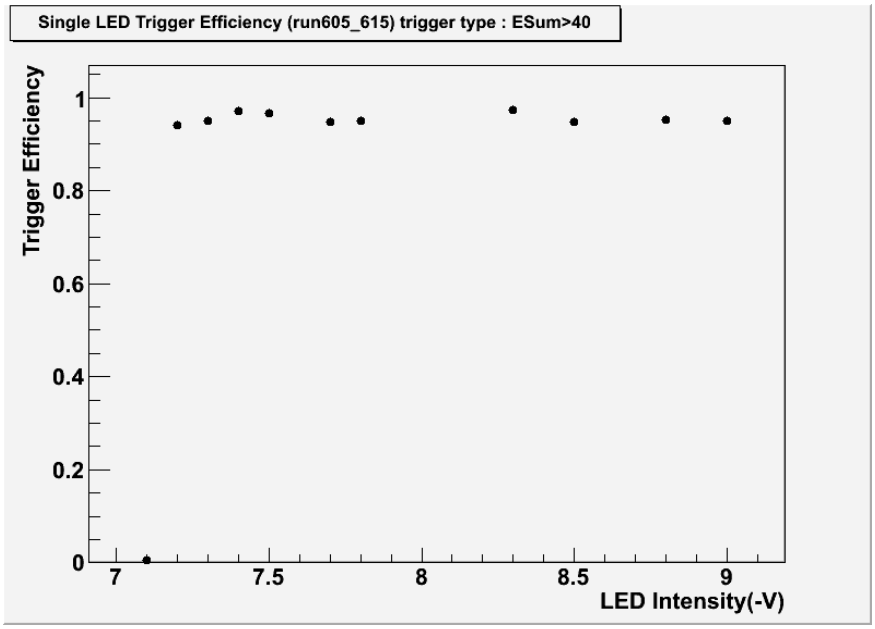


图 8-18 ESum 触发模式下 LED 光强对触发效率的影响

图 8-19 是一组 Nhit 触发模式的 run585~592 得到的触发效率曲线，触发阈值的设置为Nhit ≥ 8，PMT 增益设置为 2e7。从图中可以看到随着 LED 驱动电压幅度的提高（从-7.02V 到-7.4V），即 LED 光强的提高，触发效率得到了显著的提高。

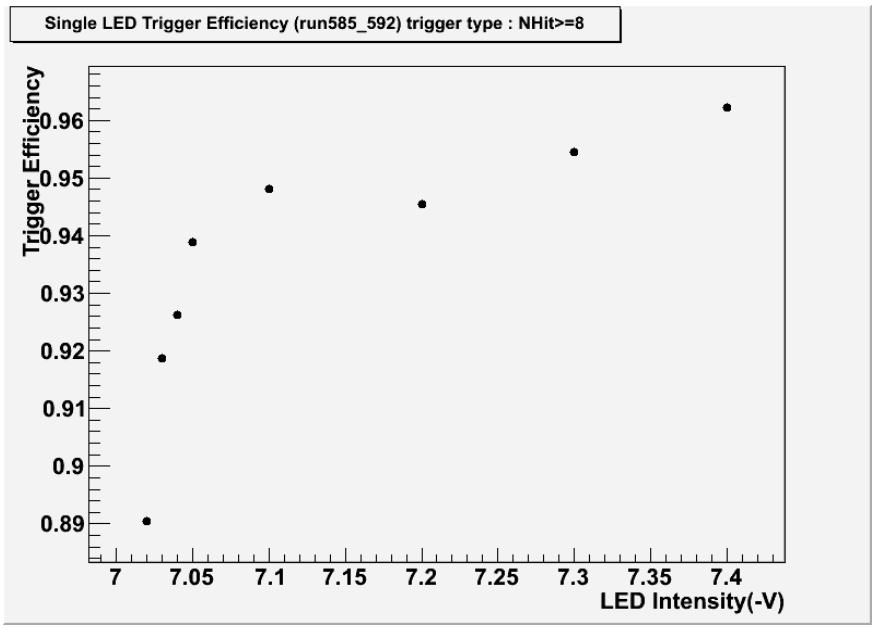


图 8-19 Nhit 触发模式下 LED 光强对触发效率的影响

8.2.6.3 PMT 增益对触发效率的影响

图 8-20 中展示了两组不同条件下的数据得到的触发效率曲线。这两组 run 都采用 Nhit 触发模式，其中 run542~555 中，LED 驱动电压为-7.4V，PMT 增益为 2e7；而 run708~725 中，LED 驱动电压为-7.2V，PMT 增益为 1e7。通过两组数据的比较可以看出，在触发阈值设置较低时，触发效率并无明显差别，随着阈值的增大，PMT 增益与 LED 光强的差异对触发效率起到显著的影响（其中还包括了 FEE 阈值的影响：FEE 阈值设置均为 750，在 2e7 的增益下约为 0.25p.e.，而在 1e7 的增益下约为 0.5p.e.）。

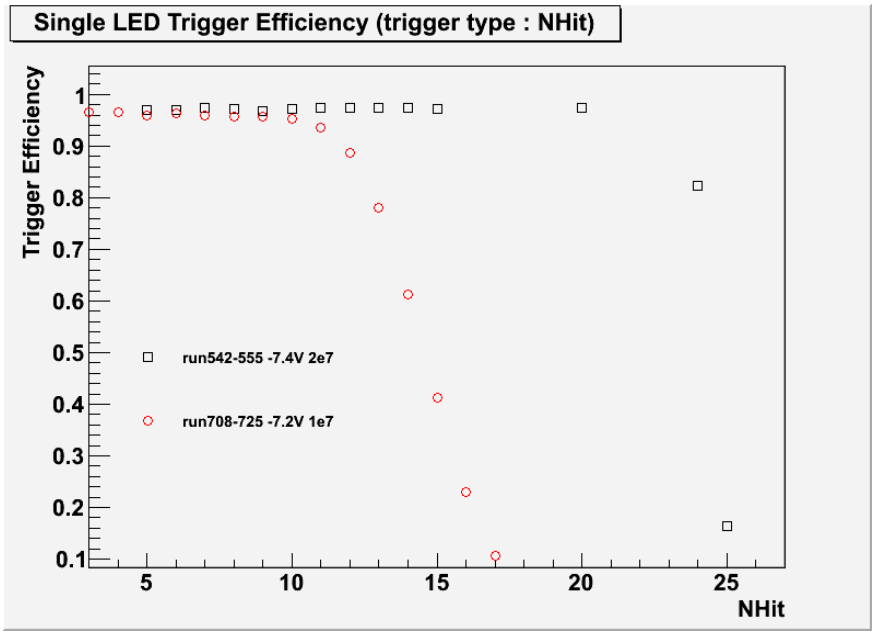


图 8-20 PMT 增益对触发效率的影响

8.2.6.4 PMT 与 LED 的相对位置对触发效率的影响

图 8-21 展示了 FEE 击中率的分布情况，每个通道的击中率 hitRate=该通道有击中的触发次数/总触发次数。根据 miniDryRun 中 PMT 与 FEE 通道的对应关系，从 FEE_hitRate 可以看出，ladder #7 中三个 column 的中间三圈 ring4、ring5、ring6 的击中率相对较高，而每个 column 的 ring1 和 ring8 击中率相对最低。这是因为实验中用到的 LED 位于 ladder #7 正前方纵向正中间的位置，所以位于纵向中间的三圈 PMT 击中率较高，而离 LED 最远的 ring1、ring8 两圈 PMT 的击中率较低。不同 FEE 通道的击中率呈现出与相对 LED 位置相关的分布，这也就解释了图 8-20 中为什么随着 Nhit 阈值的增加，触发效率会呈现下降的趋势。

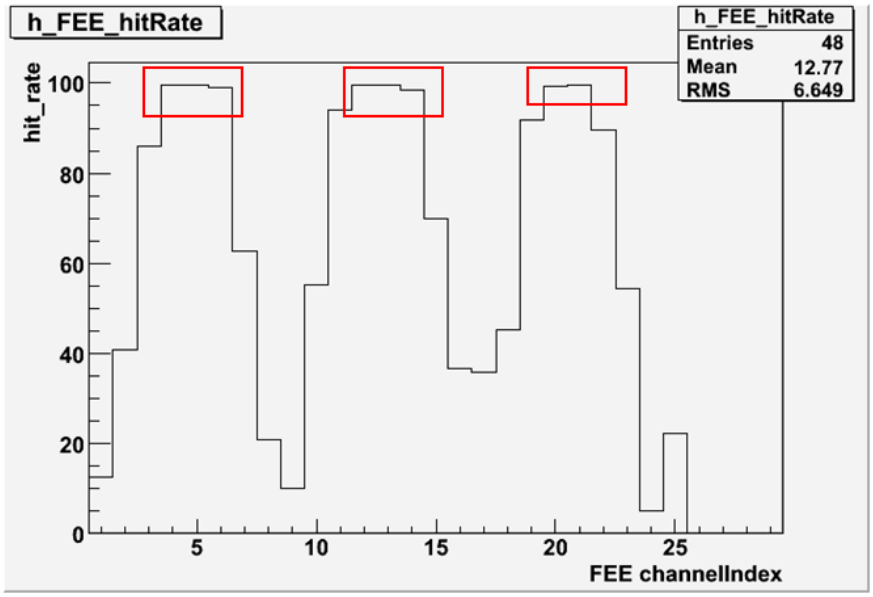


图 8-21 FEE 击中率 hit_rate 分布

8.2.6.5 小结

综上所述，在基于 miniDryRun 实验数据进行的触发效率研究的过程中，影响触发效率的因素主要有：

(1). 触发阈值的设置：在 LED 光强以及 PMT 增益都确定的情况下，触发阈值的设置存在一个合理的区间，在此区间内所有好事例都应当能够正确的产生触发，而触发阈值设置过高后，将导致 LED 事例无法过阈，从而影响触发效率。

(2). LED 光强：LED 光强较弱时，LED 事例会由于在 PMT 产生的信号过弱无法过阈（包括 FEE 板的阈值以及 LTB 板的触发阈值）而导致触发效率偏低。

(3). PMT 增益：同样的 LED 光强下，PMT 增益的提高会使 FEE 板接收到的信号强度增大，从而提高触发效率。

(4). PMT 与 LED 的相对位置关系：离发光 LED 位置最近的 PMT 的击中几率要高于其他 PMT，因此发光 LED 与所有参与实验的 PMT 的相对位置也会对触发效率产生影响。

8.2.7 关于触发效率的研究小结

前面几节详细的分析了在 miniDryRun 的实验条件下，利用 LED 事例对触发效率进行研究的情况以及得到的相关结论。通过改变实验条件，比较触发效率的变化情况从而给出了实验中影响触发效率的几个主要因素。

在进行触发效率研究的过程中，需要提取数据文件中 LTB 板与 FEE 板中的相关电子学信息，利用这些信息分析、计算触发效率，并且借助这些信息能够更好的理解实验结果。分析数据的过程也是对整个电子学 DAQ 系统的一个完整的检验过程。在对触发效率进行研究的过程中，发现的问题有：

(1). LED 有大约 3~4%的几率没有发光

(2). LED 的发光周期的偏差（~120ns）

(3). LTB 板的触发时间记录有误（该问题在新版本的 LTB 逻辑中已经解决）

8.3 miniDryRun 实验中的 DAQ 系统

从 2009 年 12 月开始，DAQ 系统共取数 1069 个 run，其中 miniDryRun 期间取数约 500 个 run，存盘 2054 个数据文件，合计文件大小约 29GBytes。

DAQ 系统在 miniDryRun 期间得到了充分的测试，包括了除在线直方图显示以外的各项基本功能，例如电子学配置、数据库功能、数据流传输、文件存盘等等，与各系统的接口都能够正常工作，运行模式中的物理取数、台阶模式以及电子学自诊断模式都在实验期间发挥了相应的作用。实验中进行了约 2.5 个小时的稳定性测试，整个系统运行稳定，取数过程中没有异常现象。

并且为了方便用户的使用与访问，DAQ 系统的 IGUI 用户界面提供了对运行记录信息（MRS log）的显示功能，同时也提供了通过网页访问数据库信息的功能，包括了配置信息、运行参数等。

miniDryRun 实验也为 DAQ 系统提供了一些反馈信息，例如电子学与 DAQ 系统应当具备对缓冲区的监测功能，以便能够对系统死时间进行分析；

DAQ 系统需要提供运行备注的日志，以便在将来的实验中能够更好的对取数运行进行组织管理等等，DAQ 系统会及时根据用户的反馈对系统进行相应的升级，使系统在正式取数之前尽可能的准备充分，以满足系统与用户的要求。

整个 miniDryRun 实验期间，DAQ 系统表现可靠、性能稳定，能够满足实验需求，通过 miniDryRun 实验的检验，相信 DAQ 系统能够在即将到来的大亚湾实验的运行中稳定的发挥其重要作用。

8.4 本章小结

本章介绍了 2009 年 12 月到 2010 年 1 月在大亚湾实验站进行的 miniDryRun 实验，该实验是基于部分组装完成的单 AD 探测器的一次系统集成测试，实验涉及了 PMT、电子学、DAQ、慢控、离线以及刻度等各个系统。进行 miniDryRun 实验可以对整个探测器电子学系统有一个初步的检验，并且通过 LED 实验进一步的了解系统的整体状况。本章重点介绍了利用 LED 实验进行的相关触发效率的分析，最后简单介绍了 miniDryRun 实验中数据获取系统的工作情况。

9 总结

本论文介绍了大亚湾中微子实验数据获取系统的研究与实现。大亚湾实验的 DAQ 软件在借鉴了 ATLAS TDAQ 与 BESIII DAQ 软件的基础上，针对大亚湾实验的具体要求进行了设计开发。

论文中主要以 AD 探测器的 PMT 读出电子学为硬件平台，介绍了数据获取系统的设计思想与实现方案。通过与 PMT 电子学系统的联合调试检验了系统的可行性与稳定性，对系统运行情况做出了详细的分析。并且在此基础上提出了多探测器取数模式下的数据获取系统的核心设计思想与实现方案。

本论文重点介绍了数据获取系统中的读出系统的研究与实现。大亚湾数据获取读出系统创新性的使用了嵌入式 Linux 作为软件开发平台，为验证这一方案的可行性进行了充分细致的测试与研究，对遇到的问题提出了解决措施，对影响系统性能的软件瓶颈提出了优化方案。在实现基于嵌入式 Linux 的读出系统的过程中综合使用了 VME 总线传输、中断响应、多线程通讯、缓冲区保护以及数据库等一系列技术，这些技术手段的应用对数据获取读出系统的实现和性能的提高起着关键的作用。其中对系统性能的关注是本论文工作的一大特点，这符合 DAQ 系统“数据重于一切”的工作原则。凡是有可能成为系统瓶颈的地方，都通过实验数据进行了细致的研究分析。

基于 AD 机箱的数据获取系统已经成功应用于在大亚湾实验站进行的 AD miniDryRun 的实验中，系统运行稳定，为实验的成功提供了有效的保障。论文最后介绍了基于 miniDryRun 的实验数据对触发效率进行的研究以及得到的初步结论。

2009 年 12 月，本论文中重点讨论的针对大亚湾实验中 PMT 读出电子学开发的数据获取系统的最终设计方案通过了专家组的评审。然而软件开发是一个不断迭代和螺旋渐进的过程，大亚湾数据获取系统仍然有很大的

改进空间，需要更多的调试，在实践的检验与终端用户的有益反馈中逐渐走向完善，从而在将来大亚湾实验的运行中发挥重要的作用。

参考文献

[1] 郑志鹏，朱永生, 北京谱仪正负电子物理. 1998.广西科学技术出版社.

[2] 谢一冈，陈昌，吕军光等, 粒子探测器与数据获取. 2003.科学出版社.

[3] 中国科学院高能物理研究所, 大亚湾反应堆中微子实验初步设计报告，内部文献. 2007.

[4] 中国科学院高能物理研究所, 大亚湾实验综述，内部文献. 2007.

[5] 朱科军，高能物理实验中的数据获取，第十二届全国计算机、网络在现代科学技术领域的应用学术会议，2005.

[6] 朱科军，BESIII 数据获取系统，内部报告. 2007.

[7] CMS Collaboration, CMS Technical Proposal. 1995. CERN/LHCC/Trigger and Data Acquisition. http://cmsinfo.cern.ch/TP/TP.html.

[8] ROD Crate DAQ Task Force. Data Acquisition for the ATLAS Read-out Driver Crate (ROD Crate DAQ) - Definition. 2002.

[9] BESIII 合作组，BESIII 设计报告（中文版），中国科学院高能物理研究所，2003.

[10] ATLAS High-Level Trigger, Data Acquisition and Controls Technical Design Report. 2003.

[11] ATLAS DAQ Backend Software Group. ATLAS DAQ Back-end software High-Level Design. 1998.

[12] 任震宇. BESIII DAQ 事例组装研究与 MDC 模型联调: 博士论文. 北京: 中国科学院高能物理研究所, 实验物理中心, 2006.

[13] H. P. Beck, S. Gadomski, C. Häberli, S. Klous, C. Meessen, A. Negri, and S. Wheeler, EFIO: Protocol Specification,

DataCollection Note 035. [Online]. Available: https://edms.cern.ch/document/391570.

[14] 李飞, BESIII DAQ 分布式数据监测研究与实现，博士论文. 北京: 中国科学院高能物理研究所, 实验物理中心, 2007.

[15] Fei Li，The raw data file format in the Daya Bay DAQ(v0.6)，June 2009.

[16] VME Technology FAQ，http://www.vita.com/vme-faq/.

[17] Motorola, MVME5500 Series VME Single-Board Computer data sheet.

[18] Tundra Universe II，http://embedded-computing.com/tundra-universe-ii.

[19] 范永开，杨爱林，Linux 应用开发技术详， 2006.人民邮电出版社.

[20] 魏忠，蔡勇，雷红卫，嵌入式开发详解，2003.电子工业出版社.

[21] 王齐，Linux PowerPC 详解—核心篇，2008.机械工业出版社.

[22] 黄思源，嵌入式 Linux 应用现状和趋势，中科红旗软件技术有限公司，2003.

[23] https://linuxlink.timesys.com/3/Products.

[24] GE Fanuc Automation. VME_UNIVERSE: VMEbus Driver and Tools for Linux.2004

[25] https://linuxlink.timesys.com/forum/522# What_is_the_VMIC_VME_Universe_dr.

[26] http://www.vita.com/vme-faq/readwritecyc.html.

[27] Tundra Semiconductor Corporation. Universe II VME-to-PCI Bus Bridge User Manual. 2002.

[28] 朱海涛. BESIII DAQ 读出子系统初步设计和框架研究，博士论文. 北京: 中国科学院高能物理研究所, 实验物理中心, 2007.

[29] ROD Crate DAQ Task Force. Data Acquisition for the ATLAS Read-out Driver Crate (ROD Crate DAQ) - Definition. 2002.

[30] http://atlas.web.cern.ch/Atlas/GROUPS/DAQTRIG/ ReadOut/ros.php.

[31] Soh Yamagata Suzuki, Masanori Yamauchi, Mikihiko Nakao, Ryosuke Itoh, Hirofumi Fujii, The BELLE DAQ system, Nuclear Instruments and Methods in Physics Research A 453 (2000) 440-444

[32] M. Nakao, M. Yamauchi, S. Y. Suzuki, R. Itoh, H. Fuji, Data Acquisition System for the Belle Experiment, lEEE TRANSACTIONS ON NUCLEAR SCIENCE, VOL. 47, NO. 2, APR 2000

[33] WindRiver Inc. VxWorks Programmer's Guide 5.4. 1.

[34] Motorola Computer Group. MVME2400 Series VME Processor Modules data sheet. 2001.

[35] 雷广坤. BESIII DAQ 前端读出系统优化设计与工程实现，博士论文. 北京: 中国科学院高能物理研究所, 实验物理中心, 2008.

[36] Rik Hemsley，Context switch overhead timing，http://rikkus.info/context-switch-timing.html，Feb 2005.

[37] Motorola Computer Group, MVME5100 Series VME Processor Modules data sheet.

[38] 朱海涛、陶宁，读出程序说明书- UDOL-0101-0114(v1)，内部资料，Sep 2005.

[39] 田雨润、李飞等，ATLAS 在线软件到 PowerPC-VxWorks 系统的移植，核电子学与探测技术，2005，Vol.25(No.6)：p. 647-650.

[40] Technical Committee on Microprocessors and Microcomputers of the IEEE Computer Society，IEEE Standard for A Versatile Backplane Bus: VMEbus，January 2008.

[41] M. Joos，A package for the allocation of contiguous memory on Linux systems(v1.6)，ATLAS Internal Note，November 2006.

[42] J. Almeida，et al. The ATLAS DAQ System Online Configurations Database Service Challenge，15th IEEE-NPSS Real-Time Conference, vol.1，2007.

[43] Robert Love 著，陈莉君等译， Linux 内核设计与实现，北京，机械工业出版社，2004.

[44] John Shapley Gray 著，张宁等译，UNIX 进程间通信（第二版）. 北京：电子工业出版社，2001.

[45] Fei Li，The raw event format in the Daya Bay DAQ(v0.6)，May 2009.

[46] DYB FEE Electronic Group，CBLT Data format for FEE(v1.3)，June 2009.

[47] Alex Gong, Hui Gong, Xi Wang，Dayabay Reactor Neutrino Experiment Local Trigger Board User manual(v3.8)，Dec 2009.

[48] DIM Homepage，http://dim.web.cern.ch/dim/.

[49] Hui Gong，et al. The Trigger System in Daya Bay neutrino experiment，the 1st international conference on Technology and Instrumentation in Particle Physics，Tsukuba，2009.

[50] 薛涛,龚光华,陈少敏,邵贝贝，大亚湾中微子实验时钟时标广播系统的初步设计，核电子学与探测技术，2009，Vol.29(No.6)：p. 1253-1256.

[51] Wu Qun，Event Readout Process，Daya Bay Collaboration meeting，CUHK，Jun 2009.

[52] CVS Homepage，http://www.nongnu.org/cvs/.

[53] 刘英杰、李飞等，在线数据获取系统软件开发管理平台，核电子学与探测技术，2006，Vol.27(No.6)：p. 808-812.

[54] LTB+FEE+DAQ test group，DAQ initialization flow for LTB+FEE (v1.0)，Aug 2009.

[55] DYB FEE Electronics Group，DYB FEE VME Interface

Descriptions(v.5)，Oct 2009.

[56] John Rynearson，VMEbus Speed / Bandwidth，VITA，Mar 1997.

[57] Zheng Wang，FEE design and performance，PMT Readout Review，IHEP：Jun 2009.

[58] The ROOT team，The ROOT Users Guide(v5.21)，CERN，Dec 2008.

[59] 陈丽平. BESIII 在线非物理取数模式的研究与实现，硕士论文. 北京: 中国科学院高能物理研究所, 实验物理中心, 2009.

[60] Zheng Wang，FEE Test Process(v1)，July 2009.

[61] FEE Group，Daya Bay Project PMT FEE Board Technical Design Report(Draft 2st)，June 2008.

[62] Muon Electronics Group，RPC Electronics Review Design Report，USTC，Oct 2009.

[63] Dan Dwyer，Mini Dry Run Overview，Mar 2010.

[64] Mini Dry Run Group，Mini Dry Run Action Items(v5.0)，Feb 2010.

附录A 论文中常见词汇解释

ACU(Automatic Calibration Unit): 大亚湾实验中的刻度系统

AD(Anti-neutrino Detector): 反中微子探测器，即中心探测器，大亚湾中微子实验中探测器的一种

ADC(Analog to Digital Converter): 模数转换器，在本文中特指模数转换器的输出值

ADOH(Address-Only-with-Handshake): VME 总线一种工作模式

API(Application Programming Interface): 应用程序接口

ARM(Advanced RISC Machine): 是一个 32 位元精简指令集（RISC）处理器架构

ATLAS: 在 CERN 的 LHC 对撞机上进行的粒子物理实验之一

BES (Beijing Spectrometer): 北京谱仪

BESIII: BES 三代

BLT(Block Transfer): 块传输，VME 总线传输方式的一种

CAMAC: 总线的一种

CBLT(Chained Block Transfer): 链状块传输，VME 总线传输方式的一种

CCG(Central Clock Generator): 大亚湾实验的一种时钟插件

CERN(European Organization for Nuclear Research): 欧洲核子研究中心

CFB(Clock Fanout Board): 大亚湾实验的一种时钟插件

CMEM: ATLAS 开发的基于 Linux 操作系统的分配连续内存空间的软件工具

CMS(Compact Muon Solinoid Detector): 在 CERN 的 LHC 对撞机上进行的粒子物理实验之一

ConfDB(configuration database): DAQ 系统的配置数据库

CORBA (Common Object Request Broker Architecture): 公共对象请求代理体系结构

CP: CP 是粒子物理学中两个对称运算的乘积：C 是反粒子共轭运算，P 是宇称。在强相互作用和电磁作用中 CP 转化运算对整个物理系统不产生任何影响（CP 对称），但是在一定的弱相互作用中这个对称被微小地打破（CP 破坏）。

CPU(Central Processing Unit): 中央处理单元，即处理器，计算机体系中执行操作指令的单元，是计算机的核心组件

CVS(Concurrent Versions System): 并行版本系统，是一种开放源码的版本控制软件

DAC(Digital to Analog Converter): 数模转换器，在本文中特指数模转换器的输入数值

DAQ (Data Acquisition System): 数据获取系统

DB(database): DAQ 系统的数据库

DIM(Distributed Information Management System): CERN 开发的分布式信息管理系统

DMA(Direct Memory Access): 存储器直接访问，VME 总线传输方式的一种

EFD(EventFlow Distributer): DAQ 的重要组件，负责数据流的分发传递

EFIO: DAQ 系统内部数据传输使用的协议

EMC(Electro-Magnetic Calorimeter): 量能器，BESIII 探测器的一种

FEE(FrontEnd Electronics): PMT 电子学读出插件

FIFO(First In, First Out): 是一种先进先出的数据缓存器

GCC(GNU Compiler Collection): 是一个用于 linux 系统下编程的编译器

GNU(GNU's Not Unix): GNU 计划的目标是创建一套完全自由的操作系统

GPS(Global Positioning System): 全球定位系统

IGUI(Integrated Graphical User Interface): DAQ 系统的用户界面

I/O(input/output): 输入输出端口，即输入输出地址。每个设备都会有一个专用的 I/O 地址，用来处理自己的输入输出信息

IP (Internet Protocol): 网际协议，RFC791

IRQ(Interrupt Request): 中断请求

IS (Information Service): 信息服务，DAQ 系统普遍使用的信息发布组件

ISR(Interrupt Service Routine): 中断处理例程

KamLAND: 位于日本神岗的中微子实验，探测反应堆中微子

KEK(High Energy Accelerator Research Organization, Kō Enerugī Kasokuki Kenkyū Kikō): 日本高能物理实验机构

LBNL(Lawrence Berkeley National Laboratory): 美国劳伦斯伯克力国家实验室

LED(Light Emitting Diode): 发光二极管

LHC(Large Hadron Collider): CERN 建造的质子-质子对撞机

LTB(Local Trigger Board): 大亚湾实验的本地触发板，存在于每个 PMT 电子学读出机箱中

MIPS(Microprocessor without Interlocked Piped Stages): 一种 RISC 处理器

MO(mineral oil): 大亚湾实验中心探测器使用的矿物油

MRS(Message Reporting System): DAQ 的消息报告组件

MTB(Master Trigger Board): 大亚湾实验的全局触发板，每个站点有一块

NFS(Network File System): 网络文件系统

OKS(Object Kernel Support): DAQ 的重要组件，负责对象管理

PCI(Peripheral Component Interconnect): 一种由英特尔（Intel）公司 1991 年推出的用于定义局部总线的标准

PMG (Process Manager): DAQ 系统采用的分布式进程管理器

PMT(photomultiplier (tube)): 光电倍增管。大亚湾实验中用来收集信号的一种光学设备。

PowerPC(Performance Optimization With Enhanced RISC – Performance Computing): 有时也简写为 PPC，是一种精简指令集（RISC）架构的中央

处理器，由 Apple–IBM–Motorola 创建于 1991 年

RC(Run Control): DAQ 的运行控制组件

RDB(remote database): 远程数据库

RMW(Read-Modify-Write): VME 总线的一种工作模式

ROAK(Release On ACKnowledge): 中断撤销方式的一种

ROM(ReadOut Module): 大亚湾实验 RPC 电子学机箱中的数据读出插件（出现于第 2 章与第 7 章中）

ROM(ReadOut Module): 大亚湾实验数据格式中以插件为单位的结构（出现于第 5 章与附录 B 中）

ROOT: CERN 开发的数据分析软件框架

RORA(Release On Register Access): 中断撤销方式的一种

ROS(ReadOut System): DAQ 的前端读出系统

RPC(Resistive Plate Counter): 阻性板探测器，大亚湾实验中探测器的一种

RTM(RPC Trigger Module): 大亚湾实验 RPC 电子学机箱中的触发模块

SAB(Surface Assembly Building): 大亚湾实验中的地面装配大厅

SLAC(Stanford Linear Accelerator Center): 美国国家加速器实验室

SFO (Sub Farm Output): DAQ 核心组件，负责存储

SLC (Scientific Linux Cern): Cern 版的 SL，加入了其个性化的设置

SNO(The Sudbury Neutrino Observatory): 位于加拿大的地下中微子实验，探测太阳中微子

Super-K(Super-Kamiokande): 在日本进行的中微子实验，主要目的是观测太阳中微子与大气中微子

TCP (Transmission Control Protocol): 传输控制协议，RFC793

TDC(Time to Digital Converter): 时间数字转换器，本文中特指转换器的输出值

TDAQ (Trigger and Data Acquisition System): ATLAS 实验的触发判选和数据获取系统

VME (VersaModule Eurocard): 一种通用的计算机总线

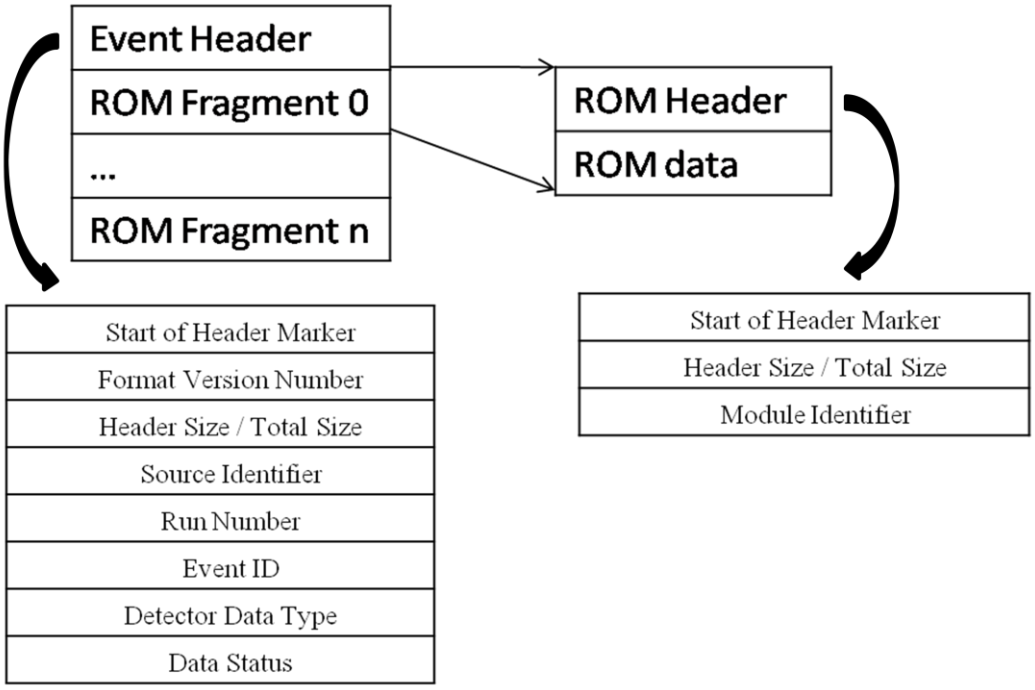
WP(WaterPool Detector): 水池探测器，即水切伦科夫探测器，大亚湾实验中探测器的一种

XML(Extensible Markup Language): 可扩展标记语言

附录B 大亚湾 DAQ 数据格式说明

B.1 基本格式组成

大亚湾 DAQ 系统采用的数据文件基本格式如图 B-1 所示。



B-1 DAQ 数据格式

B.2 分项格式说明

B.2.1 Start of Header Marker

每个数据段都是以 Start of Header Marker 作为开始标记的。这个标记值的不对称性可以用来分辨文件采用的字节序。

Fragment type	Header Marker
Event	0xaa1234aa
ROM	0xbb1234bb

B.2.2 Format Version Number

这里的版本号出现于 Event Header 中，是指数据格式的版本号。

Bit	31~24	23~16	15~0
	Major version number	Minor version number	reserved

B.2.3 Header Size and Total Size

Header Size 与 Total Size 都是以整型数目为单位的，并且 Total Size 中包含了 Header Size 的大小。

Bit	31~24	23~0
	Header size	Total size

B.2.4 Source Identifier

Event Header 中 Source Identifier 的结构如下所示，

Bit	31~30	29~24	23~16	15~0
			Detector ID	

其中 Detector ID 的结构分为 Site ID 和 Position ID 两部分

Bit	7~4	3~0
	Site ID	Position ID

Detector ID 的定义如下表

Site Name	Abbr.	ID
Test	TST	0
Daya Bay Near	DYB	1
LinAo Near	LAN	2
Far	FAR	3
Mid		4
Filling		5
SAB	SAB	6

Position ID 的具体定义如下表

Position		Position ID	Detector Name
-		0	TST
AD	1	1	AD1
	2	2	AD2
	3	3	AD3
	4	4	AD4
Inner WP		5	IWP
Outer WP		6	OWP
RPC		7	RPC

B.2.5 Detector Data Type

Detector Data Type 定义如下

Bit	31~28	27~24	23~16	15~0
	data type			Specific value

其中具体定义如下表

Data Type	Value(bit31~28)	Specific Value(bit15~0)
Normal	0	-
Calibration	1	-
FEEDiag	2	DAC Value

B.2.6 Data Status

Data Status 定义如下表

Bit	31~24	23~16	15~0
		Data Format Flag	Data Status Flag

其中 Data Format Flag 目前定义了两种 Flag 类型，如下表

Bit	Description
17	Reserve two words of CBLT header and footer
16	CBLT raw readout data, no ROM fragment

而 Data Status Flag 目前也定义了两种 Flag 类型，如下表

Bit	Description
1	Trigger number of modules don’t match
0	Invalid data

B.2.7 Module Identifier

ROM Header 中的 Module Identifier 定义如下表所示

Bit	31~30	29~24	23~16	15~8	7~0
			Detector ID	Module Type	Slot number

其中 Detector ID 的定义与 Event Header 的 Source Identifier 中相同。Module Type 的定义见下表。Slot Number 即为插件在 VME 机箱中的插槽号。

Module Type	ID
PMT FEE	1
FlashADC	2
LTB	3
RPC ROM	4
RPC RTM	5
MTB	6

附录C 大亚湾数据文件格式说明

C.1 数据文件的基本格式

大亚湾数据文件格式如下表所示：

File Start Record
File Name Strings
Metadata Strings (optional)
Run Parameters Record
Calibration Parameters Record (optional)
Data Seperator Record
Data
Etc…
Data Seperator Record
Data
File End Record

C.2 文件格式的具体实现

格式中每项的详细解释如下图。

Field	Type	Comment
file_start_record		
marker	u32	0x1234aaaa
record_size	u32	Size of this record .
version	u32	File format version.
file_number	u32	Number of the file in the run.
date	u32	When the file was open .
time	u32	When the file was open .
sizeLimit_dataBlocks	u32	Max file size in data blocks .
sizeLimit_MB	u32	Max file size in MB.
file_name_strings		
marker	u32	0x1234aabb
record_size	u32	Size of this record .
length	u32	Size of the following string.
appName	string	As in file name.
length	u32	Size of the following string.
fileNameTag	string	As in file name. From IS .

Field	Type	Comment
Metadata strings (optional)		
marker	u32	0x1234aabc
record_size	u32	Size of this record.
number	u32	Number of the free metadata strings.
length	u32	length of the following string.
Metadata string	string	(<tag>=value) from IS
run_parameters_record		
marker	u32	0x1234bbbb
record_size	u32	Size of this record.
run_number	u32	From IS.
rec_enable	u32	From IS.
trigger type	u32	From IS
detector_mask	u32	From IS.

Field	Type	Comment
calibration_parameters_record (optional)		
marker	u32	0x1234bbdd
record_size	u32	Size of this record .
DetectorID	u32	From IS
CalibrationAxis	u32	From IS
SourceID	u32	From IS
Zposition	u32	From IS
Duration	u32	From IS
LEDFreq		From IS
LTBMode	u32	From IS

Field	Type	Comment
data_separator_record		
marker	u32	0x1234cccc
record_size	u32	Size of this record.
data_block_number	u32	Number of the following block of data within this file sequence.
data_block_size	u32	Size of the following block of data in bytes.
file_end_record		
marker	u32	0x1234dddd
record_size	u32	Size of this record.
Date	u32	When the file was closed.
Time	u32	When the file was closed.
events_in_file	u32	In this file.
data_in_file	u32	In this file. In MB.
events_in_run	u32	In this run.
data_in_run	u32	In this run. In MB.
status	u32	Not zero if this file is the last.
end_marker	u32	0x1234eeee

(1). 文件格式中的每个结构都以一个整型标记位作为起始，紧接着便以一个整型来描述这段 record 的大小。

(2). File_start_record 和 file_end_record 结构中日期和时间分别占用一个整型的空间，以 DDMMYYYY 和 HHMMSS 格式描述，使用十进制编码。

(3). DAQ 组件可以通过配置文件来选择每个存盘文件的大小和每个文件中的事例数目（见 file_start_record），当实际运行时，一旦满足这两个条件之一，当前文件就会结束，并且打开下一个文件继续写盘。

(4). File_name_strings 中记录了用于产生文件名的字符串信息。为了保证数据的四字节对齐，可能会有额外的占位空间，而这些额外的空间并不被计算在 record_size 里。

(5). Run_parameters_record 中的 detector_mask 的定义见下表

Bit	Detector ID	Detector Name	Bit	Detector ID	Detector Name
0	-	-	12	-	-
1	0x11	DYB-AD1	13	0x25	LAN-WPI
2	0x12	DYB-AD2	14	0x26	LAN-WPO
3	-	-	15	0x27	LAN-RPC
4	-	-	16	-	-
5	0x15	DYB-WPI	17	0x31	FAR-AD1
6	0x16	DYB-WPO	18	0x32	FAR-AD2
7	0x17	DYB-RPC	19	0x33	FAR-AD3
8	-	-	20	0x34	FAR-AD4
9	0x21	LAN-AD1	21	0x35	FAR-WPI
10	0x22	LAN-AD2	22	0x36	FAR-WPO
11	-	-	23	0x37	FAR-RPC

C.3 文件命名规则

数据文件的格式必须采用以下的命名模式：

daq.<tag name>.<run number>.<stream type>.<stream name>.<application name>._<file number>.data

例如：

daq.Dryrun.0012345.Physics.SAB-AD1.SFO-1._001.data

daq.Dryrun.0012346.Calibration.SAB-AD2.SFO-1._001.data

daq.NoTag.0012347.Physics.DYB-AD1.SFO-1._001.data

其中，

- tag name 是由值班员在 DAQ 的 IGUI 用户界面上输入的，默认值为“NoTag”
- run number 是从 IS 获取的
- stream type 目前设定为运行模式的字符串，默认值为“ALLStreams”
- stream name 定义为探测器名的缩写，默认值为“ALLStreams”
- 空缺位全部用 0 填充

附录D 电子学配置数据库


```
<class name="Crate">
    <superclass name="HW_Object"/>
    <attribute name="NumberOfSlots" type="u16" init-value="20"/>
    <attribute name="Position" type="u16" init-value="0"/>
    <attribute name="LogicalId" type="u32" init-value="0"/>
    <attribute name="FEEDiagRunTimesPerDAC" type="u32" init-value="1000"/>
    <attribute name="PedeatalRunTimes" type="u32" init-value="10000"/>
    <attribute name="LTBfirmwareVersion" type="u32" init-value="0"/>
    <attribute name="FEEBoardVersion" type="u32" init-value="0"/>
    <attribute name="FEECPLDVersion" type="u32" init-value="0"/>
    <attribute name="FEEWaveform_enable" description="whether enable waveform
    readout" type="enum" range="disable,enable" init-value="disable" is-not-null="yes"/>
    <attribute name="FEEBaseline_enable" description="whether enable baseline
    readout" type="enum" range="disable,enable" init-value="disable" is-not-null="yes"/>
    <relationship name="Modules" class-type="Module" low-cc="zero" high-cc="many"
    is-composite="yes" is-exclusive="no" is-dependent="no"/>
    <relationship name="LTB" class-type="LTB" low-cc="zero" high-cc="many"
    is-composite="yes" is-exclusive="no" is-dependent="no"/>
    <relationship name="ControlHost" description="Points to a Computer where
    programs, communicating with crate modules (e.g. via VME) should/can be launched.
    Typically this is a zero Module in the Crate, but can also be a PC, connecting to a
    Crate by other interface." class-type="Computer" low-cc="zero" high-cc="one"
    is-composite="no" is-exclusive="no" is-dependent="no"/>
</class>

<class name="Module" description="Class Module describes a generic module.It has a lot
of subclasses.">
```

```
    <superclass name="HW_ConnectableObject"/>
    <attribute name="PhysAddress" type="u32" format="hex" init-value="0"/>
    <attribute name="Position" type="u16" init-value="0"/>
    <attribute name="Width" type="u16" init-value="1"/>
    <attribute name="LogicalId" type="u32" init-value="0"/>
    <attribute name="ModuleType" description="Modlue type unified ID of Daya Bay."
    type="u8" format="hex" init-value="0x1"/>
    <attribute name="ModuleEnable" description="module enable flag." type="bool"
    init-value="1"/>
    <attribute name="EnableCBLT" description="CBLT enable flag." type="bool"
    init-value="1"/>
    <relationship name="CPUs" class-type="Computer" low-cc="zero" high-cc="many"
    is-composite="yes" is-exclusive="yes" is-dependent="no"/>
</class>

<class name="FEE">
    <superclass name="Module"/>
    <attribute name="FEEBoardSerialNumber" type="u8" init-value="0"/>
    <attribute name="EnableTriggerDelay" description="Enable trigger signale delay in
    FPGA" type="bool" init-value="1"/>
    <attribute name="NHitOutputFormat" description="Set NHit output format. When 0
    output NHit; When 1 output looped number from 0~16; When 2 output value of
    NHitOutputTestPattern." type="u8" init-value="0" is-not-null="yes"/>
    <attribute name="NHitOutputTestPattern" description="The fixed value of NHit
    output for test." type="u8" init-value="0"/>
    <attribute name="FEEWorkMode" description="FEEWorkMode:Test Mode(1) OR
    Online Mode(0)" type="u8" init-value="0"/>
    <attribute name="ADCOutputDebugData" description="ADC(AD9222) outputs the
    debug data 0x800." type="bool" init-value="0"/>
```

```
    <attribute name="MaxHitPerTrigger" description="Set max number of hits per trigger." type="u16"/>
    <attribute name="TriggerDelay" type="u32" init-value="0"/>
    <relationship name="FEEDACThreshold" class-type="FEEDACThreshold" low-cc="zero" high-cc="many" is-composite="yes" is-exclusive="yes" is-dependent="no"/>
</class>

<class name="FEEDACThreshold">
    <attribute name="DACSelect" description="select DAC value : uniform value or different values" type="enum" range="UniformVal,DiffVal" init-value="UniformVal"/>
    <attribute name="DACUniformVal" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel0" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel1" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel2" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel3" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel4" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel5" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel6" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel7" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel8" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel9" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel10" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel11" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel12" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel13" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel14" type="u32" init-value="0"/>
    <attribute name="DACVal_Chanel15" type="u32" init-value="0"/>
```

```
</class>

<class name="LTB">
    <attribute name="PhysAddress" type="u32" format="hex" init-value="0"/>
    <attribute name="Position" type="u16" init-value="0"/>
    <attribute name="LogicalId" type="u32" init-value="0"/>
    <attribute name="ModuleType" description="Modlue type unified ID of Daya Bay."
    type="u8" format="hex" init-value="0x2"/>
    <attribute name="ModuleEnable" description="module enable flag." type="bool"
    init-value="1"/>
    <attribute name="EnableCBLT" description="CBLT enable flag." type="bool"
    init-value="1"/>
    <attribute name="timeStampSource" type="enum" range="internal,external"
    init-value="internal"/>
    <attribute name="ext_VETO_trigger_enable" type="bool" init-value="0"/>
    <attribute name="multi_cycle_HSUM_enable" type="bool" init-value="0"/>
    <attribute name="HNUM_SET_REG1" type="u32" format="hex" init-value="0x0"/>
    <attribute name="HNUM_SET_REG2" type="u32" format="hex" init-value="0x0"/>
    <attribute name="periodic_trig_setting" description="frequency of periodic trigger
    (xHz)" type="u32" format="hex" init-value="0x001200"/>
    <attribute name="manual_trigger_dly" description="delay after manual trigger
    activation" type="u32" format="hex" init-value="0x4c4b400"/>
    <attribute name="local_trigger_negative_width_reg" type="u32" format="hex"
    init-value="0x1000"/>
    <attribute name="cross_trigger_negative_width_reg" type="u32" format="hex"
    init-value="0x1000"/>
    <attribute name="ESUM_ADC_trigger_latency" type="u16" format="hex"
    init-value="0x10"/>
    <attribute name="ESUM_COMP_trigger_latency" type="u16" format="hex"
```

```
    init-value="0x10"/>
    <attribute name="Multiplicity_trigger_latency" type="u16" format="hex"
    init-value="0x10"/>
    <attribute name="cross_trigger_positive_width" type="u32" format="hex"
    init-value="0x0008"/>
    <attribute name="local_trigger_positive_width" type="u32" format="hex"
    init-value="0x0008"/>
    <attribute name="days" type="u32" format="hex" init-value="0x080325"/>
    <attribute name="time" type="u32" format="hex" init-value="0x230325"/>
    <attribute name="Analog_mux_address" type="u8" format="hex" init-value="0x0"/>
    <attribute name="internal_local_trigger_max_number" type="u8" format="hex"
    init-value="0x2"/>
    <attribute name="NHIT_link_test_result_sel" type="u8" format="hex"
    init-value="0x0"/>
    <attribute name="FEE_buffer_full_block_trigger_enable" type="bool"
    init-value="1"/>
    <attribute name="LTB_trigger_data_buffer_full_block_enable" type="bool"
    init-value="1"/>
    <attribute name="fee_data_ready_check_disable" type="bool" init-value="0"/>
    <attribute name="irq_latency" type="u8" format="hex" init-value="0x20"/>
    <attribute name="vme_interrupt_enable" type="bool" init-value="1"/>
    <relationship name="LTB_variableReg" class-type="LTB_variableReg"
    low-cc="zero" high-cc="many" is-composite="yes" is-exclusive="yes"
    is-dependent="no"/>
</class>

<class name="LTB_variableReg">
    <attribute name="LTB_triggerSource" description="LTB trigger source" type="u32"
    format="hex" init-value="0x0"/>
```

```
    <attribute name="DAC_total_value" type="u16" format="hex" init-value="0x080"/>
    <attribute name="DAC_half_value" type="u16" format="hex" init-value="0x0"/>
    <attribute name="ESUM_adc_trigger_threshold" type="u16" format="hex"
    init-value="0x010"/>
    <attribute name="HSUM_trigger_threshold" type="u16" format="hex"
    init-value="0x010"/>
    <attribute name="ESUM_comp_insig_filter_threshold" type="u8" format="hex"
    init-value="0x1"/>
</class>
```

发表文章目录

[1] Xiaolu Ji, Fei Li, Yongshun Ye, Mei Ye, Kejun Zhu 《Research and Design of DAQ System for Daya Bay Reactor Neutrino Experiment》, 2008 IEEE NUCLEAR SCIENCE SYMPOSIUM AND MEDICAL IMAGING CONFERENCE (2008 NSS/MIC)，VOLS 1-9

[2] 季筱璐，叶梅，朱科军，李小男，王贻芳 《基于 VME 总线的嵌入式 Linux 应用的研究》核电子学与探测技术 2010 年第 30 卷第 1 期

[3] 季筱璐，叶梅，朱科军，李小男，王贻芳 《大亚湾中微子实验 DAQ 读出子系统模型》2007 年第 13 届全国计算机、网络在现代科学技术领域的应用学术会议论文集

[4]季筱璐, 雷广坤, 朱科军 《BESⅢ量能器光纤监测系统的数据获取软件》2008 年第十四届全国核电子学与核探测技术学术年会论文集

[5] Li, Fei, Zhu, KeJun, Chen, LiPing, Chen, Mali, Ji, Xiaolu 《Online data processing and analyzing in BESIII DAQ》, 2009 16th IEEE-NPSS Real Time Conference - Conference Record

[6] 陈丽平,方建,朱科军,常劲帆,胡涛,李飞,季筱璐 《BESⅢ EMC 台阶模式的研究与实现》核电子学与探测技术 2009 年第 29 卷第 4 期

[7] 章红宇，盛华义，朱海涛，季筱璐，赵东旭 《BESⅢ主漂移室电子学测试系统软件》核电子学与探测技术 2007 年第 27 卷第 4 期

致谢

五年的博士生涯转眼即逝，回首走过的岁月，心中感慨良多。在本文即将结束之际，在这里由衷的感谢所有给予我指导、关心、帮助、支持的人。

衷心感谢我的导师王贻芳研究员。王老师以他严谨的治学态度、渊博的专业知识、孜孜不倦的敬业精神和诲人不倦的育人品德，给了我专业上的指引以及为人处事上的谆谆教诲。王老师对我的严格要求督促我在学习工作中不断进步。在此对他表示深深的谢意！

感谢朱科军研究员。在几年的学习、科研中，朱老师给予了我极大的指导和帮助。朱老师对待科研的认真严谨，是我工作学习的榜样。四年前，我在几乎陌生的领域中从零起步，正是在朱老师的指导和帮助下，才能获得今天的阶段性成果。在此，我向他表示最真挚的感谢。

感谢沈肖雁老师、李小男老师在博士五年期间给予我的帮助和关心。感谢叶梅对我工作的指导。感谢高能所实验物理中心所有给予我帮助的老师和同学。

特别感谢数据获取组的初元萍老师、张炳云老师、章红宇老师，感谢陈玛丽、庄建、雷广坤、刘英杰、王靓、朱海涛、陈丽平。尤其感谢李飞在整个课题期间给我的帮助。组里各位老师同学的帮助关心，我将铭记在心。

感谢在 BESIII 与大亚湾实验联调期间所有电子学组的合作人员，没有你们的帮助与配合，我的论文工作很难顺利开展。

感谢吴群、何苗、温良剑、王玲玉、姜巍巍在论文期间给予我的帮助。

感谢百忙之中抽空为我的论文审稿的各位老师。

感谢所有帮助过我的人，谢谢。

最后的感谢留给我的父母和亲人。感谢他们对我的养育之恩，感谢在这二十多年来的成长道路中，他们为我倾注的汗水和心血，感谢他们对我的支持，无论何时都是我最坚强的后盾。是他们的关心、理解和支持，才使我顺利地走到今天，对他们的感激唯以行动表达。