BESS 実験のための
新データ収集装置の開発

神戸大学大学院自然科学研究科物理学専攻

大和　一洋

要　旨

反陽子スペクトルの測定など、これまで様々な成果をあげてきた BESS 実験(Balloon-borne Experiment with a Superconducting Spectrometer)で、今後 2 つの大きな観測が計画されている。一つはニュートリノ振動の問題に関連し、大気ニュートリノ絶対流束計算に不可欠な TeV 領域までの一次宇宙線を精密観測する BESS-TeV 実験。もう一つは、一次反陽子の有無に決着をつける南極周回フライトである BESS-Polar 実験である。どちらの実験でも検出器の改良もさることながらその性能を発揮させるには、大容量の記録システムを用意することが不可欠である。本論文では、このどちらの計画にも対応できる収集装置、特に記録装置の開発について述べ、さらに BESS-Polar 実験で必須になるオンラインでの粒子同定システムについて、そのソフトウェアのの研究を行った。その結果を踏まえて、今後どのようなハードウェアを構築すべきかを議論した。

目　次

第 1 章　Introduction

1.1 BESS 実験とこれまでの成果

･宇宙線反陽子の観測

BESS 実験は、気球搭載型スペクトロメータを残留大気圧が約 5g/cm^2 の高度 36km まで打ち上げ、各種宇宙線を精密に測定することにより、宇宙における素粒子現象を研究することを目的としている。その測定器は、超伝導ソレノイド、高性能飛跡検出器、TOF カウンター、Aerogel Cerenkov カウンター、並列型高速データ収集システムより構成され、荷電粒子の運動量測定および質量による同定が可能である。

現在では、その観測対象は多岐にわたるが、とりわけ宇宙線反陽子の観測に力を注いできた。それまでの気球搭載型検出器に比べ 10 倍以上の大きなアクセプタンスを活かし、宇宙線中にごくわずか含まれる反陽子を 93 年に 4 例検出した。それ以降、TOF カウンターの改良や Aerogel Cerenkov Counter の導入・データ収集系の改良などにより、97 ～ 2000 年には毎年数百例以上の反陽子を捕らえていることに成功している。これにより、これまでの実験に対し圧倒的な精度で反陽子流束を求めるなど、重要な物理的成果を上げてきた[1][2][3]。

図 1.1 は、これまでの BESS 実験での反陽子スペクトルの観測結果と宇宙線伝搬モデルの予測を示す。通常、反陽子は一次宇宙線と星間物質との衝突により二次的に生成されると考えられているが、運動学的制限から 1GeV 以下の低いエネルギーの反陽子の生成は抑制される。このような衝突過程では、2GeV 付近に鋭いピークを持ち、低エネルギーへ向かって急速に減少していくと予言されている。BESS による観測は、まさにそのような結果を示しており、宇宙線伝播のモデルが大筋で正しいことを検証した。

また低エネルギー側に注目すると、統計誤差が大きいもののその観測結果は、理論予測よりも若干流束が大きいように見える。つまり、これは衝突により生成される二次的反陽子の他に、一次的な反陽子源の存在を示唆しているという解釈もあり得る。例えば、宇宙初期の相転移などの激しい攪乱によるごく短波長揺らぎから生じる PBH(Priomodial Black Hole)が Hawking radiation により蒸発する直前に反陽子を放出する可能性が指摘されてい

る[4]。また、ダークマターの候補の 1 つであるニュートラリーノが銀河ハローに蓄積し互いに衝突、対消滅して低エネルギーの反陽子を生成する可能性もある[5]。これらを起源とする反陽子のスペクトラムは、図 1.1 に併せて示したように衝突起源のものとは異なるので、衝突起源のフラックスが減少する低エネルギー側を精度よく観測することがこれらの探索で重要となる。しかし、現在の統計では一次起源反陽子の存在の是非を問うには、十分な精度といえない。今後より多くのデータを収集し、精密なスペクトルを得ることが重要であり、このため我々は 1.3 節で述べる BESS-Polar 実験の計画推進している。これが実現されれば、1 回のフライトで現在の 10 倍以上のデータを得ることが可能となる。

･宇宙線基礎データの測定

BESS 測定器は汎用の宇宙線観測装置であり、電荷が 1 および 2 の粒子は数十 GeV 程度ならば、一部のアイソトープを除いてスペクトラム測定が可能である。この特長を生かしこれまでに一次宇宙線の主成分である陽子・ヘリウムや地上や山頂におけるミューオンなどの絶対流束測定を行ってきた[6][7][8]。

最近 Super-Kamiokande で大気ミューオンニュートリノの天頂角分布異常が発見され、ニュートリノ振動の問題が注目されている[9]。大気ニュートリノの流束計算には大きな不定性があり、この実験ではその絶対値を用いずに ν_e/ν_μ の理論と実験値の比を元に解析を行っている[10]。大気ニュートリノは、一次宇宙線が大気と相互作用して生成されるので、そのスペクトルは一次宇宙線のスペクトルに比例する。その一次宇宙線のスペクトラムは、いろいろな実験で測定されているが、それらの結果には 50GeV で最大 2 倍の不一致があり、それが大気ニュートリノの流束計算の不定性の一因となっている[11][12][13][14][15][16]。

BESS 実験でもこのような問題に答えるべく、98 年の測定結果により 100GeV 付近までの陽子スペクトルを詳細に決定した(図 1.2)。大気で生成されるニュートリノは親となる一次宇宙線の約 1/10 のエネルギーをもつので、この 100GeV 付近までの精密な陽子スペクトラムは Super-Kamiokande での"fully-contained events"に相当する～ 10GeV のニュートリノ流束を計算するための重要なデータとなっている。また宇宙線と大気の相互作用の不確定性を小さくすること、とりわけ μ 粒子の生成・崩壊に関する正確なデータを用いることも計算精度の向上につながるので、99 年からは気球上昇中にもデータ収集を行っている。

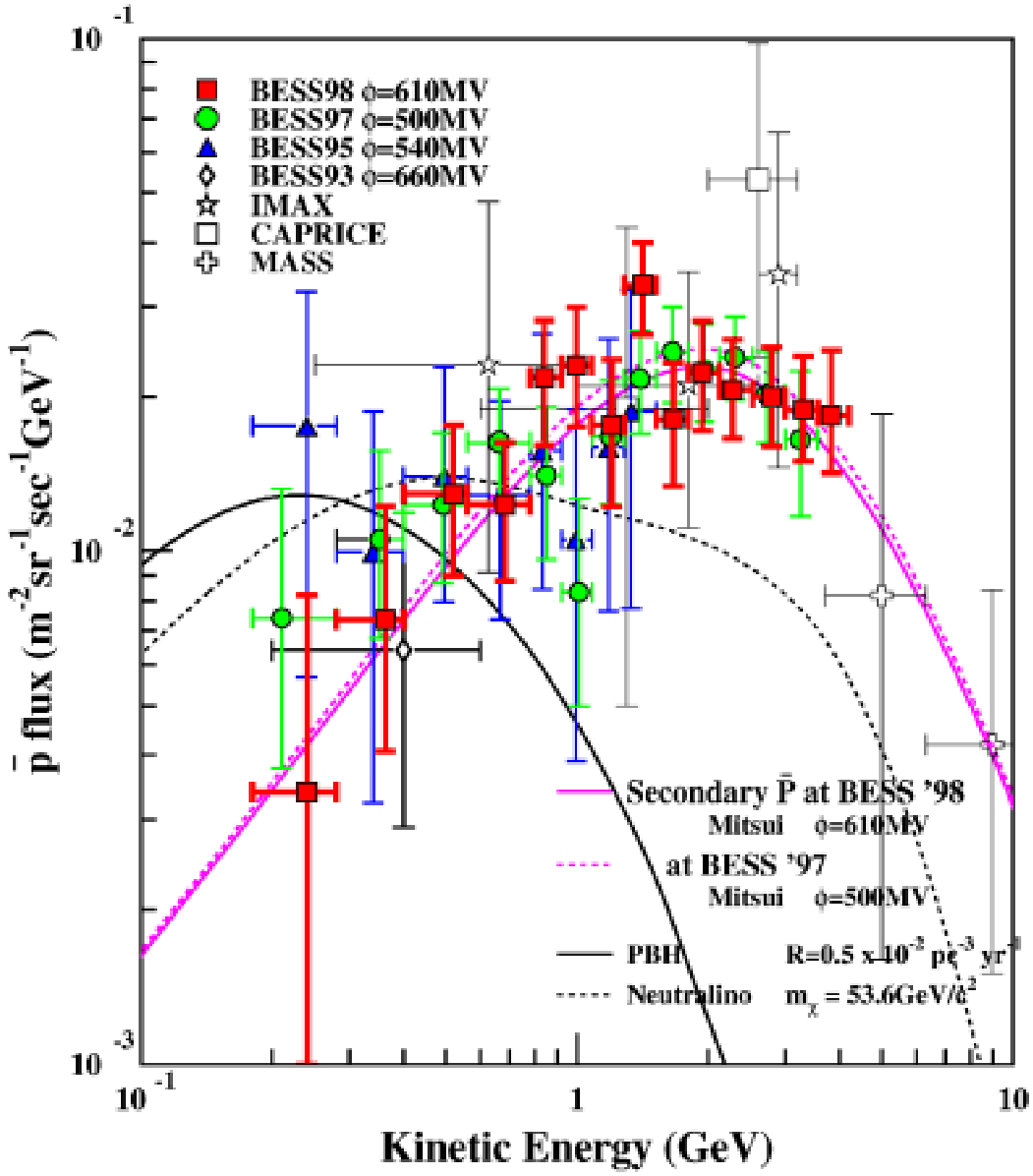


図 1.1 これまでのフライトで得られた反陽子スペクトル

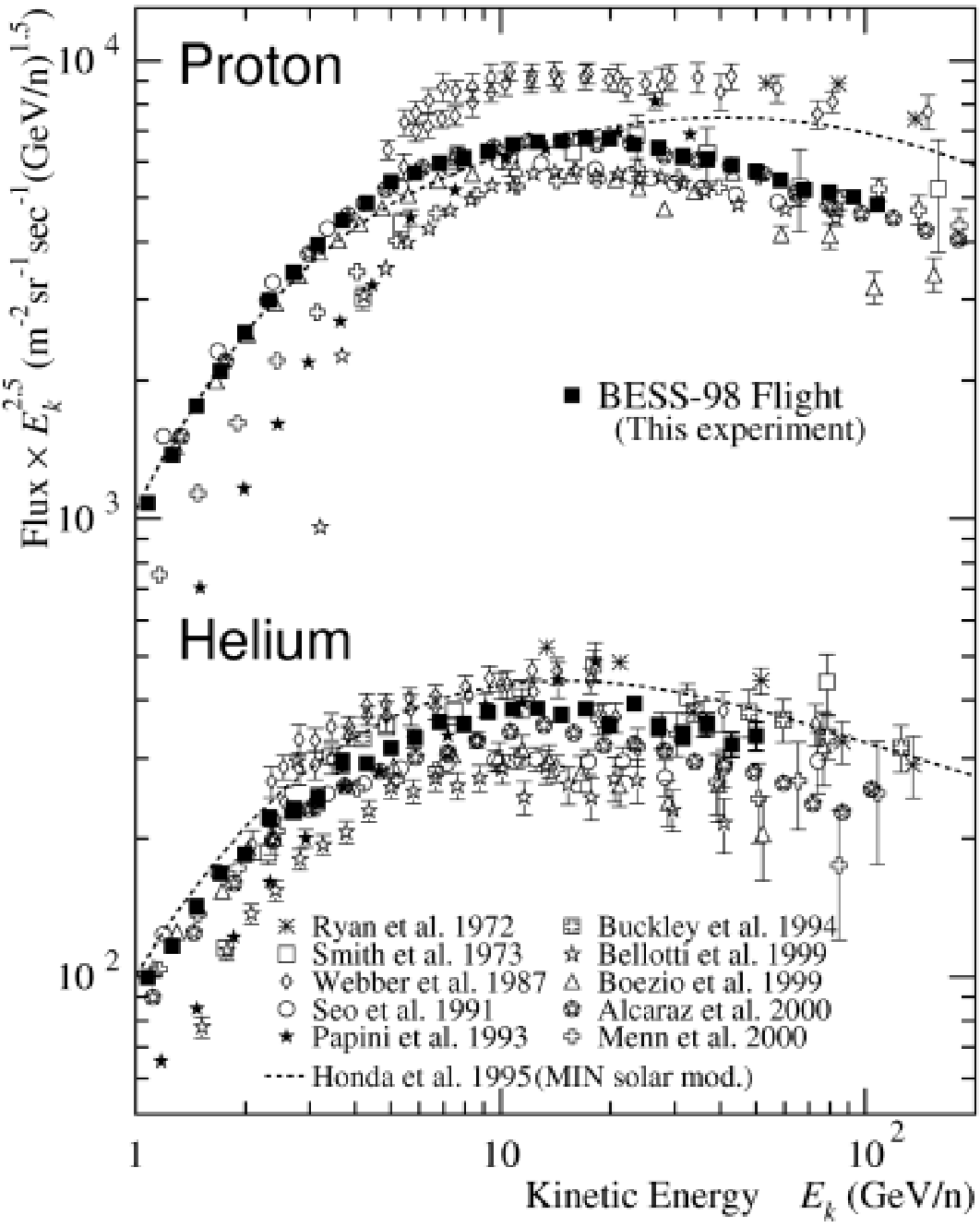


図 1.2 陽子、ヘリウムのスペクトラム

1.2 BESS TeV 計画

前述したように、これまで BESS 実験では、大気ニュートリノのスペクトルを計算する際に必要な一次宇宙線の絶対流束を提供してきた。今後 Super-Kamiokande で "partially-contained events"、"upward through-going muons"、"upward stopping muoons"に分類される～ 100GeV のニュートリノに対して、絶対流束を計算するためには、おおよそ 1TeV までの一次宇宙線のスペクトラムの精密なデータが不可欠である。

この領域は、カロリーメータを使った実験で測定されているが、宇宙線のエネルギースペクトラムは高エネルギー側へ行くにつれて急激に減少するため、エネルギーの絶対値測定には誤差が大きいこの方法では、スペクトラムの絶対値を求めるのは難しい。絶対値を測定するためにはスペクトロメーターがもっとも適しており、実績のある BESS 測定器に改良を施して 100GeV ～ 1TeV の宇宙線スペクトラムを詳細に測定する BESS-TeV 実験が計画された。これは 2001 年および 2002 年の夏にフライト予定されている。

よりエネルギーの高い宇宙線を計測するため、円筒形の測定器の外周を大きくし、そこへ新たに開発された ODC と呼ばれるドリフトチェンバーを搭載する(図 2.4)。また位置分解能を高めるために中央部の飛跡検出器もより高性能なものを新たに開発した(図 2.3)。これにより測定できる宇宙線のエネルギーは約 1TeV まで拡張され、図 1.3 に示したような精度の高い観測結果が得られると予想される。

これら新しいチェンバーの追加したことにより､読み出すワイヤーのチャンネルが増え、1 イベントあたりのデータサイズは数十%増大する。これまでの BESS 実験は、反陽子の検出を主目的としていたため、トラックトリガーと呼ばれるオンラインの粒子識別機構で宇宙線の 90%をしめる低いエネルギーの陽子の大部分を排除していた。それは、すべてのイベントを記録するには、記録装置の容量が 10 倍近く不足しているためである。したがって目的とする反陽子などの事象を中心に記録を行い、大量に降り注ぐ陽子はトラックトリガー自身の efficiency を求めるため、その選別に関わらず、数十回に一度はトリガーを発生しイベントを記録していた。しかし GeV ～ TeV にわたる広範囲で一次宇宙線を観測するこの実験では、このような選択を受けていない非バイアスデータも大量に観測したい｡記録できる容量が増えればこのような非選別トリガーのレートを増やすことができる。現在 BESS 測定器に搭載されている SCSI-1 規格のテープドライブは、記憶容量が 1 台あたり 20GB 程度であり、転送に必要な帯域も十分といえない。BESS 実験の様な気球実験ではスペースや重量の制限がつきまとうので、記憶容量を増やす場合にテープドライブの数を簡単に増やすことが難しい。したがって新たに開発された測定器の能力を十分に発揮させ、より精密なデータを収集するためには、データ記録系の改良が不可欠である。そこで、転送能力が 40MB/s である SCSI-3 規格のテープ装置を 2 台用いて、最高 120GB(非圧縮)まで記録ができるよう収集装置の開発を行った。この詳細を本論文の第 3 章で述べる。

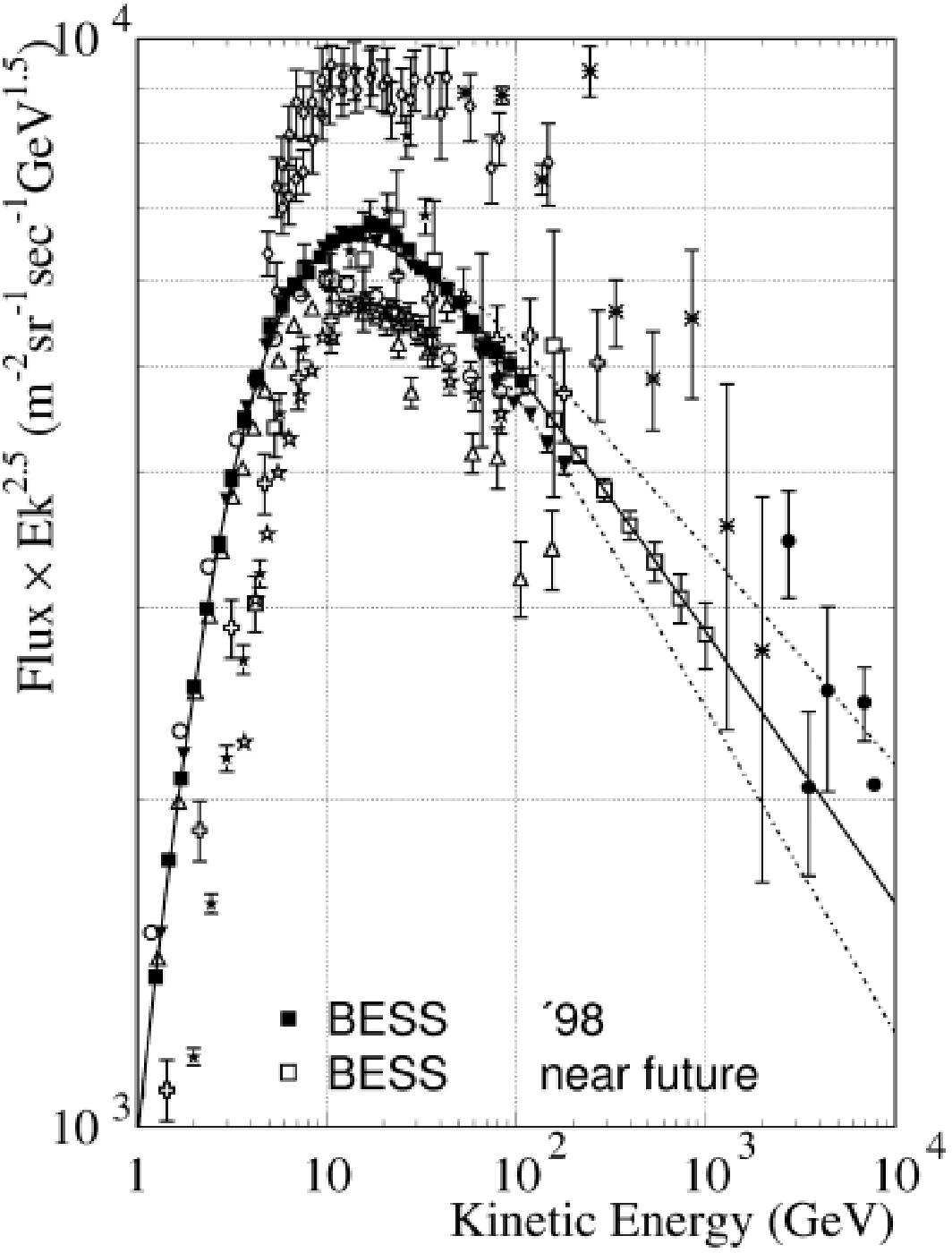


図 1.3 BESS-TeV で予測される陽子のスペクトラム

1.3 BESS-Polar 計画

1.1 節でもでふれたようにこれまでの BESS 実験では、低エネルギー領域の反陽子に注目し観測を行ってきた。その結果、低エネルギー領域において若干の過剰が示唆されているが、統計誤差のため、その原因を結論づけることが困難であった。これを検証するには、数十倍の統計を稼ぐ必要があり、これまでのようなカナダ北部における約 1 日の観測ではその達成は難しい。したがって、十分長い時間観測可能な別の方法が必要である。現在では宇宙線の観測手段として、スペースステーションや人工衛星などが考えられ、これらは数年以上宇宙空間に滞在することができるので魅力的である。また、BESS がカナダで行っているような気球実験の延長として、北欧から北米へ北極海の半周を数日かけるフライトや南極においてその周回を 10 日ないしは 20 日フライトする実験が NASA を中心にすでに行われている。

そこで、我々が目的とする反陽子の大量収集はどの様な方法がもっとも有利であるかが検討された。現在計画されているスペースステーション実験(AMS[17][18])と極軌道人工衛星による観測実験(PAMELA[19][20])で 3 年間で観測を行った場合、それに現在の BESS とほぼ同じアクセプタンスをもつ測定器が気球により南極で 20 日間フライとした場合の実質的な sensitivity を比較すると図 1.4 のようになる。AMS は大きな面積立体角をもつ測定器で数年間観測をするため、数 GeV 以上の宇宙線は BESS の 10 倍から 100 倍以上捕らえることができる。しかし、スペースステーションは赤道を中心として± 51.7 度の軌道をとるため、低エネルギー側へ行くほど地磁気によるカットオフの影響が顕著になり、PBH の蒸発が予測されているような数百 MeV の領域では、格段に感度を落としてしまう。また、極軌道衛星は、大型の観測装置が搭載できないため、数年間の観測でも統計を稼ぐことができない。さらに極上空を飛行する時間は、全体の約 1/3 であり、低エネルギー側で若干感度が悪くなる。ところが、南極における気球実験は、常にカットオフ rigidity が 100 ～ 300MV 程度の条件で観測できることと、ある程度大きな観測装置を用いることができるため、他の方法よりも低エネルギー反陽子の精密測定に有効であることがわかる。

また、一次起源反陽子と二次起源反陽子は、スペクトルの違いにより、太陽モジュレーションの大きさが異なる。したがって、太陽活動の異なる時期に 2 回測定することにより、さらに明確に一次起源反陽子の存在の有無を確認することができる。気球実験は毎年行うことができるので、この観点からも有利である。

このようなことから、次回の太陽活動の極小期にむかう 2004 年と極小期の 2007 年に南極において 10 ～ 20 日にわたる長期観測をおこなう BESS-Polar 実験が計画されている。しかしながら、その遂行にあたっては、現在の BESS 測定器に対して大幅な改良が必要とされる。第一には、測定器全体の小型化・軽量化であり、その対策として Pressure Vessel をマグネットのクライオスタットが兼用し、TOF カウンターなど検出器の一部を真空中

に配置する。さらに現在は、電源をすべて一次電池により得ているが、20 日にわたるフライトの全電力を同様にまかなおうとすると電池の重量だけで 1000kg を越えてしまい、このような方法をとるのは到底不可能である。これに対しては、ソーラーパネルによる電源装置が開発が進んでいる。

また、消費電力を大幅に減少させるには、データ収集系全体にわたる根本的な改造が必要であるが、まず長時間フライトで収集される莫大なデータを記録する方法を考えなければならない。データ量は単純に考えて現在の 20 倍になるため 1TB 近い記録媒体が必要となる。これに対して大容量の記録装置を用いるのはもちろんであるが、現在のところ、2～3 台のドライブでこの容量を満足する装置は存在しない。したがって、宇宙線の大部分を占める陽子をリアルタイムに同定し、選択的に記録する機構が必須がある。この際に、どのようなハードウェアの能力・構成が妥当であるかをそのアルゴリズムとともに研究した。これについて第 4 章で述べる。

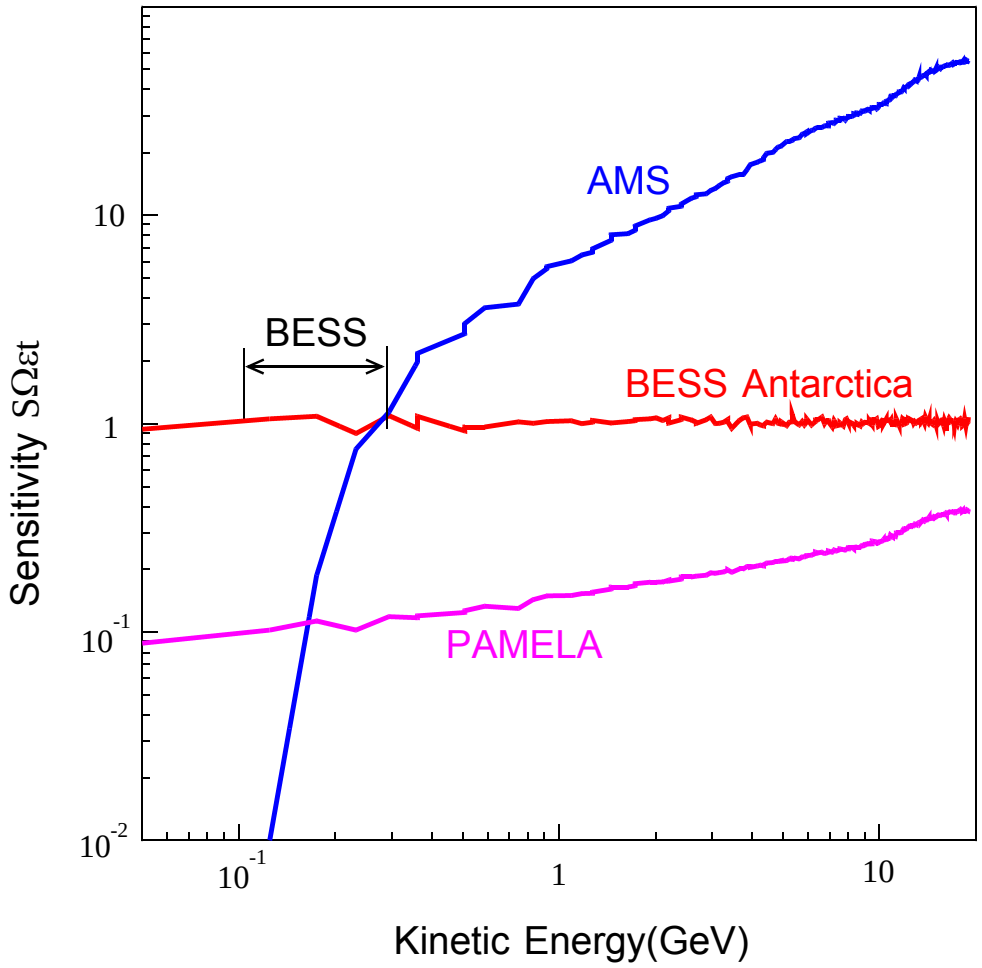


図 1.4 観測手段と sensitivity の評価

第 2 章　BESS 測定器

2.1 測定原理

BESS 測定器は、図 2.1 に示したようにもっとも外側に TOF カウンターが配置され、粒子の速度を測定する。また中心部に飛跡を測定する JET チェンバーを搭載している。超伝導ソレノイドの内部は 1T の磁場があるため、そこを通過した荷電粒子は、Rigidity（R = pc / Ze)に比例する曲率の円弧を描く。また、運動量 p と TOF カウンターで求められる速度比 β の間には、相対論での関係式 $\beta = p / (p^2c^2 + m^2c^4)^{1/2}$ が成り立ち、TOF と JET の dE/dx によって粒子の電荷の大きさ|Z|が分かるので、図 2.2 のように 1/β と Rigidity 平面にプロットすると質量 m をパラメータとしてバンドを形成する。運動量が大きいと磁場であまり曲げられないので、粒子のエネルギーが数 GeV を越えると識別は難しくなるが、スレッショルド型の Aerogel Cerenkov カウンター(屈折率 1.02)を搭載しているため、反陽子は 4.2GeV まで他のバックグラウンドとなる粒子(e^-, μ^- etc.)と識別することが可能である。

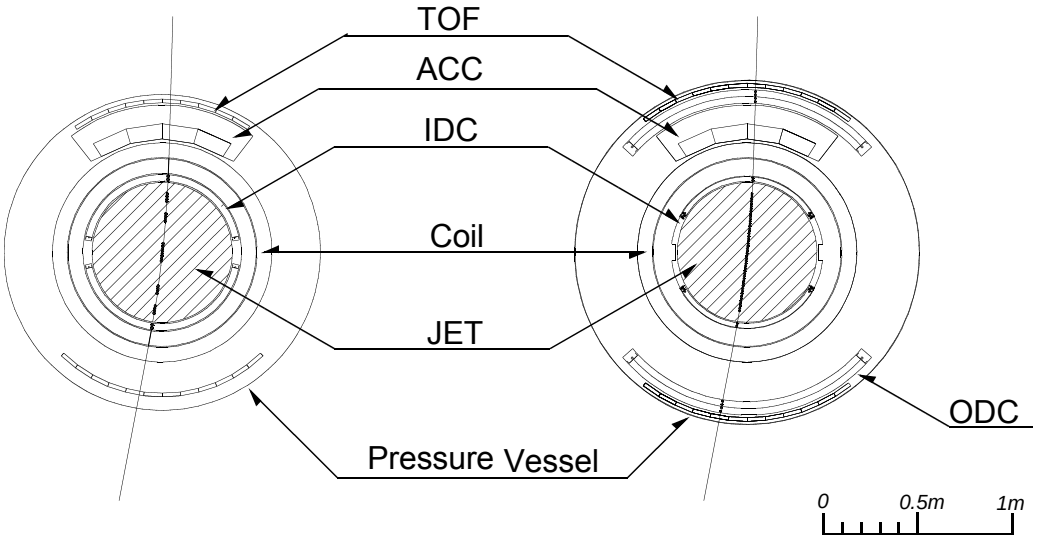


図 2.1 BESS 測定器(97 ～ 2000 年)(左)と BESS-TeV 用測定器(右)の断面図

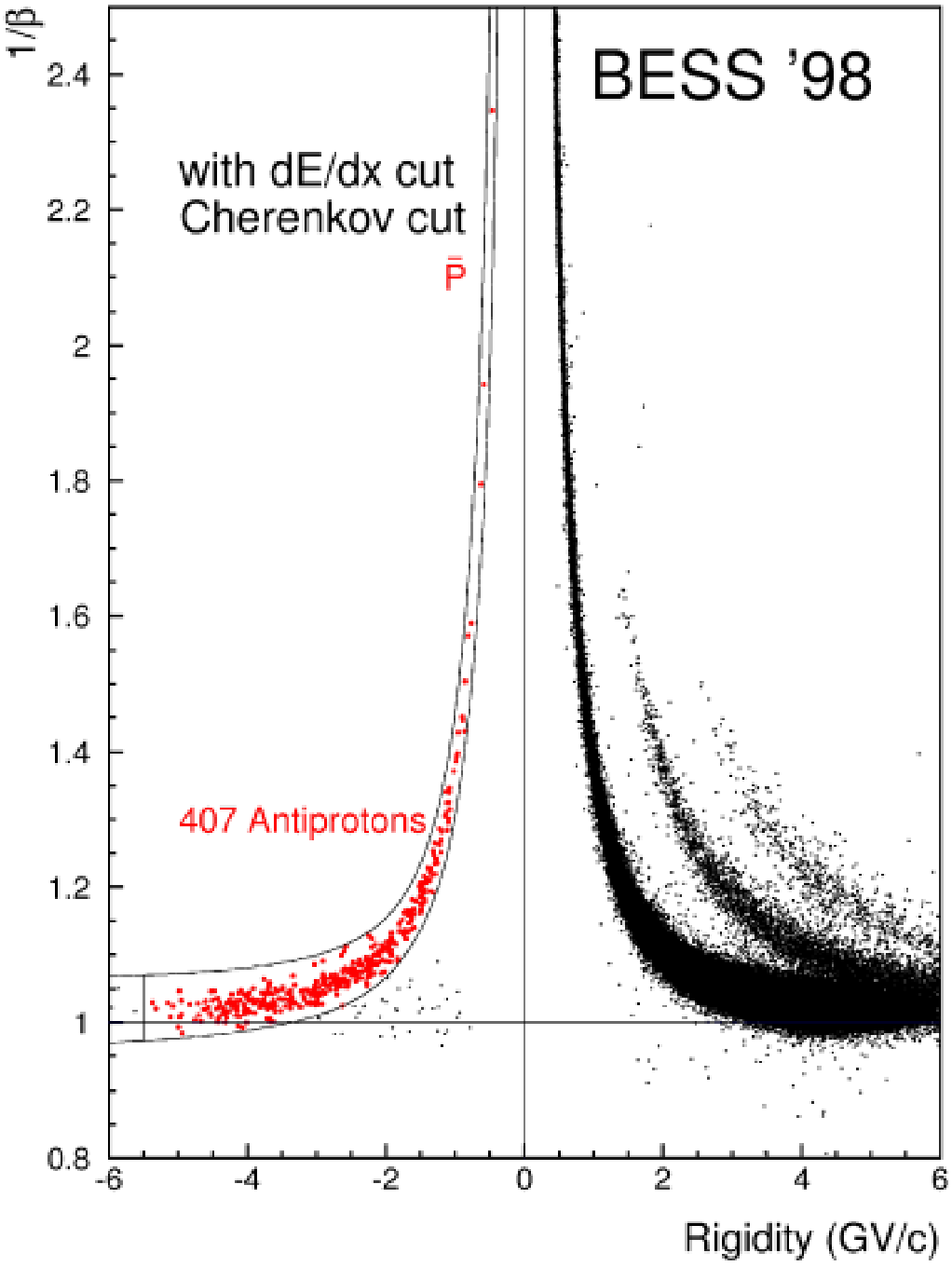


図 2.2 質量同定による反陽子の識別

2.2 BESS-TeV 測定器

BESS-TeV 測定器は、従来の BESS 測定器をもとにより高い運動量測定を目指し、改良が加えられた。一様な磁場中での運動量 p の分解能は以下の式で表される。

$$\frac{\delta p}{p^2} \sim \frac{1}{\sqrt{N+4}} \frac{\sigma^2}{L^2 B}$$

σ:測定点の分解能 *N*:測定点数 *L*:トラックの*path length* *B*:磁場

したがって、まずトラックの path length *L* を約 2 倍にするため Pressure Vessel の外周を大きくし、新たに ODC を開発・導入する。また位置分解能 σ を向上させるため、中央部の飛跡検出器である Jet Chamber/Inner Drift Chamber もより高性能なものが開発された。さらに測定点 N を増やすために、低消費電力タイプの FADC が開発され、これまでとほとんど同じ消費電力で 2 倍のチャンネルを読むことが可能になった。

これらの改良により $\delta p = p$ となる MDR(Max Detectable Rigidity)は 220GV から 1.5TV となる。実際に精度よく測定できる値はほぼ半分なので、測定可能な最大エネルギーは約 100GeV から約 700GeV に上がる。

2.2.1 BESS-TeV での検出器

･超伝導ソレノイド

粒子を曲げるための磁場は、従来の BESS に搭載されていた薄型ソレノイド超伝導電磁石をそのまま使用する。直径 0.8m、長さ 1m の空間に 1T の磁場を 15%の均一度で発生する。これは典型的な入射粒子の軌跡に対して運動量のずれが 2.5%以下である。しかし、これはオフラインで補正をする上、あまりにずれを大きい部分を通った粒子は解析に用いないのでほとんど問題はない。永久電流モードで動作することにより実験中電源の供給なしで磁場を発生し続けることが可能である。低エネルギー反陽子の測定のために物質量は片側 4.7g/cm^2(0.2 輻射長)と非常に薄く、入射粒子が反応を起こす確率が少なくなっている。

･ JET type drift chamber / Inner Drift Chamber

ソレノイドの内側には、円筒形 JET セル型ドリフトチェンバー(JET)とその外側に円弧状の Inner Drift Chamber(IDC)が配置されており(図 2.3)、ソレノイドの磁場で曲げられた粒子の軌跡を最大で 28 点測定し、そのヒットパターンから ridigity を求める。位置分解能は JET 175μm, IDC 220μm である。JET チェンバー単体では、トラックの曲率の分解能 Δ(1/R) = 0.005GV^{-1} であり、MDR(Max Detectable Rigidity)は 200GV であると見積もられる。

・Outer Drift Chamber

新たに開発・搭載されるドリフトチェンバー。TOF カウンターのすぐ内側に設置され、測定点の pass length を 2 倍にしているので、これだけでも全体の運動量分解能を 4 倍にすることが可能である。厚さ 70mm の領域に 4 層の sensitive　plane を円弧状に配置している(図 2.4)。ODC1 台分の物質量は、0.48g/cm^2 であり上下あわせて 1g/cm^2 以下ときわめて薄くできている。位置分解能は、約 150μm である。

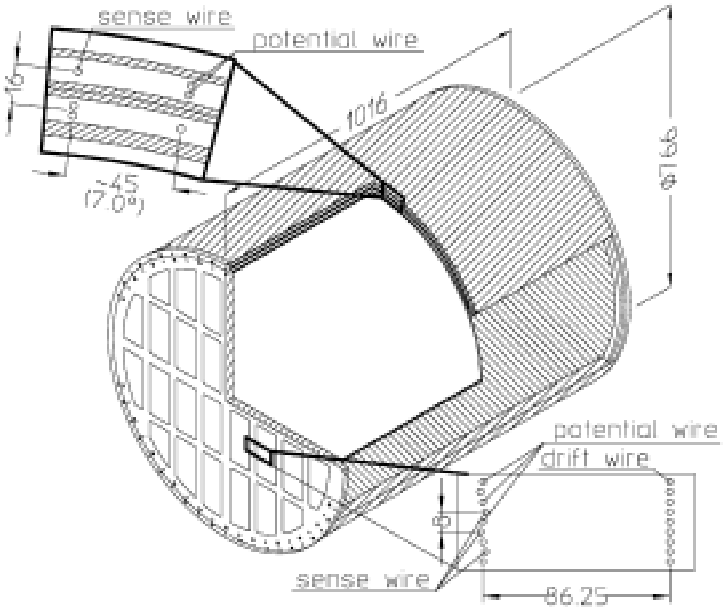


図 2.3 JET/IDC チェンバー

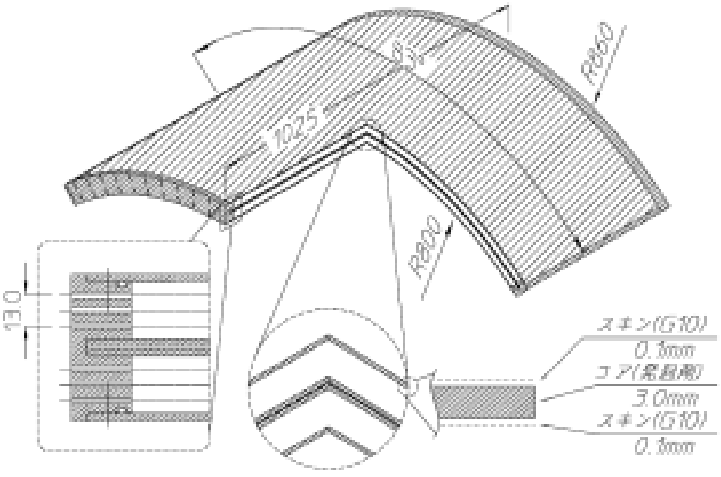


図 2.4 ODC チェンバー

・Time Of Flight Counter

入射粒子の飛行時間を測定するプラスチックシンチレーションカウンター。幅 100mm、長さ 950mm(light guide を併せて 1526mm)で、上部 10 本、下部 12 本が検出器最外部に設置されている。磁場中での使用となるため、光検出にはファインメッシュ型の光電子増倍管(PMT)を用いている。TOF の信号は入射粒子のエネルギー損失の測定や、BESS 測定器の 1st level triggger、2nd level トリガーにも使用される。これまでにシンチレータの材質、PMT の変更などの改良が重ねられ、カウンター単体の性能としては、$\sigma_{counter}$ = 50ps、TOF カウンターとしては、σ_{TOF} = 75ps の時間分解能を達成している。

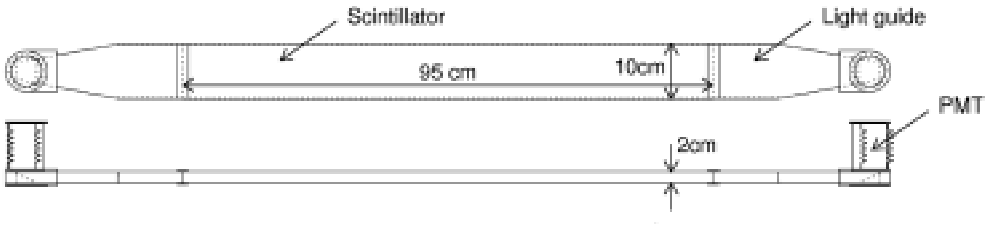


図 2.4 TOF カウンター

・Aerogel Cerenkov Counter

屈折率 1.02 のシリカエアロジェルを用いたスレッショルドタイプのチェレンコフカウンター。反陽子とそのバックグラウンドとなる電子、μ⁻とを区別する。厚さ 2cm のエアロジェルを 4 枚重ねてポリエチレンラップでラッピングされた 12 個のエアロジェルブロックがシンチレーション光を検出するファインメッシュ型 PMT とともに収納容器の中に納められており、これが Magnet の上部に設置される。

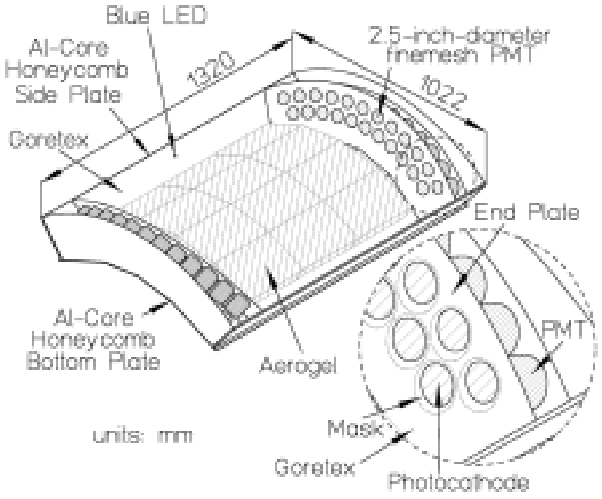


図 2.5 Aergel Cerenkov カウンター

2.2.2 BESS-TeV におけるデータ収集システム

・Central Intelligent Infomation Bridge (C-Brige)

地上から送られたコマンドをその文字列により対応する装置に転送したり、BESS 内部のモニター情報やイベントのサンプルデータを地上に送信する情報の一元管理装置である。これまでは、V40_COM、V40_MON、V50_DAQ、V50_CTS と名付けられた 4 台のコンピュータシステム(NEC 製 CPU V40 または V50 を使用)が Omni-net と呼ばれる Network を通じて通信し、適切にデータの配信を行っていた。これらは、それぞれ地上との交信、環境のモニター、DAQ 装置の制御、テープドライブの制御を専門に担当する。

しかし、BESS 実験が始まったの 1990 年前後においては、高性能であったらこれの CPU や Network System も、現在では関連 LSI の入手すら困難であり、メインテナンスもままならない。さらに当時は機能を分散させることで性能対する電力比をもっとも適当な構成にしていたが、現在ではその仕事を 1 つの低消費電力型 CPU が十分なし得る。したがって、Network のようにオーバーヘッドの大きい処理をなくして、1 台の CPU の内部でデータのルーティングを行った方が効率的である。そこで、組み込み機器向けの RISC CPU(三菱セミコンダクタ製 M32R)を用いて、これら 4 つのサブシステムに置き換わる集中情報制御装置を開発した。

この装置は、通称 C-Brigde と呼ばれ、多くの I/O を装備するのが特長である。また、60MIPS を越える整数演算能力に加え、CPU に 2MB(バス幅 128bit)の DRAM を内蔵している。CPU 単体では、これほどの高機能に関わらず 500mW と極めて低消費電力である。なお、BESS-TeV 実験の最初の年である 2001 年のフライトに搭載される予定である。

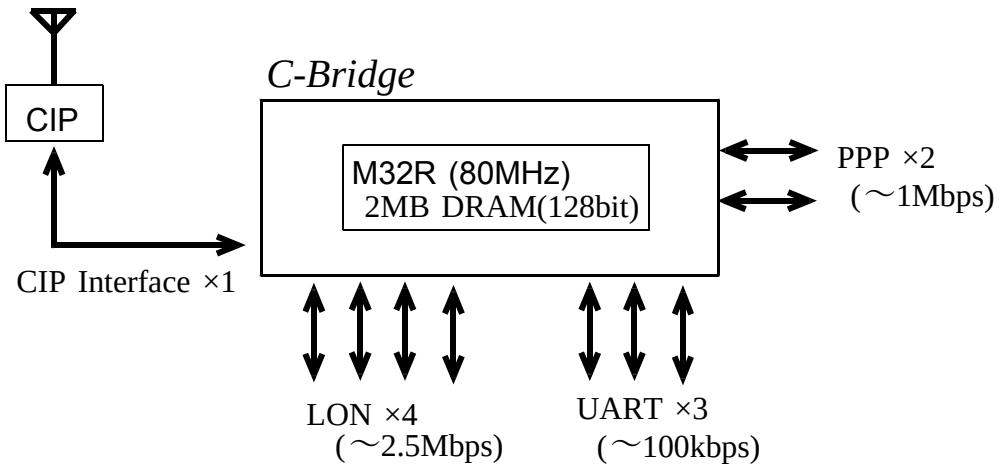


図 2.6 C-Brige の Input/Output Interface

・Low Power Flash ADC & Pre-Amp

BESS 全体の消費電力は約 1.2kW であるのに対し、現在の FADC の消費電力は 300W である。BESS-TeV 測定器ではで新たにチェンバーが追加され、読み出しチャンネルは 2 倍近い 1024 チャンネルとなる。現在のものを 2 台搭載するのは、電力・設置スペースの点で不可能であり、新たに低消費電力型の FADC が開発された。新しい FADC モジュールでは、アンプや ADC を厳選することで 1 チャンネルあたり 1/2 から 1/3 程度にした。

また、JET/IDC, ODC の 3 台のドリフトチェンバーを新たに開発するため、そこに取り付ける pre-amp も製作された。開発期間などを考慮し、Fermilab の CDF バーテックスチェンバーに使われていた pre-amp のベアチップを流用した。現在の BESS でつかわれている Fujitsu MB43458 の後継機であり、高ゲインと低消費電力、低ノイズ、軽量が特長である。電流-電圧ゲインの実測値は 15.7mV/μA であり、現行の MB43458 のゲインである 7.2mV/μA の 2 倍となっている。

・Event Aquisition System

BESS 測定器を粒子が通過したとき、上下の TOF カウンターのコインシデンスにより、初段の T0 トリガーが出力される。上空での T0 トリガーのレートは約 2kHz である。T0 トリガーが発生すると、トラックトリガーモジュールが TOF と IDC のヒットパターンから rigidity を算出する。その結果は、マスタートリガーモジュールに渡され、Aerogel での情報と併せて、その Event のデータを記録するかどうか判断される。そこで、棄却された場合、各測定モジュールに対してクリアー信号が入力され、T0 トリガーが発生する前の状態に戻る。

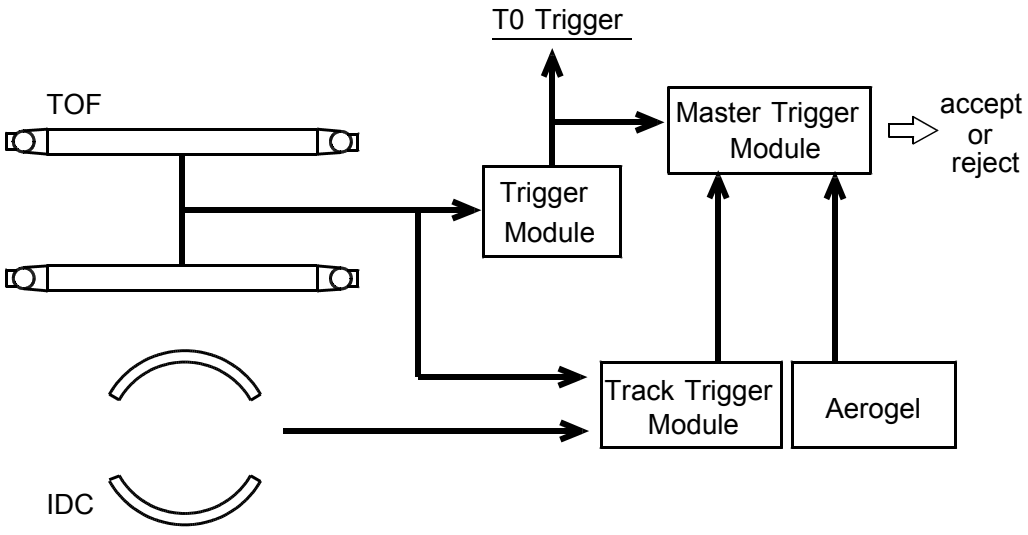


図 2.7 Trigger システム

データ収集が行われる場合、TOF, Aerogel のデータは CAMAC 上の ADC, TDC モジュールで測定され、JET/IDC, ODC のデータは、FADC により測定される。CAMAC および FADC でデジタイズされたイベント情報は、Transputer と呼ばれる並列処理型の Micro Processor を介してイベントビルダーに送られ、1 つイベントとしてパッケージングされる。この際、各モジュールからデータを読み出ししつつイベントビルダーに逐次転送が可能なこと、イベントビルダー内では DMA と Dual Port RAM を用いた完全なハードウェア・プロセッシングを行っていることにより、システム全体の dead time が 10%程度とたいへん小さくなっている。

さらに 15 台の Transupter を搭載した Transupter Bank とよばれる Unit でより高度で柔軟なイベントの取捨選択がリアルタイムで行われ、最終的に選択されたイベントのみが磁気テープに記録され保存される。

・Networking Environment Monitors

各部温度や圧力・磁場などを測定する汎用モニター(64ch)が 1 台。その他、H.V.専用モニター(56ch)が 3 台、Vessel 外部で残留大気圧や外気温を計るための小型 Unit が 1 台、GPS の出力する RS-232C Level の Digital 信号を解釈する Unit が 1 台、合計 6 台が搭載されている。これは本論文の著者が学部時代に気球実験に特化して製作したものであり、1ch あたりの消費電力が 10mW と非常に小さい。これらは互いに LON と呼ばれる Network で接続されており、情報の転送が容易に行える。モニター情報のほとんどは、一端 C-Bridge に送られたあと地上とテープ装置に転送される。また、バストポロジーで接続されているため、配線に無駄が少なく省スペースである。

2.3 BESS-Polar 測定器

BESS-Polar 用の測定器については、その大部分が開発中であるため、流動的な事項が非常に多い。したがって、今後その仕様が変更されることも十分あり得るが、現在、研究・開発が進められている部分を中心に述べておく。

BESS-Polar 実験においては、その目的から約 20 日にわたる長時間フライトが大前提であり、そのために様々な変更を余儀なくされる。まず、BESS 測定器の特長の 1 つである超伝導による励磁システムは、現在はヘリウムタンクの容量が数日分しかないので、南極での 20 間のフライトでも液体ヘリウムを供給できるよう新たに開発される。

また、南極での気球打ち上げ装置や回収用飛行機の制限から、測定器のサイズや重量は、従来の半分程度にしなければならない。これは、測定器全体のサイズを小さくしなければならず、従来の Pressure Vessel は、超伝導マグネットのクライオスタットがこの役目を兼ねる。測定器のもっとも外周に配置する TOF カウンターと ACC カウンターおよび、それらを読み出すための PMT、ADC/TDC は、クライオスタット外部の真空中に配置される予定である。

さらにこの長時間にわたるフライを 1 次電池でまかなうことは重量の点で不可能であり、電源としてソーラーパネルを利用した発電システムを開発する。しかし、この場合でも、供給できる電力は従来の 1/2 程度で、それは BESS に搭載されているデータ収集系をトリガーからストレージに至るまでの全系統について再検討し、もっとも合理的になるよう統合・撤廃を行わなくてはならないことを意味する。その概要は 2.3.3 節で述べる。そのほか、太陽フレア等により無線通信が妨害され、24 時間以上本体と地上基地で交信が断絶する可能性もあり、自律的に様々な問題を復旧できるよう設計する必要がある。

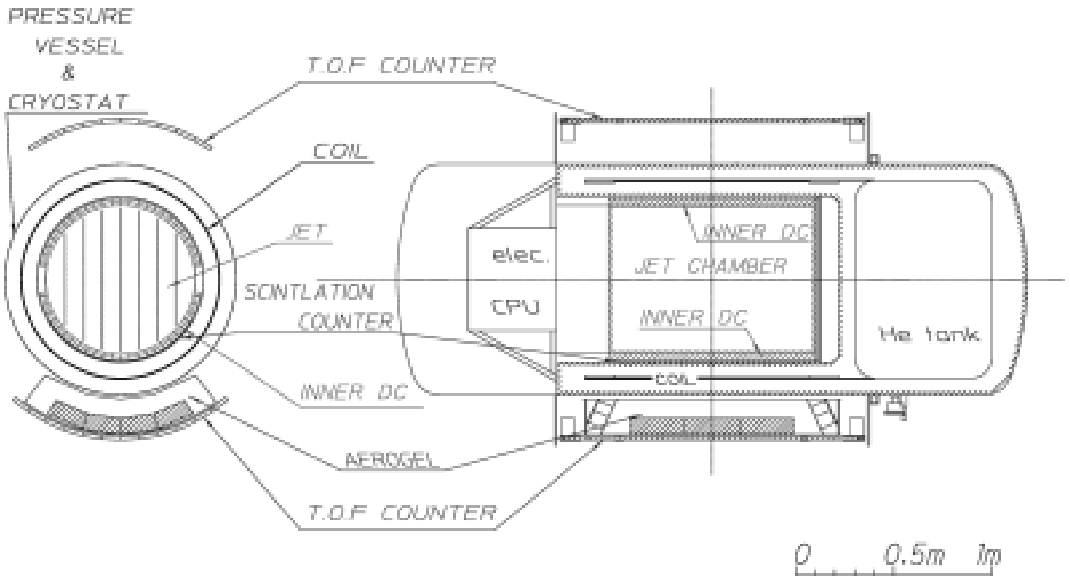


図 2.8 BESS-Polar 測定器

BESS-TeV で用いた以下の測定器については、南極でのフライト用に調整を施して BESS-Polar にも搭載する予定である。

・JET type drift chamber / Inner Drift Chamber

BESS-TeV で搭載される JET/IDC は BESS-Polar 実験での使用も考慮され、Polar 用に開発される超伝導ソレノイドの中にそのまま搭載できるサイズに設計されている。また Polar 用ソレノイドと JET/IDC をあわせたサイズ・重量は、南極での回収が可能な値になっている。

・Time Of Flight Counter / Aerogel Cerenkov Counter

現在搭載されている検出器をほぼそのまま使用する。しかしながら、これらの検出器の真空中(大気圧 5/cm^2)で使用に際し、特に PMT が圧力の低下に伴い、放電を起こしやすくなる。この問題に対策を施した PMT が開発されているので、カウンター自身とともに真空中での性能試験を行う。

・C-Bridge / Networking Environmet Monitors

これらは比較的最近導入されたものであり、BESS-Polar での使用も視野に入れ設計されている。たとえば、C-Brige の特長の 1 つは、多くの Serial/Parallel I/O や Protocol を実装していることであり、地上との交信装置等、今後たいていの仕様変更には対応できる。また、各モニターが結んでいる Network(LON)の構成は極めて柔軟に変更でき、必要に応じてモニター装置を追加・削除することが容易にできる。

2.3.1 太陽パネル発電システム

現在の BESS 測定器の実質の消費電力は 900W であり、レギュレータやケーブルでのドロップも含めると電池の消費は約 1200W になり、電池の重量は約 200kg である。南極でのフライト時間は 10 ～ 20 日にのびる一方でペイロードの重量は 1400kg 以内に押さえる必要がある。また、消費電力の大幅な削減も難しく、現在の 900W から 600W への削減を目標としているのが現実である。従って、重量制限の観点から一次電池による電力供給は非現実的であり、これらの条件を満たす一次電池として太陽電池を使用する。

太陽電池自身は、ソーラーカー用に製造されている民生品を用いる予定である。それらの多くは単結晶シリコンのセルを用いてあり、典型的な性能として 16%と高効率である。出力は 1m^2 あたり 160W で温度係数は、-0.4% / ℃である。

図 2.9 にソーラーパネルを展開した状態の測定器を示す。本体の下につり下がられているのが太陽電池パドルであり、太陽電池パネルの出力はスリップリングを通じて本体側の電力制御機器に送られ、負荷に供給される。パドル上には太陽電池パネルの駆動装置も配置され、太陽センサーにより追尾を行う。太陽電池のモニター情報や駆動装置へのコマンドは、無線およびスリップリングを通じて有線により、本体側に備えられているコントローラーへ送受信される。電力制御機器内には充放電制御回路があり、二次電池の充放電を制御する。

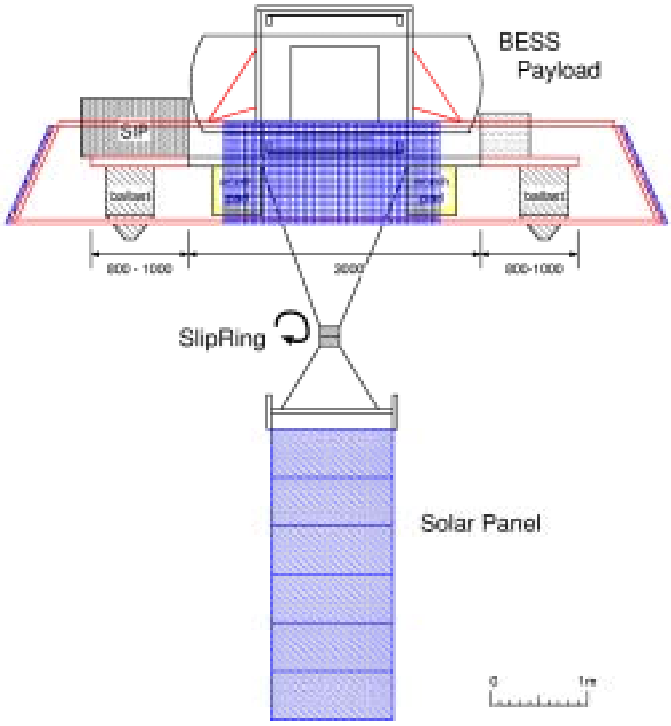


図 2.9 太陽電池パネル取り付けた BESS-Polar 測定器の外観

2.3.2 BESS-Polar 用超伝導ソレノイド

BESS-Polar 用の超伝導ソレノイドは、100MeV 近い反陽子を捕らえるため、片側 2g/cm^2 ときわめて薄い物質量で設計され、現在 KEK において開発が進められている。しかしながら、コイルは自分自身が発生する 0.8 〜 1T の磁場に耐える構造でなければならい。そのため、4.2K で 100MPa 以上の応力を持つアルミニウムによって強化された NbTi/Cu を線材とする。これを用いてこれまで 4 層構造であったコイルを 2 層にし、物質量の削減を行う。また 20 日にわたるフライトに対応するため液体ヘリウムのタンクも拡張され、内部に設置される飛跡検出器の圧力容器もクライオスタットが兼ねる。表 2.1 に現在のものと Polar 用に開発されているマグネットの各パラメータを示す。

表 2.1 現在と BESS-Polar 用ソレノイドコイルのデザインパラメータの比較

	Present		BESS-Polar	
	[Design]	[Operation]	[Design]	[Operation]
Cryostat Outer diameter	1.18m		1.02m	
Coil diameter	1m		0.88m	
Warm-borediameter	0.85m		0.78m	
Design magnetic field	1.2T	1.0T	1.2T	0.8T
Current	520A	433A	642A	430A
Turns(Layers)	3383turns (4layers)		2600turns (2layers)	
Stored energy	829kJ		600kJ	
E/M ration in coil	6kJ/kg		15kJ/kg	
Material@half-wall	4g/cm^2		2g/cm^2	
Radiation thickness	0.2Xo		0.1Xo	
LHe Capacity	150liter		400litter	
Over-all cryostat size	1.18m(φ)x2m		1.02m(φ)x2.6m	
Total weight	450kg		380kg	

2.3.3 統合型データ収集システム

・Trigger/TDC/ADC Module

TOF カウンターおよび ACC が真空中に配置されるのに伴い、これらを読み出す Discr/ADC/TDC も真空中に置かれる。またトリガーシステムは、上下 TOF のコインシデンスのみによる 1 レベルトリガーとする。上空では、イベントレートが約 2kHz と高レートであるが、各モジュールがトリガー信号でデータを収集し、すぐさま FIFO に収納して Busy 信号をクリアする。これにかかる時間は、40 〜 60μsec なので dead-time は数%にする事ができる。また、FIFO は数 Event 分(10 以下)の容量を持つだけで十分である。

FIFO に納められたデータは、Event Builder が出力する Ready 信号により調停されながらシリアル線でイベントビルダーに転送される。TOF および Aerogel のデータ量は 200 〜 400Byte 程度なので回線の速度は 1Mbps 以下で十分であり、配線長が 1 〜 2m であることを考えると極めた安定した通信が行えると考える。

従来、これらのモジュールは CAMAC 上に配置されており、スレッショルド等の設定は CAMAC バス経由で行われていた。しかし、CAMAC は低速な上に消費電力も大きいのでデジタイズの機能のみを抽出した独自のモジュールを作成する。各種コマンドやスレッショルドの設定は、電源制御・モニターで実績とノウハウのある LON 経由でおこなう。

・Low Power Flash ADC System

FADC は BESS-TeV 用に低消費電力なものが開発されたのでそれを流用する。この場合、64ch/枚の FADC ボード数枚とクレートコントローラから構成されるが、このクレートコントローラーの出力 I/F は Transuputer 用に設計されており、また FIFO を持ってないことから、そのまま使用することはできない。そこで、クレートコントローラのみ Polar 用に開発する。マスタートリガーモジュールおよび Event Builder との I/F は、上述の ADC/TDC とほとんど同じになるが、出力データサイズが約 2000Byte と若干大きいため、高速な回線を使用するか、パラレル通信にするなど多少の施策は必要である。

・Integarated Data Aquisiton System

BESS-Polar 用の Event Processing System は Compact PCI 規格のバス上に、Event Builder, SCSI カード、Online Analysis カードで構成される。磁場等の影響を避けるため別の小型圧力容器に納めされれる予定である。これは、最高転送能力が 100MB/s を越える PCI というの強力なバックボーンを通じて互いに高速な通信ができる上、ネットワークを使用するときのような消費電力の大きいトランシーバを必要としない。

まず、ADC / TDC / FADC のデータはイベントビルダーカードが受ける。これらのデータは、前述したように非同期に送られてくるので、パッケージングの際にマスタートリガーモジュールが割り当てたイベント番号をチェックする。何かの都合でこれらの不一致が起こった場合、マスタートリガーモジュールに対してリセットを要求し、ADC/TDC/FADC 内の FIFO を一端クリアしてからデータ収集を再開する。また、ある一定時間内に全チャンネルからのデータが転送されてこない場合もリセットを要求する。

ついで 1 パケットになったイベントデータは、まずリアルタイム粒子同定機構に送られ、Rigidity 等が計算される。その結果テープに書き込きが accept されたデータは、SCSI コントローラーへ送られ記録される。

BESS-TeV 用のデータ記録システムは Compact PCI 規格の CPU カードと SCSI カードおよび Transputer I/F カードで構成されていた。CPU カードには Intel 系 x86 プロセッサを用いたが、この理由は市販の CPU カードが出回っており、開発の手間が省けることや Linux 等の既存のソフトウェアが利用できるからである。しかしながら、熱的な問題も含め x86 プロセッサは多くの電力を必要とし、BESS-Polar においては必ずしも最適ではない。したがって、システム全体のに制御を行う CPU やリアルタイム粒子同定機構上のプロセッサは、低消費タイプの CPU/DSP を用いる予定である。その際必要な能力等については第 5 章で議論する。

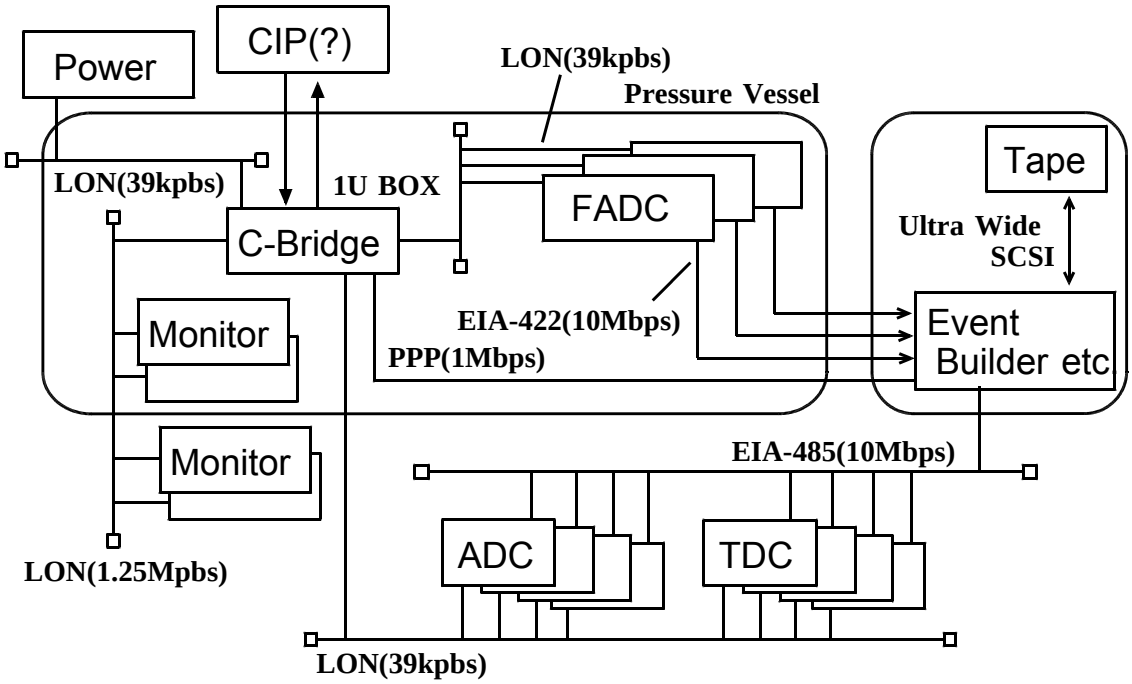


図 2.10 BESS-Polar におけるデータ収集システムの概要

第 3 章　高速･大容量データ記録装置の開発

3.1 新データ記録システムの概要

第 1 章で述べたように、今後の BESS 実験においては大容量の記録装置が必要とされる。BESS-TeV 実験ではチェンバーからの読み出しが増え、結果として単位時間あたりに処理するべきデータも増大するのに加え、非バイアスデータも今まで以上に収集したい。現在の BESS では SCSI テープドライブを用いてデータの蓄積を行っているが、そのコントローラは SCSI-1 規格であり、転送能力やドライブとの互換性の点で不安が多い。また、SCSI コントローラ・システムは、Pressure　Vessel 本体の中にあるが、テープドライブは Vessel 外部に設置された小型の防磁真空容器に納められている。その間にはコネクタが 2 カ所あり、50 本近い SCSI の信号線がそこを通過している。コネクタ・ケーブルとも対真空仕様のため、太く、重いので取り回しが著しく困難になっている。また次々に登場する大容量のテープドライブは SCSI-3 Wide LVDS 仕様である。ここで Wide はバスが 16bit であり、LVDS は信号が Low Voltage Differential Signaling であることを意味する。この規格は最高転送能力が 40MB/s 程度であり、我々の収集するデータのレート(～ 1MB/s)に対しても十分である。これらのドライブを使用するためには SCSI コントローラー側も対応してなければならないので、現在の BESS にそのまま接続することは不可能である。もし対応するコントローラを開発しても、Wide 規格の 68 本の信号線の取り回しが必要になる上、数十 MHz で動作する電線が何回もコネクタを通過するのはインピーダンスの整合性を考えても賢明な方法とは言えない。

また、BESS-Polar 実験においては 2 章で述べたようにイベントビルダーからストレージ装置までを一体化し、外部の小型真空容器に納める予定である。そこで SCSI コントローラー、テープドライブおよび制御用コンピュータの一体システムを開発し、Pressure Vessel と外部容器の間はシリアル信号で通信することは、上述の問題を解決するとともに、BESS-Polar 実験へ向けスムーズな移行が可能となるのでたいへん合理的である。そこでこのような記録システムの開発を行った。

この大容量記録システムを導入した BESS-TeV におけるイベントデータの流れは、図 3.1 のようになる。Transputer Bank を出たデータは、Transputer Link により記録システムの I/F ボード上に搭載された Transputer に転送される。ここでは、イベントデータが到着するとそれを知らせるためにメイン CPU に対してイントラプタ信号を発生する。それを受けてメイン CPU はイベントデータの読み込みを行い、テープドライブへ書き込む。

また、このメイン CPU に対してデータ収集の開始や停止等のコマンドは RS-232 経由で行われる。例えば、地上からデータ記録の開始命令が送信されるとコマンドの先頭に付加された記号から C-Bridge が本システムへのコマンドであると判断し、シリアルポートへコマンドを転送する。本システムの CPU はシリアルポートを常に監視しているので、コマンドが到着し次第、それを解釈し実行する。また、この小さい真空容器内にひとつのコンピュータシステムが構築されているので、その電源を制御したり、温度・圧力等をモニターするシステムは安全性の点で非常に有効である。そこで、Pressure Vessel 内部と同様に LON と呼ばれるネットワークシステムを用いてこれらの処理を行う。このシステムは拡張がきわめて簡単なことが特長で、ネットワークを外部真空容器まで延長して、その内部に新たに開発した電源の制御装置とモニターが行えるノードに接続した。これら制御情報の流れを図 3.2 に示す。

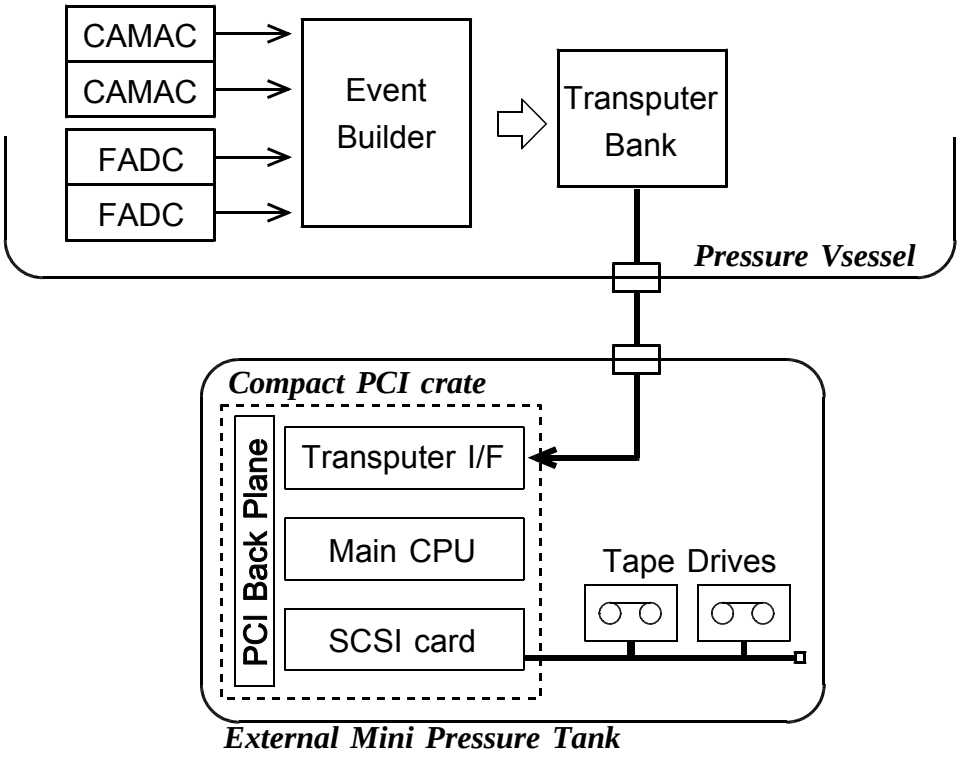


図 3.1 イベントデータの流れ

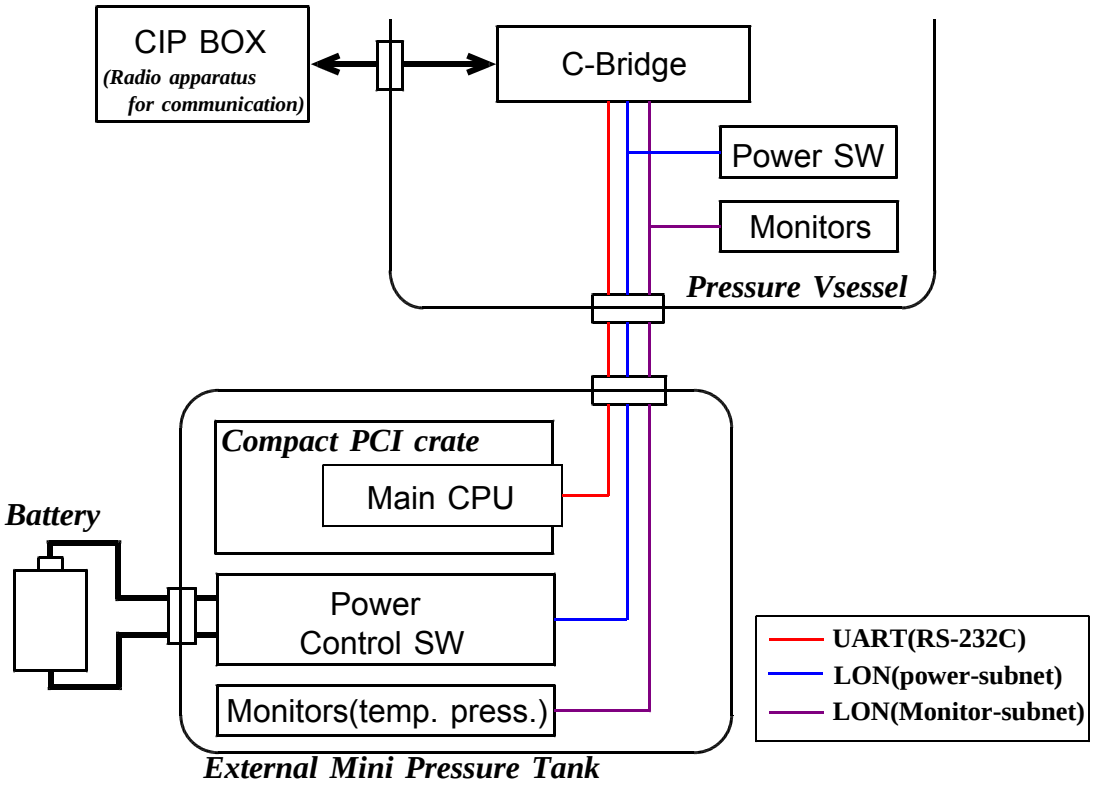


図 3.2 制御情報の流れ

3.2 Transputer-PCI Interface

Transputer が受けたデータをメイン CPU に転送するためには PCI バスを経由しなければならいが、今回使用した Transputer(T805)は、ネイティブな外部バスとして PCI をもっていない。したがって PCI バスに接続する何らかの方法を採らなくてはならない。ところで、インテル x86 系やモトローラ 68000 系をはじめとする大半の CPU は、外部メモリインターフェイスとして PCI バスを採用していない。また、PCI バスは 1 つのバスの上にコマンドとアドレスとデータをマルチプレクスするというこれまでにあまりない方式を採用している。したがって、少々の付加回路を付け加えるだけでは、一般的なバスと PCI バスを変換するのは困難である。そこで、このような一般的なローカルバスと PCI バスの変換のためのインターフェイス用 LSI が市販されている。いくつか選択肢はあるが、多くの製品にも搭載された実績をもつ PLX テクノロジー社の PCI 9050-1 を採用することにした。この LSI の特長をあげると、

・PCI Specification v2.1 に完全対応
・直接ローカルバスを制御可能
・PCI バスの動作クロックと非同期な 40MHz のローカルバスに対応
・ローカル側のバス幅をプログラマブルに設定可能(8, 16, 32bit)
・外部 EEPROM で初期状態を設定可能

図 3.3 に Transputer-PCI Interface のブロック図をしめす。中央部にある PLD は、Transputer とメイン CPU の読み書きを調停する役割を持つ。Interrupt 信号を発生させて、読み出しや書き込みのタイミングを知らせるほか、いくつかのレジスタを構成しており、FIFO に書き込まれた。イベント数や CPU の状態などを記憶しておくことができる。

このボードの動作状態は、(1)リセットステート、(2)コマンドステート、(3)Transputer ライトステート、(4)メイン CPU リードステートの 4 つから構成される。以下にそれぞれの状態における動作を説明する。

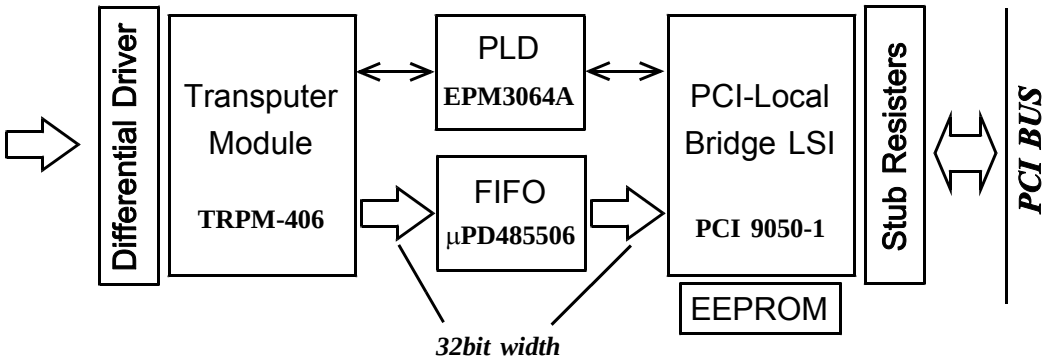


図 3.3 Transputer-PCI Interfece の構成

・リセットステート

パワーアップ時および Reset 信号が本 Interface ボードに入った時、Transputer および PLD はハードウェア的にリセット動作が行われる。PLD 内のレジスタや各種信号線の状態も初期状態となる。このとき、Transputer は待機状態となり、上流からデータが送られてきても何も行わない。

・コマンドステート

メイン CPU は、ボード上の Transuputer に命令を行うとき、PLD のコマンドレジスタに対しての書き込みを行う。書き込みサイクルの終了に伴って Transputer の Interrupt である Event Request 信号を PLD が生成する。Transputer は、これをトリガーとして、その内容を読み出だし、命令に対応する処理を行う。命令はリセット、動作開始、動作停止、FIFO 書き込み許可がある。リセットはリセットステートに遷移し、Transputer のメモリにあるイベントデーもすべて破棄される。リセット状態では Transuputer にイベントデータが上流の Transputer Bank から送られてきても無視するが、動作開始命令を受け取るとデータが送られてきたときに FIFO への書き込みを行うようになる。動作停止命令は、イベントデータの書き込みを停止するが、リセットと違い Transputer のメモリー内にあるデータは保持する。FIFO 書き込み許可については次で述べる。

・Transputer ライトステート

Transputer はデータを受け取ると FIFO にデータを書き込む。この FIFO のサイズは 20KB であり、典型的なデータサイズは 2KB である。したがって 10 イベント程度はキューする事ができる。そこで Transputer が書き込んだサイズを計算しながら、イベントが上流から送られてくるたびにデータの書き込みを続ける。次のデータを書き込むと FIFO の容量を超える状態になると、Transputer は PLD のイベントステータスレジスタに FIFO 内のイベント数を書き込む。この動作が終了すると PLD がメイン CPU に対して Interrupt を発生し、メイン CPU リードステートに移行する。これ以降、イベントが届いても Transupter 自身のメモリーに保存しておき、FIFO への書き込みは行わない状態になる。

・メイン CPU リードステート

Interrupt を受け取ったメイン CPU は FIFO に書き込まれているイベントデータをすべて読み出し、最後にコマンドレジスタに書き込み許可を発効する。これにより Transputer は再び FIFO への書込が可能な状態(Transputer ライトステート)へ遷移し、サイズをチェッ

クしながら FIFO データへの書き込みを続ける。収集が終わるまでこのサイクルが繰り返される。

なお、上述の方法では、数イベントごとにまとめて転送を行っているが、これは CPU に対する負荷を低減するための工夫である。BESS-TeV 実験の場合、平均 2ms でデータが転送されてくるが、Interrput が一度おこるとメイン CPU 内ではレジスタの待避など、いくつかのオーバーヘッドの大きい処理が行われる。今回採用した Linux システムで 2ms に一度 Interrput を発生させるのは、全体のパフォーマンスを考えるといささか無駄が多い。そこで、このように数イベントまとめて転送するバッファリング方式を採用した。また FIFO のサイズは 20KB とそれほど大きくないが、これには次のような２つの理由がある。１つは、大容量の FIFO は消費電力が大きいためである。たとえば、数 MB の FIFO は～500mA 程度の消費電力の製品が多いのに対し、今回採用した μPD485506 は～ 100mA である。もう 1 つは、Interrupt が起こったときにホストシステムは、データを読み終わるまで、その読み出しプロセスに占有されるためである。この時間があまりにも長いと他のアプリケーション(例えばテープにデータを書き込むプロセス)に悪影響を与える可能性がある。したがって、あまりに大容量なバッファを設けることは、その影響にに対するよけいな処理をしなくてはならないのでよりシンプルな方法を選択した。

3.3 制御用ソフトウェア

3.3.1 Interface ボード上の Transputer 制御プログラム

Transputer は並列処理型の CPU であり、4 本のシリアル Link を用いて複数の Transputer と情報を交換しながら動作することが可能である。そのため、ソースは occam といわれる並列処理記述のための言語によってかかれる。Transputer 自体の処理の流れは、前節で述べたとおりなので詳細は省くが、おおざっぱにはメイン CPU の調停を受けながらイベントの到着ごとに FIFO に書き込みを行うのが基本的な制御の方式である。図 3.4 にその状態遷移図を示す。また、イベントデータ書き込みの際、最初の 2 バイトにデータのサイズを挿入する。これはメイン CPU に対して FIFO から読み出すべきサイズを知らせるためである。

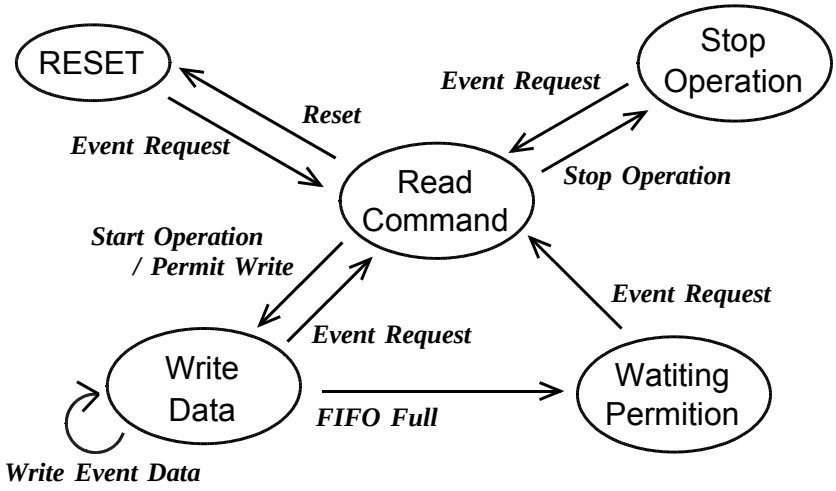
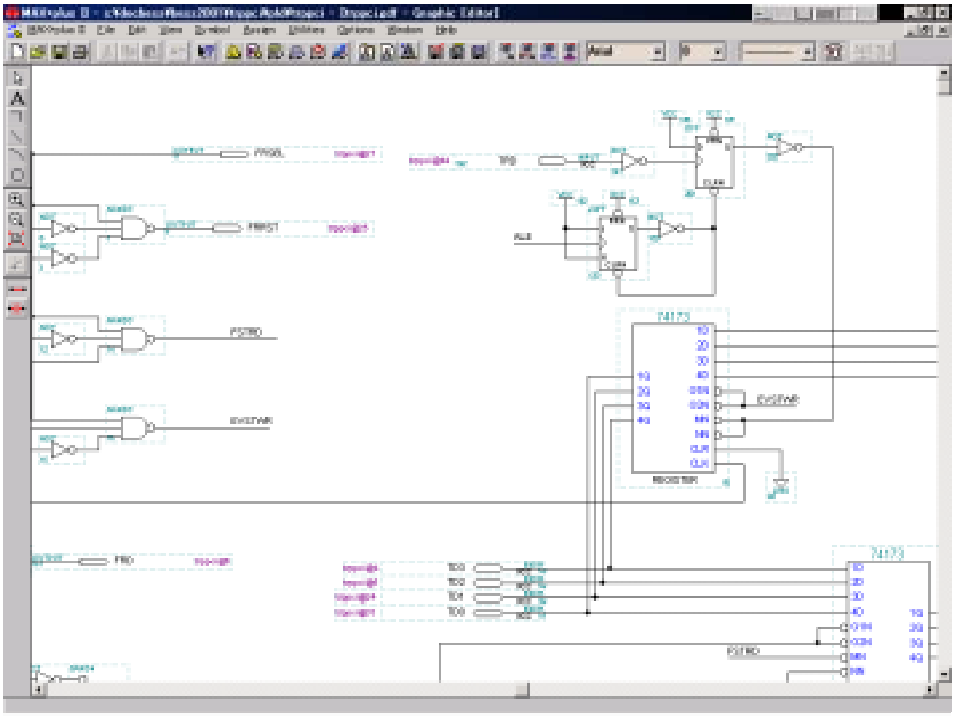


図 3.4 Transputer の状態遷移図

3.3.2 PLD の内部構築

PLD の内部構造の記述は、HDL や ABEL を用いて行われることが多いが、今回は、より手軽な回路図入力を採用した。これは内部に構築すべき回路が比較的簡単で、セル数をあまり気にしなくていいことや、極限まで速度を追求しなくてもよいためである。作業は専用のソフトウェアを用いて一般的な CAD のように画面上で回路図を構成してゆく(図3.5)。このソフトは今回使用した PLD のメーカの Altera より無償でダウンロードできる。

設計の流れは、まず、アプリケーションソフト上で回路図を入力する。このとき、各ピンの初期状態のロジックレベルや出力ピンのタイプ(トーテムポール・オープンドレイン等）も設定する。また、各信号と PLD の端子番号との対応など、必要なことはすべて画面上でのマウス操作と簡単な数値入力のみで行える。回路情報の入力が終わると、論理シュミレーションを行い、期待されたロジックが出力するかを確認する。ここで正常であることが確認できれば、実際にデバイスを指定し遅延配線を行う。ここで、もう一度シュミレーションを行い、各出力の遅延が許容範囲であることを確認する。これで設計はすべて完了し、あとはその情報をデバイスに書き込めばよい。今回用いた Altera の PLD (EPM3064A)は JTAG ピン 4 本による ISP(In System Programming)をサポートしており、PC のパラレルポートとこれらのピンを専用のケーブルで結ぶことにより書き込みが行える。専用の ROM ライターを用意しなくてもよい上、PLD が基板に実装されてからでも書き換えが行える。



3.5 回路図入力による PLD の設計

3.3.3 Interface ボードのデバイスドライバ

・コンフィグレーションレジスタ

今回、ホストシステムの OS には Linux を採用した。多くの OS がそうであるように Linux においてもハードウェアを制御する場合はデバイスドライバを書かなければならない。その前に PCI 用のデバイスドライバ作成に関して必要なコンフィグレーションレジスタについてふれておく。PCI 対応のボードは、システムの起動時に BIOS がその動作に必要な情報(I/O 空間・メモリ空間の大きさ、インタラプトの有無)をボード自身に問いかけ、他のボードと衝突しないよう適切に割り当てを行う。この結果はボードに搭載されているコンフィグレーションレジスタと呼ばれる 256 バイトのメモリーに格納され、後にそのボードを制御するプログラムに使用される。

コンフィグレーションレジスタのフォーマットを表 3.1 に示めす。ベンダー ID は PCI デバイスの製造メーカーを ID を格納するレジスタである。この値は PCI バスの規格制定を行っている PCI-SIG により割り当てられるコードであり、法人であれば取得が可能である。例えば、8086h はインテルとなっている。デバイス ID はデバイスの種類を示す ID で、ベンダによって自由に設定することができる。例えば、ベンダ ID が 8086h(インテル) で、さらにデバイス ID が 7180h のときは、Pentium Ⅱ用チップセット 440LX の PCI ホストコントローラであると判定できる。このベンダ ID とデバイス ID によってデバイスの種類を特定することができるので、PCI による Plug&Play でもっとも根幹をなすコードである。われわれは、このボードを販売するわけではないので、まだ割り当てが行われていないコードを使用すれば問題ない。本ボードでは、ベンダ ID(0817h)、デバイス ID(0204h)に設定した。

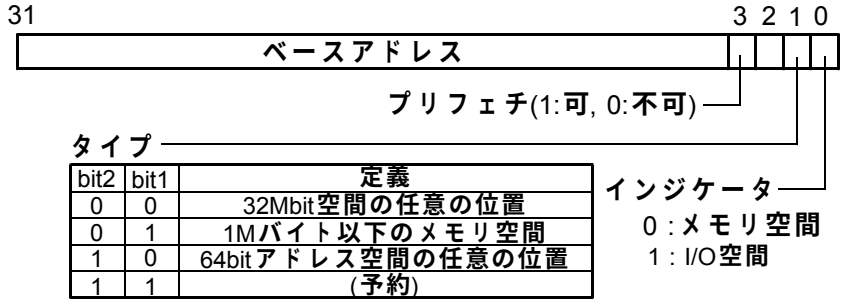
クラスコードレジスタは、PCI デバイス/ボードの主な分類を示すものであり、基本クラス、サブクラス、プログラムインターフェイスの 3 バイトから構成される。クラスコードは PCI-SIG により規定されている。最新の PCI バス仕様 Rev.2.2 におけるクラスコードの一部を表 3.2 に示す。このコードは、ボードの動作に本質に寄与しないが、互換性のあるボード等を識別するときには有効である。本ボードではデータ収集/信号処理コントローラが適当と判断し(11h, 80h 00h)に設定してある。

ベースアドレスレジスタは、その PCI デバイスに割り当てられた物理アドレスを保持するレジスタである。ベースアドレスレジスタは 0 ～ 5 まで最大 6 本あり、一般的には 0 から使用する。図 3.6 にベースアドレスレジスタのフォーマットを示す。ベースアドレスレジスタの再下位ビットは、そのベースアドレスレジスタがメモリ空間を要求しているか、I/O 空間を要求しているのかを示す。ベースアドレスレジスタを読み出したとき、このビットの値が"0"であればメモリ空間を、"1"であれば I/O 空間を要求している。

ベースアドレスレジスタに設定されたアドレスは、メモリサイクルや I/O サイクルのアドレスフェーズで出力される AD バスの値と比較され、比較結果が一致したときにそのボードに対してのアクセスであると判定される。ISA ボードなど従来の方法は、このアドレスが最初から固定されていたので、同じアドレスを持つボードを 2 枚さすと不具合が生じた。PCI ではこの部分が、レジスタで構成されているため、動的な変更が可能であり、システムの起動時などに他と重複しないように設定することができる。

インタラプトピンレジスタは PCI バスにある INTA#～ INTD#の 4 本のうち、どの割り込みラインを使うかを示すレジスタである。INTA#の場合は"1"、 INTB#は"2"となる。単機能デバイスでは INTA#を使うことになっているので、読み出し専用レジスタとして"1"を定義する。本ボードでも割り込みを使用するのでこの値は"1"となっている。インタラプトラインレジスタは、実際システムの IRQ の何番に対応しているかを示す。PC/AT アーキテクチャのハードウェア割り込みは、IRQ0 ～ IRQ15 までを使うので、PCI デバイスが出力した INTn#の割り込みが IRQ0 ～ IRQ15 のどの割り込み線に接続されたかを示す値が書き込まれている。

以上が、コンフィグレーションレジスタの概要であるが、これには表 3.1 で示したように読み込み専用レジスタと読み書き可能レジスタがある。読み込み専用レジスタは、ベンダ ID やデバイス ID のように常に固定した値を読み出され、このレジスタに書き込みが動作が行われてもその動作を無視するレジスタである。本ボードでは、コンフィグレーションレジスタは、PCI ブリッジ LSI である PCI-9050 の内部に存在する。この LSI は起動時に一度だけ外部の EEPROM のデータを読み込んで、これら読み出し専用レジスタの値を設定する。逆に読み書き可能レジスタとはリセット時にはゼロクリアされ、任意の値が書き込めるレジスタである。当然、書き込んだ値は同じレジスタをリードすれば読み出すことができる。ベースアドレスレジスタやインタラプトラインレジスタがこれらの代表であり、おもに BIOS によって書き込みが行われる。表 3.1 では読み書き可能レジスタは背景に網を付けてある。



bit2	bit1	定義
0	0	32Mbit空間の任意の位置
0	1	1Mバイト以下のメモリ空間
1	0	64bitアドレス空間の任意の位置
1	1	(予約)

図 3.6 ベースアドレスレジスタのフォーマット

表 3.1 コンフィグレーションレジスタ

デバイスID		ベンダID		00h
デバイスステータス		デバイス制御		04h
基本クラス	サブクラス	プログラムインターフェイス	リビジョンID	08h
セルフテスト	ヘッダータイプ	マスタレイテンシタイマ	キャッシュライン	0Ch
ベースアドレス0				10h
ベースアドレス1				14h
ベースアドレス2				18h
ベースアドレス3				1Ch
ベースアドレス4				20h
ベースアドレス5				24h
カードCISポインタ				28h
サブシステムID		サブシステムベンダID		2Ch
拡張ROMベースアドレス				30h
予約			CAP_PTR	34h
予約				38h
最大レイテンシ	最小レイテンシ	インタラプトピン	インタラプトライン	3Ch
デバイス固有レジスタ領域				40h ～ FFh

表 3.2 クラスコードの例

基本クラス	サブクラス	プログラムI/F	適用
01h 大容量記憶装置コントローラ	00h	00h	SCSI
	01h	xxh	IDE
	02h	00h	フロッピディスク
	03h	00h	IPI
	04h	00h	RAID
	80h	00h	その他
02h ネットワークコントローラ	00h	00h	イーサーネット
	01h	00h	トークンリング
	02h	00h	FDDI
	03h	00h	ATM
	80h	00h	その他
07h シンプルな通信コントローラ	00h	00h	XT互換シリアルポート
		01h	16450互換シリアルポート
		02h	16550互換シリアルポート
	01h	00h	パラレルポート
		01h	双方向パラレルポート
		02h	ECP 1.X準拠パラレルポート
	80h	00h	その他
11h データ収集/信号処理	00h	00h	DPIOモジュール
	80h	00h	その他

･デバイスドライバ

Linux において PCI デバイスドライバは、初期化時にまず、ボードにアクセスするためのベースアドレスを取得しなければならい。これは上述したようにコンフィグレーションレジスタを参照すればよいのだが、コンフィグレーションレジスタそのものが、メモリ空間のどこにマッピングされているのか分からない。したがって直接参照することができないが、Linux カーネルの内部には接続されている PCI デバイスの一覧を記憶しいるテーブルがあるので、そのなかからデバイス ID とベンダ ID を頼りに一致するものを探す。このテーブルは、1 つの PCI デバイスの情報を保持する構造体 pci_dev のチェーンで構成される。この構造体は<linux/pci.h>で定義される。以下にその部分を抜粋する。

```
struct pci_dev {
  struct pci_bus  *bus;           /* bus this device is on */
  struct pci_dev  *sibling;       /* next device on this bus */
  struct pci_dev  *next;          /* chain of all devices */

  void            *sysdata;       /* hook for sys-specific extension */
  struct proc_dir_entry *procent; /* device entry in /proc/bus/pci */
  unsigned int    devfn;          /* encoded device & function index */
  unsigned short  vendor;
  unsigned short  device;
  unsigned int    class;          /* 3 bytes: (base,sub,prog-if) */
  unsigned int    hdr_type;       /* PCI header type */
  unsigned int    master : 1;     /* set if device is master capable */
  unsigned int    irq;            /* irq generated by this device */

  unsigned long   base_address[6];
  unsigned long   rom_address;
};
```

また、<linux/pci.h>のなかで pci_dev 型のポインタ、pci_device が定義されている。これがチェーン構造の始まりで、next ポインタに次の構造体のアドレスが格納されている。チェーンの最後は next ポインタが null である(図 3.7)。

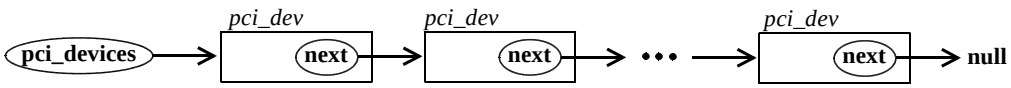


図 3.7 pci_dev の連鎖構造

構造体中の base_address[6]や irq は、そのデバイスのベースアドレスやインタラプトラインなどがすでに格納されているので、一致する構造体を発見すれば、その情報を用いて容易にアクセスができる。

通常、UNIX 系 OS において、アプリケーション・プログラムがでデバイスドライバとデータをやりとりする場合、read/write 関数を使用する。これは、数バイト～数十キロバイトのまとまった単位で転送を行うことができる。また、デバイスやデバイスドライバ自身を制御する時は ioctrl 関数により数バイトのデータのやりとりが可能である。さらに、select 関数をもちいて、デバイスにデータが使用可能かどうかをポーリングすることも多い。実際、次節で詳説するメインの制御プログラムは、数十 ms ごとにデバイスドライバに対してステータスポーリングを行い、デバイスドライバー内にデータが存在するとそれを読み込む。したがって、デバイスドライバはこれらの関数の呼び出しに対する振る舞いを記述する必要がある。

また、本ドライバが管理しておく情報は、保持しているイベントデータの数と各々のデータサイズである。これらは、デバイスドライバ内部に作られたリング構造のバッファで管理されている。リングバッファのサイズは 4MB に設定してあり、何かの都合データの読み出しが一時停止されても、数秒は保持しておくことができる。

本ドライバの主な動作シーケンスは、Interrupt によりボード上のイベントステータスレジスタを読み込んで、FIFO に格納されているイベントの数を記憶する。次に FIFO からデータを 2 バイト読み出しデータ長を得えて、その長さ分だけ FIFO からデータを読み出する。読み出されたデータは、自分自身のリング・バッファに格納される。それを詰まっていたイベント回数だけ繰り返して最後に FIFO 書き込み許可命令を発効する。

･ドライバの使用法

メインの制御プログラムのように、このデバイスドライバを使用するアプリケーションがデータを得たい場合、select 関数を使って、まずデータが存在しているかを確かめる。もしデータが存在していれば、"1"を返し、なければ"0"を返す。ついで ioctrl 関数により、データの長さを得る。そしてこの長さの分だけ read をすればよい。

3.3.4 メイン CPU の制御プログラム

メインの制御プログラムは、デバイスドライバからデータを読みとりテープドライブに書き込むのが大きな処理の流れである。そのほかに、モニターデータやコマンドのログなどもテープに記録する。これは、CPU ボードのシリアルポートから入力される。また、SCSI テープドライブの制御には"st"と呼ばれる Linux 用のドライバを用いる。第 3 者の作成し

たフリーウェアであるが、世界中の多くのシステムで用いられており信頼性も十分といえよう。図 3.8 にこのプログラムのフローチャートを示す。

まず、初期化では、Transputer-PCI Interface ドライバ、"st"(テープ)ドライバ、シリアルポート、ネットワークポートのオープンを行う。このとき、ioctrl 関数を使ってデバイスドライバに初期化命令を送る。これを受け取ったデバイスドライバは、自分自身の Buffer を初期化するとともにボード上の Transputer に対しても初期化を行う。

その後、シリアルポートあるいは、ネットワークを通じて"init"命令を受け取ると、デバイスドライバ経由で Transputer に動作開始を伝え、デバイスドライバまでイベントが届くようになる。本メインプログラムは、初期化が終了した段階で、すでにデバイスドライバをステータスポートリングを開始しているので、デバイスドライバにデータが到着すればそれを read 関数を使って読み込み、テープドライブへの書き込みを行うことができる。

また、データ収集中に不用意にコマンドを実行すると、システムのトラブルを招きかねない。そこで、本プログラムには、設定等の命令を受けつるコマンドモードとデータ収集を行うライトモードの 2 種類の状態がある。ライトモードでは、テープを巻き戻したり、イジェクトするなどの多くのコマンドが実行不可である。また、コマンドモードでがはデータを受け取っても、その内容はテープに書き込まれることなく破棄されるが、どのようなイベントレートであるかなどの統計情報はメモリに記憶される。これは、収集系のチェックのためにデータを収集するが、テープに書き込みたくないという時に便利である。

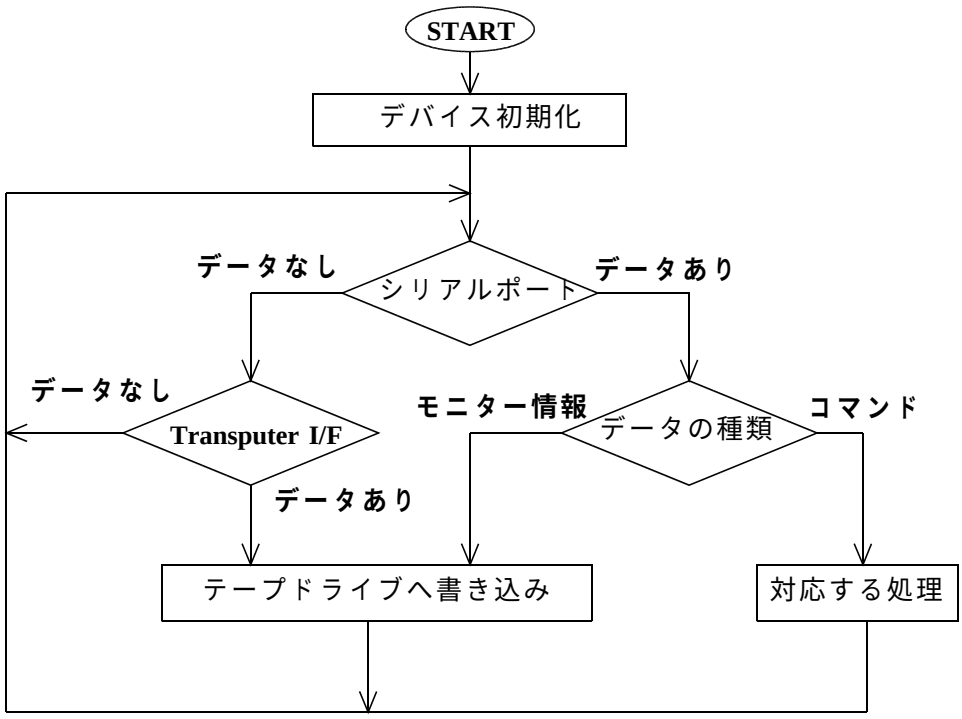


図 3.8 メインプログラムのフローチャート

3.3.5 各種設定

通常、Linux はデスクトップからサーバー用途まで幅広い要求に応えるため、さまざまな設定は汎用的な内容になっている。しかし、われわれはフライト中に印刷もしなければ、マウスやキーボードすら使う必要はない。したがって、実際の動作に不必要なデーモンなどは、起動しないように設定を変更した。本システム上で動作しているプログラムは、上記メイン制御プログラムと外部からシステムにアクセスするための sshd のみである。モジュールも Transputer I/F ドライバと SCSI ドライバ、"st"(テープ)ドライバ、それにネットワークドライバのみがロードされている。また、BESS 実験では、気球の打ち上げの際、強い衝撃が加わるので HDD の使用は控えたい。そこで今回は 512MB の IDE 互換のシリコンディスクを用いた。これだけの容量があれば、Linux システムをほとんどフルにインストールすることも可能であるが、今回は X Windows system はインストールせず、基本システムと gcc などの開発に必要な物のみで構成したので実際に使用している容量は、200MB 以下である。

シリコンディスクは基本的に書き込みも可能であるが、フライト中は、リードオンリーでマウントする。こうすることで、不意なリセットや電源のオフに対しても、内部のプログラム等が破壊されることを防ぐ。また、HDD より読み出しも早いので電源のオンからシステムのセットアップが終わるまでの時間も半分程度に短縮される。

3.4 性能評価

市販の CPU ボードや SCSI カード、ホストシステムとして Linux を採用するなど、本システムは開発期間を重視して行われた。そのため約半年で完成することができたが、第 3 者の設計した回路やソフトウェアが多く、開発者自身もすべてを把握しきれないといった側面もある。したがって、ここでは次の 2 項目についてその性能評価を行い、フライトでの使用にあたり、問題がないことを確かめた。

3.4.1 書き込みデータの正当性試験

正しく書き込みが行えている事を確認するため、フライト時のデータレート(～500kB/s)において書き込み試験を行った。上流のトランスピューターからの過去のフライトで得られたイベントデータを送りつづけそれを記録する。データを書き込まれたテープの内容は、送信したデータと比較され正しくデータ収集が行われていることを検査する。BESS 実験では、ペデスタルの収集のため約 1 時間を単位として、データ収集を行っている。したがって、このテストでも約 1 時間にわったて書き込みを行った。その結果は正常

であったが、これは最低条件であってさらに数倍の時間安定して動作するか検証する必要がある。本論文執筆時にはそこまで至らなかったが、フライトに向けて十分なテストを行うつもりである。

3.4.2 収集能力

TOF や JET チェンバーで検出された信号は、いくつかの経路を経て最終的に本ボード上の Transputer に到着する。その前段の Transputer-Bank までは、これまでの BESS に組み込まれ、実際にデータ収集を行った実績があるが、今回開発した最終段のシステムについては、地上でより注意深く性能試験を行う必要がある。とくに、上空では、データレートやデータサイズが地上を遙かにしのぐので、それに起因したトラブルがこれまでいくつかあった。このシステムでは全体のパフォーマンスをあげるため、各所にバッファをもうけてある。したがって、平均すれば上記のレートでデータが流れるが、短い時間の間には非常に高いレートにもなりうる。そこで通常の数倍以上のレートでの書き込みを行った。テープドライブの書き込み能力がの最高 10MB/s なので、それ以上の試験を行うことは無意味である。したがって 1 つのイベントのサイズを 2KB として 0.5 〜 8MB/s でテストをおこなった。その結果を図 3.9 にしめす。実際にテープに書き込まれる最高レートは 4.5MB/s 程度であることがわかる。これは、おそらくデバイスドライバ内のリングバッファがオーバーフローしたとき、新しく受け付けたデータを破棄するためと思われる。しかし、そのような状況でも正しくエラー処理が行われシステムがハングアップすることがないことを確かめられた。また、この 4.5MB/s という値は BESS-TeV 実験、BESS-Polar 実験においても十分余裕のある値である。

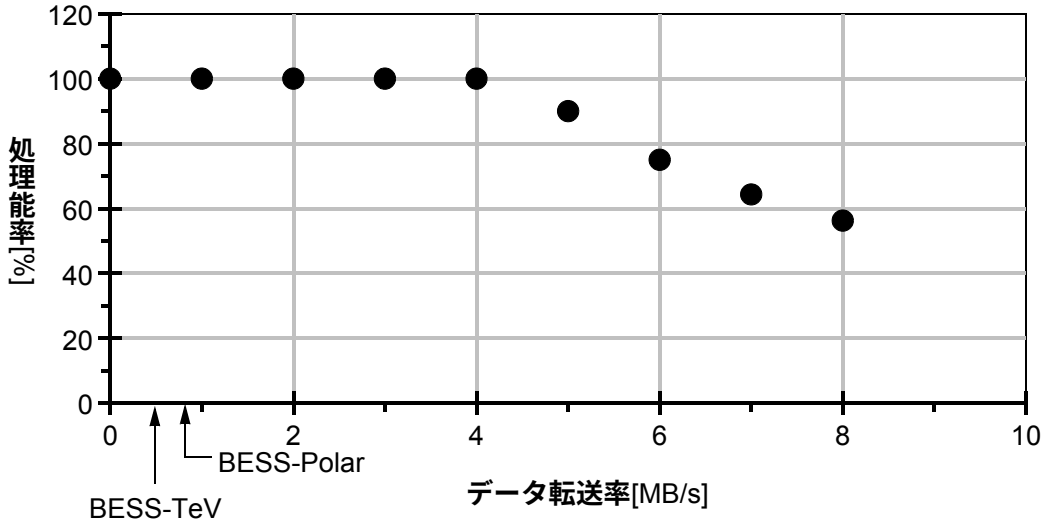


図 3.9 データ転送能力

第 4 章　BESS-Polar 実験での リアルタイム粒子同定

BESS-Polar 実験では、実験装置全体に渡って大幅な改良がなされるが、ここでは特にデータ収集システムにおいて、必須の機能であるオンラインでの粒子同定機構について議論する。

4.1 リアルタイム粒子同定機構

20 日間の南極周回フライトで収集されるデータサイズを大まかに計算すると以下のように約 7000GB となる。

$$20\text{day} \times 24\text{hrs} \times 3600\text{sec} \times 2\text{kB} \times 2\text{kHz} = 6912\text{GB}$$

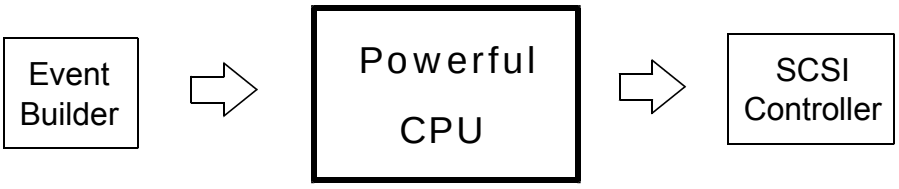
第 3 章で述べた収集システムは、SCSI-3 Wide LVD 規格のテープドライブを 2 台用いて約 120GB の記憶容量を確保することができる。従来はテープドライブがもっとも大容量の搭載可能なストレージデバイスであったが、現在では HDD の容量がテープドライブとほぼ同じ値に達している。打ち上げ時の衝撃などを考慮すると、何の対策もなく HDD を搭載することは難しいが、BESS-Polar 実験の行われる 2004 年や 2007 年には、HDD が単体で数百 GB ～数 TB に達すると見込まれる。BESS-Polar 実験のデータ収集システムは、第 3 章で解説した記録システムに、イベントベルダーと本章の粒子同定機構を付け加えた形で拡張するものと言える。したがって、最新の大容量 SCSI　HDD を接続することはまず問題ない。逆にいえば、HDD に適切な対策を施して、積極的に搭載しなければ、これだけのイベント記録することは不可能である。しかしながら、開発を行う 2001 年や 2002 年の地点で、これほどの大容量デバイスを入手するのは不可能であり、それ以外の方法で容量の埋め合わせをしなければならない。

BESS 実験で測定している一次宇宙線の約 90%は陽子で 10%がヘリウムであり、その大半は数 GeV 以下である。また、BESS-Polar 実験で主目的とする反陽子は、陽子のイベント 10^5 回に対して 1 イベント程度である。したがってリアルタイムに粒子の同定を行い陽

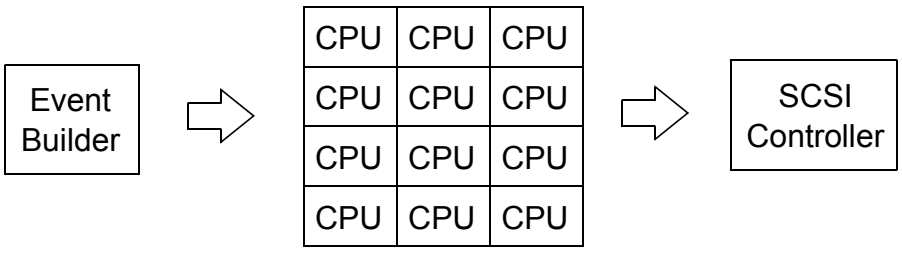
子イベントの記録を抑えることで、この問題の解決が可能である。実際、従来の BESS 測定器でもこのような機構をトラックトリガーと Transputer Bank の組み合わせで行っていた。しかし、2.2.3 節で述べたように、電力を抑えるためにトラックトリガーは、よりシンプルな構造になり、トリガーの段階で粒子を排除する事ができなくなる。現在トラックトリーガーで排除されている粒子の処理も含めた能力が、この新しいリアルタイム粒子識別機構には必要である。

それでは、新システムでは、それをどのようなハードウェア構成にするのが適当か見積もる必要がある。CPU を用いてこのような処理を行った場合でも、その演算能力によって図 4.1 に示すように(a)1 台の高速 CPU による処理、(b)並列プロセッサによる分散処理、(c)組み込み型プロセッサと専用コプロセッサによる構成、(d)自作調停回路によるマルチ CPU 化が考えられる。(a)は PC に用いられるような Pentium Processor 等の高速 CISC Processor の指す。あるいは、WS に用いられるような高速 RISC プロセッサもこの候補である。このカテゴリの CPU は、概して消費電力が大きく、それに付随する発熱の問題も避けられない。しかし、メイン制御機構とオンラインの粒子同定を 1 つの CPU でなす事が出きれば、通信等に必要な LSI や回路が節約でき、結果として全体的には消費電力を低く抑えられる可能性もある。(b)は並列処理を前提に設計された Processor が中心であり、現在の BESS で用いられている Transputer もこの部類にはいる。一般的にはこのような CPU はスーパーコンピュータや特殊用途向けなので、ラインナップが少ない上に低消費電力型のものも少ない。(c)はカーナビやマルチメディア端末向けに開発された高性能・低消費電力型プロセッサと浮動点演算のためのコプロセッサの組み合わせである。あらかじめ、コプロセッサの仕様を前提に設計された CPU は、その接続のための端子や命令が搭載されている。そのような製品を用いると開発の期間や労力も短縮できる。(d)はコプロセッサを前提としていない製品でも調停回路を加えるこで、数台接続できよう独自に設計した回路である。Low Power な DSP を Rigidity 算出の専用 LSI と見える設計すれば、実際のところは、Main CPU と専用 LSI という一般的な組み合わせに仕上げることが出来る。

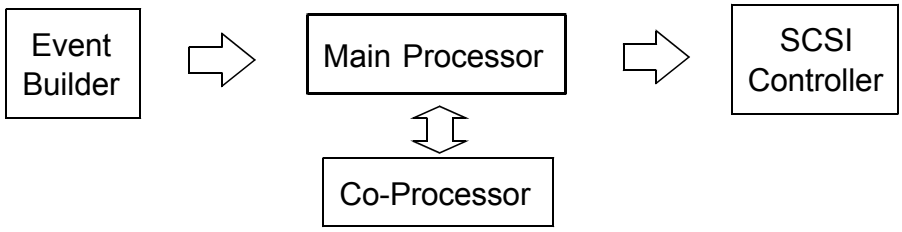
これらのどの選択肢が妥当であるかは、必要とする処理のタイプ(演算や命令の種類)に多分に依存するといってよい。そこで、本実験に特化した粒子同定のアルゴリズムを研究することで、処理のそのものの軽減を計るとともに、新たに開発するリアルタイム粒子同定機構のハードウェア構成を決定することができると考えられる。このために、実際に PC 上で BESS 実験で収集されたの上空でのデータを用いてトラックの検出を行い、Rigidity 算出をするアルゴリズムの研究を行った。基本的には粒子を同定したければ、質量を求めればよいが、実際のところ、低エネルギー陽子を排除したい場合は、|Z| = 1 で Rigidity > 0 の粒子を捨てればよい。したがって、本プログラムでは粒子の質量を求めていない。この章ではその結果と問題点について述べる。また、第 5 章においてこの結果を踏まえて BESS-Polar 実験でどのようなハードウェアを開発するべきかを議論する。



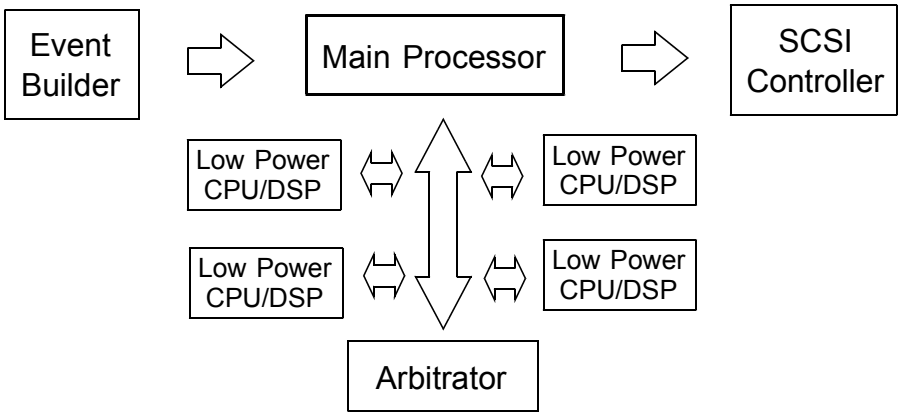
(a) 高機能CPUによる一括処理、構成はもっともシンプル



(b) 並列処理型のプロセッサによる構成



(c) メインCPUとコプロセッサによる並列処理型



(d) メインCPUと複数のプロセッサによる協調回路

図 4.1 粒子識別システムのハードウェア構成例

4.2 開発環境

プログラムの開発は、基本的に Linux 上で行った。この理由は開発に必要なツールがすでに多く用意されているからである。その中でもコンパイラである"gcc"は、多くのプラットホーム上に移植されており、この研究の後に開発するシステムでも動作する可能性があること。また、"gcc"のクロスコンパイル機能を用いて他のプラットホーム上でのネイティブコードを生成し、実際にその処理に必要なマシンコードが計算できることは、特に重要であると考えた。

また、プログラムは、ほぼ ANSI C に基づいて記述するように心がけてある。これは、上述の移植性をふまえることはもちろん、根本的にはパケット化されているデータの実座標への展開やそれから粒子の軌跡を再構成するといった純粋なアルゴリズムの問題であるため、ハードウェアに特化した記述は必要ないはずだからである。

本研究で作成するプログラムは、基本的には Rigidity を算出するものであるが、それ以外にも、本プログラムを開発・デバッグするための補助的なプログラムを開発した。それらを用いた一連の流れを図 4.2 に示す。プログラムへの入力には、フライト時と同じようにイベントビルダーで 1 パケットにされたデータを入力するのが妥当である。しかし、実際に用いるデータは、過去の BESS 実験でテープに記録された生データである。これにはイベントデータの他にモニターやコマンドのログなど様々なデータが混じっている。そこで生データからイベントデータのみを抽出するフィルター・プログラムが必要である。また、実際に再構成したトラックを視覚的に見ることが出きるとデバッグの効率が格段によくなる。そこで、再構成されたイベントのディスプレイプログラムも作成した(図 4.3)。

さらに、実際に算出された Rigidity の値をオフラインの解析結果と比較して、その評価をする必要がある。これには基本的に数万〜数百万の粒子の統計的な処理を扱うことが多いので、そのような用途に特化したソフトを用いるべきである。今回は、"root"と呼ばれる高エネルギー実験向けのパッケージを利用した。この特長は、C++ベースのインタプリタを搭載しており、Fortran ベースの同種のソフトウェア PAW に比べ構造的な記述も容易であり、また、オブジェクトの再利用が可能な点で効率的な面が多い。マクロは、ほとんどの場合、インクルードファイルの記述を追加するだけで、ネイティブコードとしてコンパイルすることができ、大量に処理を必要とする場合でも、その時間を短縮することも可能である。

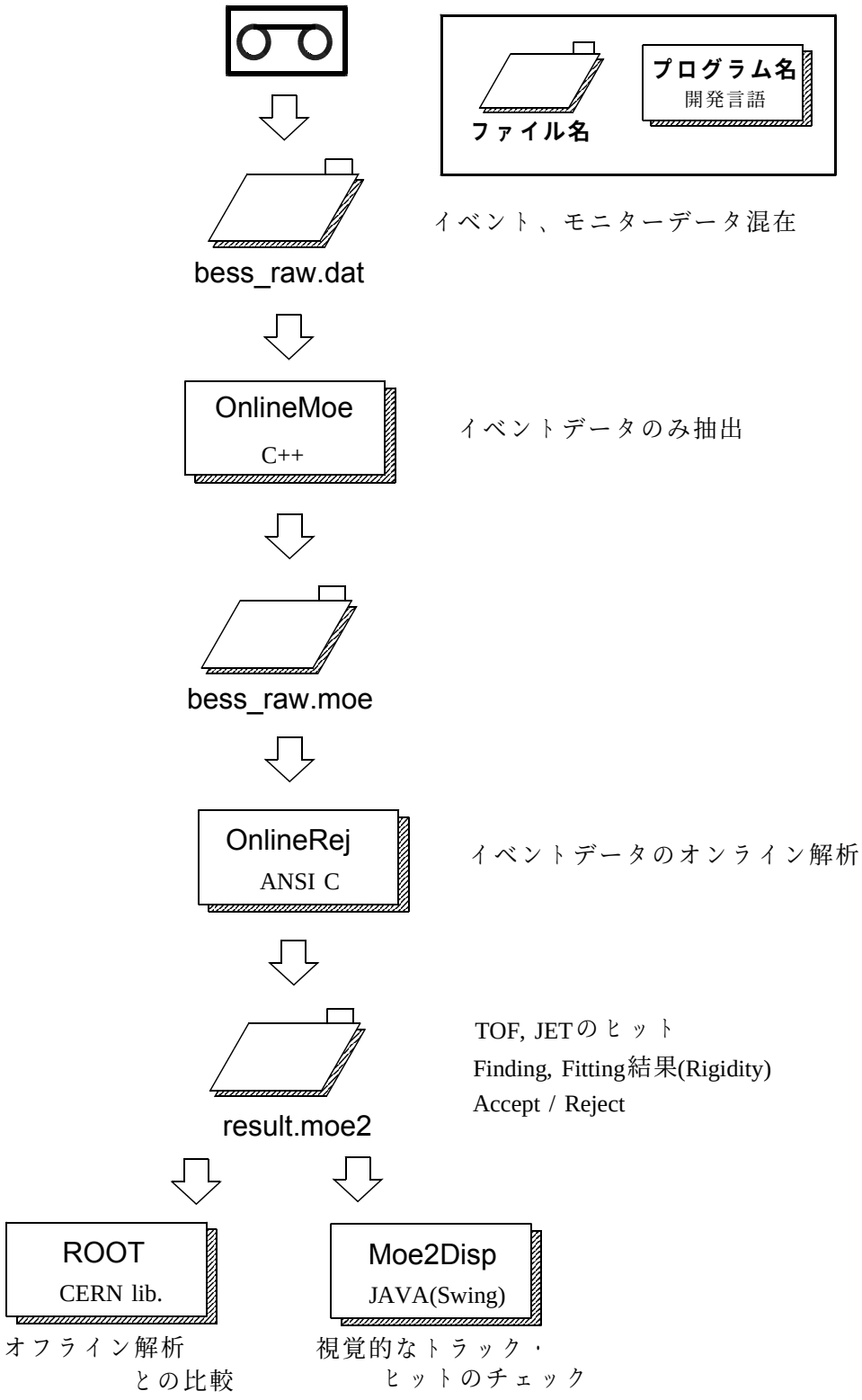
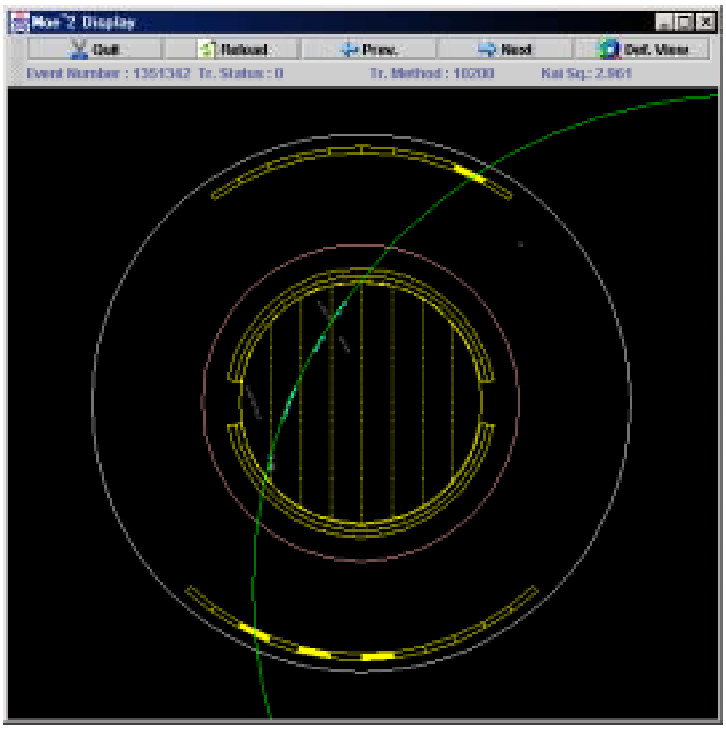


図 4.2 粒子同定プログラムの開発手順



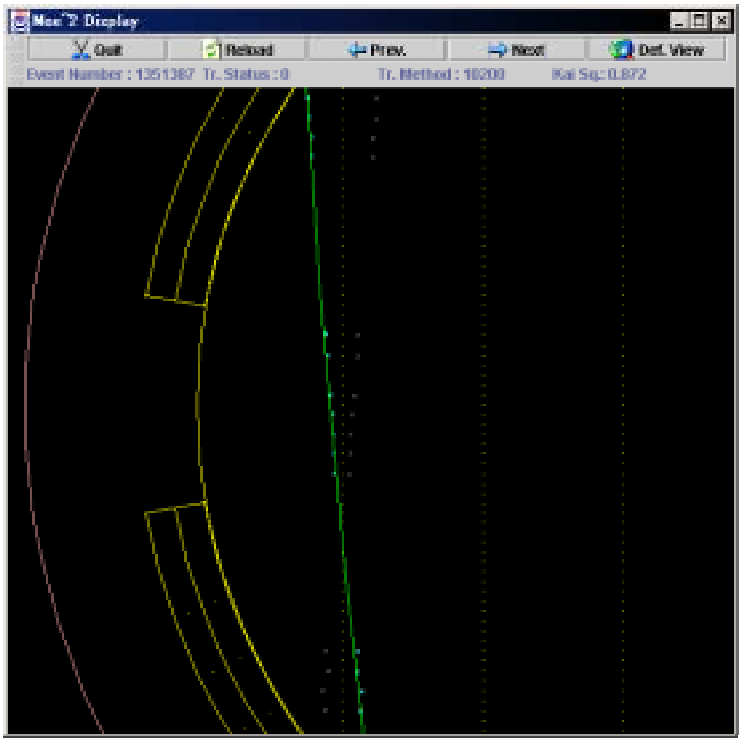


図 4.3 トラックディスプレイプログラム　全体表示(上)と拡大表示(下)

4.3 アルゴリズム

4.3.1 初期化およびプリセレクション

本プログラムのフローチャートを図 4.5 に示す。まず、プログラムが起動すると多くの初期化処理をおこなう。デジタイズされた値から実データに変換する際、毎回式の計算をしていたのでは多くの時間が浪費されるので、それらはあらかじめテーブルを作っておき、使用する時はメモリを参照するだけでよいようにした。作成するテーブルは、TOF の通し番号とその中心や端の位置座標への変換、TDC の値と実時間の対応、チェンバーのワイヤー番号と位置座標、ドリフトタイムと距離の関係などである。これは、ローレンツブーストやキャリブレーションデータによる補正も加えたあとの現実の値である。

その後、プログラムは、データファイルから情報を読み込む。これは実際のシステムでは、イベントビルダーからデータを取ってくるのに相当する。その後、イベントデータのヘッダーファイルを読み込み、TOF のデータが記録された CAMAC 系統のデータであるか、チェンバーのデータが記録された FADC のデータであるかを識別し、それぞれ検出器の情報を取り出す。基本的に CAMAC 系統のデータは、CAMAC クレートのモジュール番号とチャンネル番号、そしてそのチャンネルで測定された値が記録されている。モジュール番号からどのカウンターの ADC なのか TDC なのか、それとも別のモジュールかを判定し、TOF 等の必要なカウンターの情報を取り出す。

FADC のデータは、1 ヒットのデータが 64bit 固定であり、ヒットのあったチャンネルだけ連続して詰まっている。この中は、いくつかの情報があるが、このプログラムで使用するのは、パルス幅と電荷およびチャンネル番号である。この段階でヒットがゼロのイベントは、たまたま TOF にノイズが上下ともに入ったとみなし、このイベントの処理を終了する。また、ヒット数の少ないイベント(約 2 〜 3 ヒット以下）は、軌跡の再構成が困難なので、これもこの地点で処理を終了する。それ以外は、FADC のチャンネル番号からワイヤーの実座標へ変換する。また、これらの変換が終了すると、ノイズ等のヒットを排除するため、パルス幅と電荷がある程度、大きいもののみをプログラム自身が選択する。

4.3.2 トラックの検出とフィッティングの流れ

ここまでで、粒子の再構成に必要な情報は一通りそろった。この次の段階では粒子の軌跡を再構成するが、1 つのアルゴリズムですべてのヒットパターンのフィットをを行うのは、そのコードが複雑になってしまい、高速な処理と両立させるのは困難であろう。そこで、TOF のヒットパターンにより次の 3 通りに大別し、若干異なるアプローチで粒子のフィッティングを行う。(a)上下の TOF カウンターがそれぞれ 1 つずつヒットがある場合、

(b)上下どちらかのヒットが 1 つであり、他には複数ヒットがある場合、(c)上下とも複数ヒットがある場合である。どの場合もフィットの流れは、まずトラックの切れ端であろう 3 点を見つけだし、それらを外挿していき、最終的にそれらの点を円弧でフィッティングする。チェックとして TOF のヒットとマッチしているか、χ^2 の値が著しく大きくないかなどを試す。なお、このとき、円弧のフィッティングには、Karimaki-Method[21]という高速アルゴリズムを用いた。基本的には直線のフィットの同様に、χ^2 を各パラメータで偏微分したときの値が 0 になる条件と実用上問題の少ない近似を用いて解析的にパラメータを求める方法である。また、このチェックをパスすると、そこで得られた Rigidity は正しいものとして、アクセプトするかリジェクトするかなどの判断基準に用いられる。

この方法でフィットが出来なかった場合は、別の方法でやり直す。これには現在の BESS の Transputer Bank で用いられている汎用的な方法で行う。これは、チェンバーの中央部のヒットからトラックになるような組み合わせを探し、それを上下に向かって外挿していく方法である。トラックの数や曲率によらず、フィッティングに成功しやすいやり方であるが、組み合わせの数が多いため多少の処理時間が必要である。TOF のパターンで大別するとトラックのセグメントが見つけやすく、その際、試みる組み合わせが少ないことが多い。

もちろん、このように失敗したときに別の方法でやり直すことは、余分な処理を増やすので若干効率は落ちるが、頻繁に行う処理をより短かい時間でこなせば、あまり起こらないような処理は時間がかかっても、全体としては高速化できるはずである。これらの 3 通りについてヒット・ファインディングの方法を説明する。

4.3.3 トラックセグメントの検出

(a)の基本的な処理の流れは、まず、チェンバーの上部からワイヤーをチェックしていき、ヒットが 3 層連続してあった場合、そのすべての組み合わせを試し、その 3 点がほとんど直線上にあるものを選び出す(図 4.4)。図を見ても分かるようにほとんどどの場合、直線のものはすぐに見出される。しかし、チェンバーの信号が検出される時間よりワイヤからの距離を算出しているが、それがワイヤーの右から来たシグナルなのか左から来たのかは分からない。したがって図 4.4 のようにワイヤーを中心として 2 つの候補ががミラー状に残る。そこで、上下の TOF を結んだ直線の傾きを計算し、それに近い方を本物と見なす。正しいトラックは上下の TOF を結んだ直線と近い傾きをしているはずである。この仮定ははほとんどの場合、正しいが、粒子が検出器のほぼ真上から真下に向かって通過したような場合、正しいものとミラーの傾きの差は小さい。このような場合は、下から同様

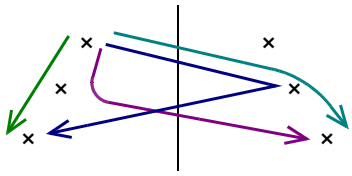


図 4.4 ヒットの組み合わせ

の処理を行い、そこで有意な差があればそのまま処理を続行する。また、それでも不明瞭な場合は、汎用ファインディングに処理を切り替える。

ついで、最初の 3 点が決まると、その 3 点を結んだ直線上のある範囲にある点を探索する。これを次の点と決める。このような操作を繰り返してトラックを延長しく。一番下の層まで探索し終わると、その点をフィッティングする。また、フィットに使った以外にもヒットが何層にもわたって残っている場合、今フィットした点を除いて、もう一度同じ処理を実行する。このようなやり方をすることで、マルチトラックにも対応する。

(b)は、基本的に(a)の処理を TOF のヒットが 1 つの方から処理をはじめる。しかし、TOF の傾きを頼りにミラーを解くことができないので、直線の延長上にヒットした TOF が片方だけ有意にある場合は、それをトラックの始まりと見なす。曖昧な場合や複数のトラックを形成していそうな場合は、汎用処理にまわす。

(c)は、Vessel 上部やコイル・ACC 等でインタラクションした粒子が上下ともヒットさせていることもあるが、実際はシングルトラックのきれいなイベントにも関わらず、TOF または測定エレクトロニクスにに、たまたまノイズが入ってシグナルを出していることが多いので、そのような場合は基本的には上と同じような処理を行えばよい。そこで、処理をはじめる層のヒットの数で条件分岐し、多い場合は汎用処理に回す。

なお、BESS-Polar 実験では、チェンバーのワイヤー数などが、一部変更されるが、実質的には、このアルゴリズを大きく変更する必要ない。しかし今後、BESS-Polar 測定器のシュミレーションプログラムを作成し、そのフルシュミレーションの結果に対して、あらためて評価をする必要があろう。

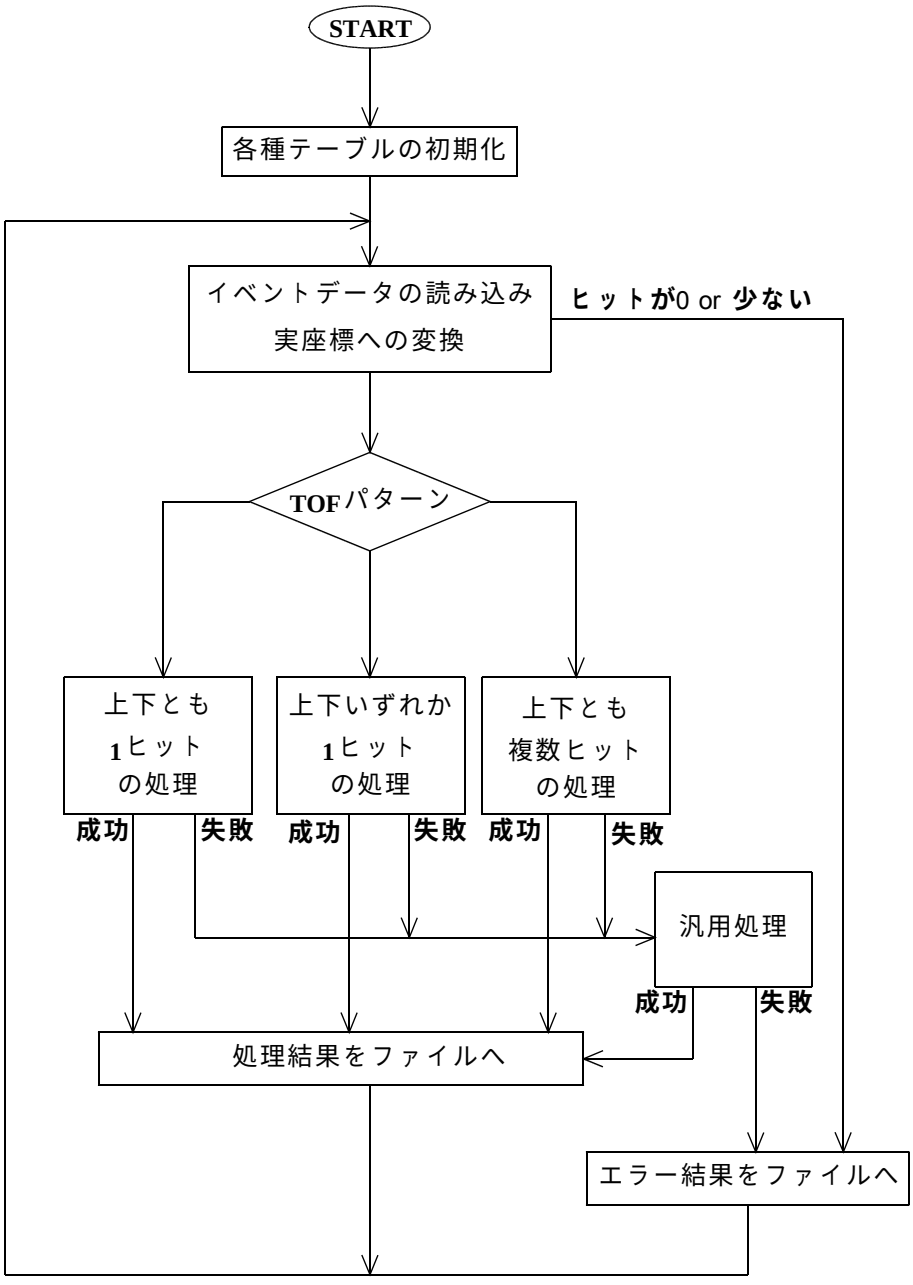


図 4.5 粒子同定プログラムのアルゴリズム

4.4 性能評価

4.4.1 トラックの検出能力と処理時間

図 4.6 〜 4.9 にイベントディスプレイで表示されるトラックの例を示す。イベントディスプレイでは処理を簡略化するためにコイルの外側でも円弧のまま表示してあるが、メインプログラムが TOF とのマッチングをチェックするときには、コイルと外側からは直線であるとして計算している。また、表 4.1 にトラック検出能力を示すが、これはプリセレクションでヒットが少ないイベントを除いたデータに対しての値である。全体のおおよそ 6 割は、トラックの検出に成功している。これらのほとんどはシングルトラックイベントあり、きれいなシングルトラックであれば検出できる割合は 90%以上である。トラック検出を保留にしたものの大半は、図 4.9 のように JET チェンバーのヒットが多すぎて、あまりにも時間がかかるために処理を行わなかったものである。そのなかの約 1/3 はオフラインではトラックを検出することに成功しているが、それらのほとんどはマルチトラックイベントである。また、2/3 は Fit ができなかったか、あるいは解析に使えないような汚いヒットパターンである。

表 4.1 Fit の成功率とその判定

結果	判定	割合
Fit成功	正	62%
	誤	2%
Fit失敗		2%
保留	Fit可能	11%
	Fit不可	23%

表 4.2 に各プロセスの処理時間を示す。計測は Pentium Processor 850MHz マシンで行った。PreProcess は、実際にトラックの探索を行う前にワイヤー番号から実際の位置座標に変換するなどにかかる時間であり、その横の 3 つの処理時間は、この時間も含んだものである。Transputer での処理時間はシングルトラックで 4ms であるので、ほとんど同じアルゴリズムを採用した汎用処理と比較した場合、約 1/40 に短縮されている。これは、Transputer の駆動クロックが 20MHz なのでおおざっぱには、850MHz に比例して 1/40 程度になったといえよう。また、このプログラムで付加されたシングルトラックに特化した処理は、汎用処理の 2 倍近い高速化が成功したといえる。

表 4.2 処理時間

	PreProcess	TOF 上下1	TOF片方1	汎用処理
平均時間[μs]	15.167	59.857	60.643	98.417
σ	9.347	25.386	20.479	53.147

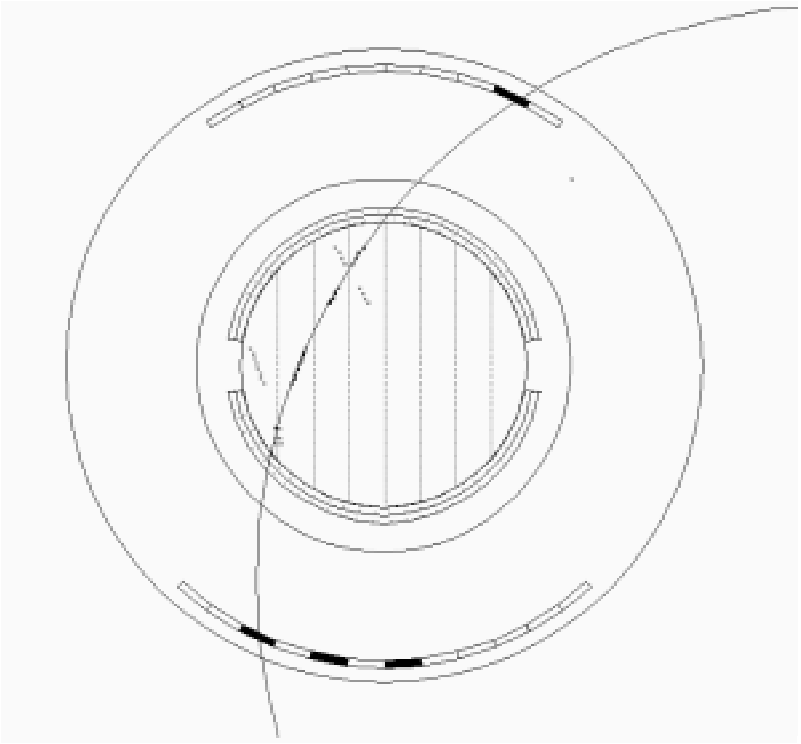


図 4.6 トラック検出の例(低エネルギー)

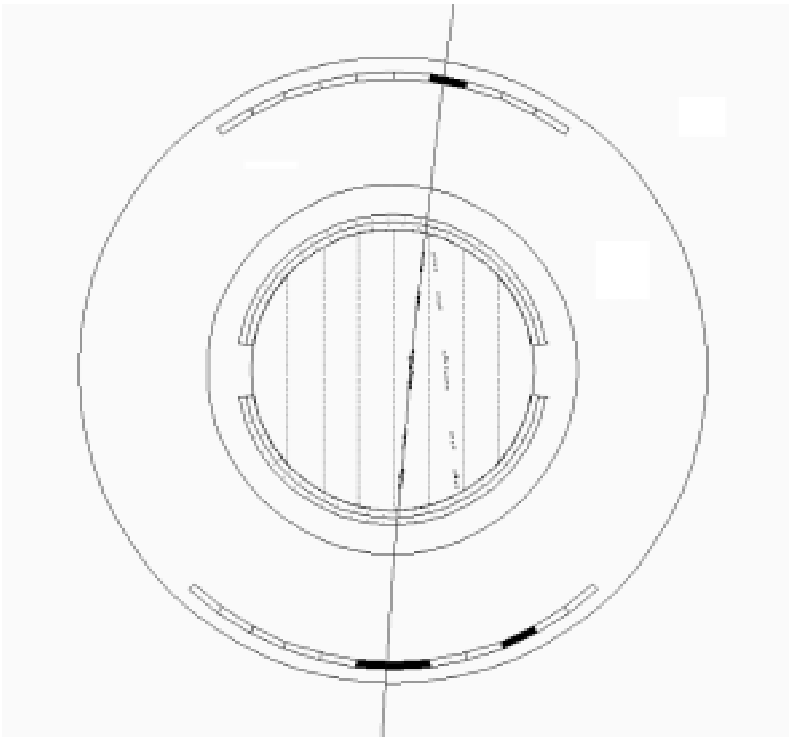


図 4.7 トラック検出の例(高エネルギー)

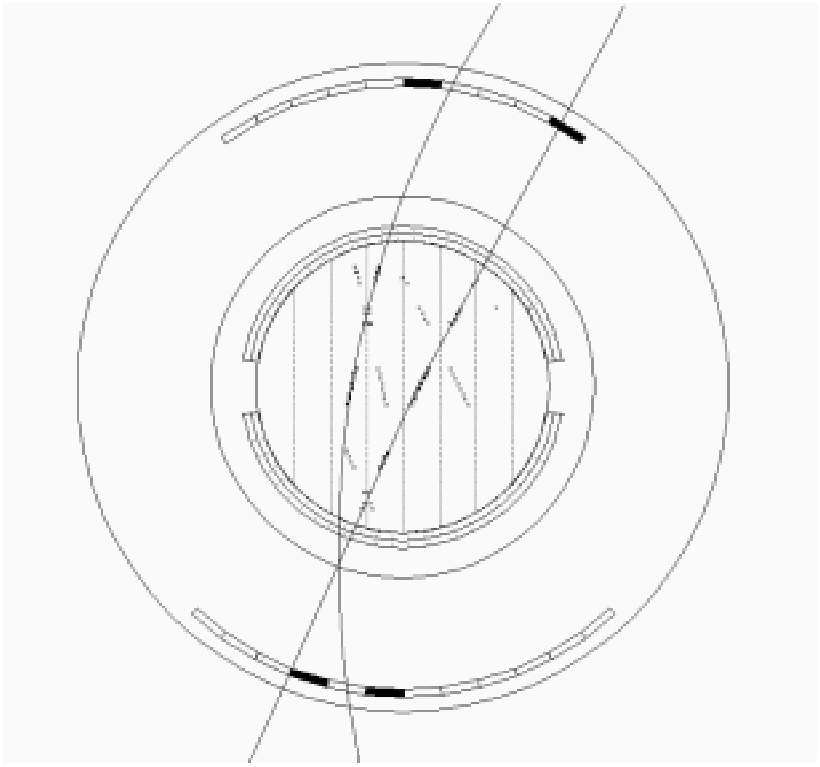


図 4.8 トラック検出の例(マルチトラック)

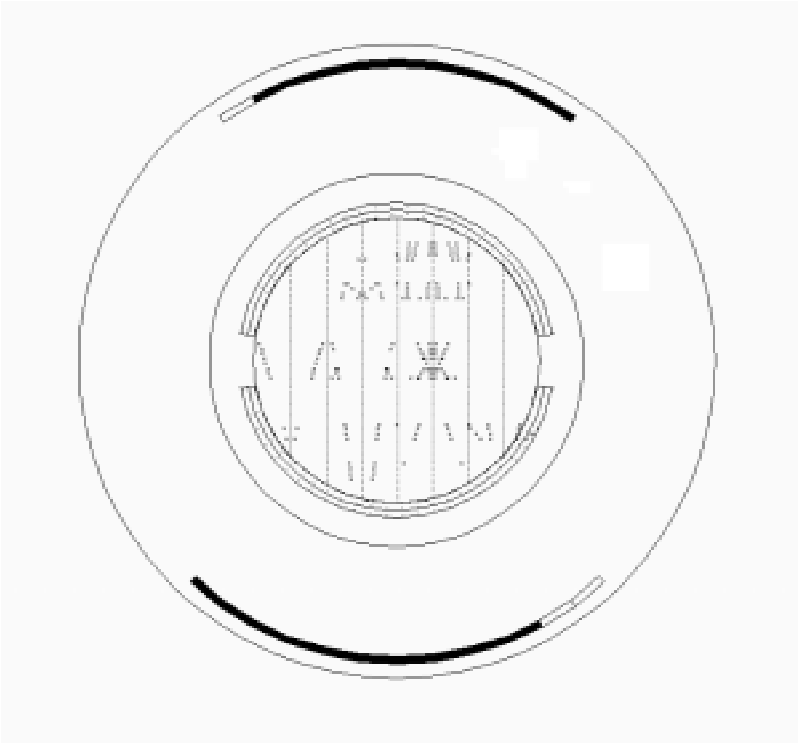


図 4.9 検出処理保留ヒットパターンの例

4.4.2 Rigidity Error

図 4.10 はフィットが成功したトラックの Rigidity の Residual　error の分布である。オフラインで解析した Rigidity の値を正しい値としている。Rigidity の分解能は、0.09GV^{-1} であり、現在の BESS のトラックトリガーの 0.53GV^{-1} に対して 6 倍近い値である。また、ほとんど同じ情報を用いて処理している Transputer　Bank は 0.08GV^{-1} なので、キャリブレーション等が十分でないオンライン処理ではこのあたりが精度の限界といえよう。

　また、これにより Rigidity が正のイベントを捨てた場合の efficiency を図 4.11 に示す。上空で大部分を占める 4GV(0.25GV^{-1})以下の粒子に対して 90%以上の efficiecy を確保している。

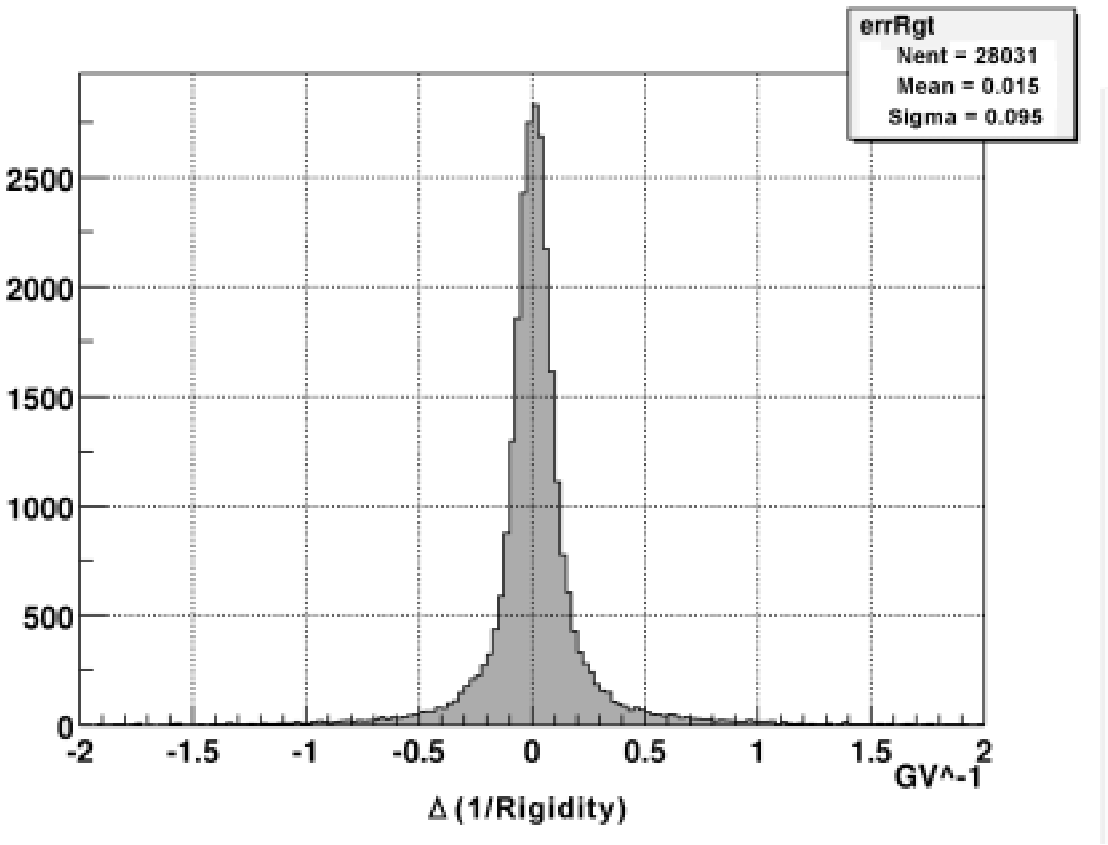


図 4.10 Residual error

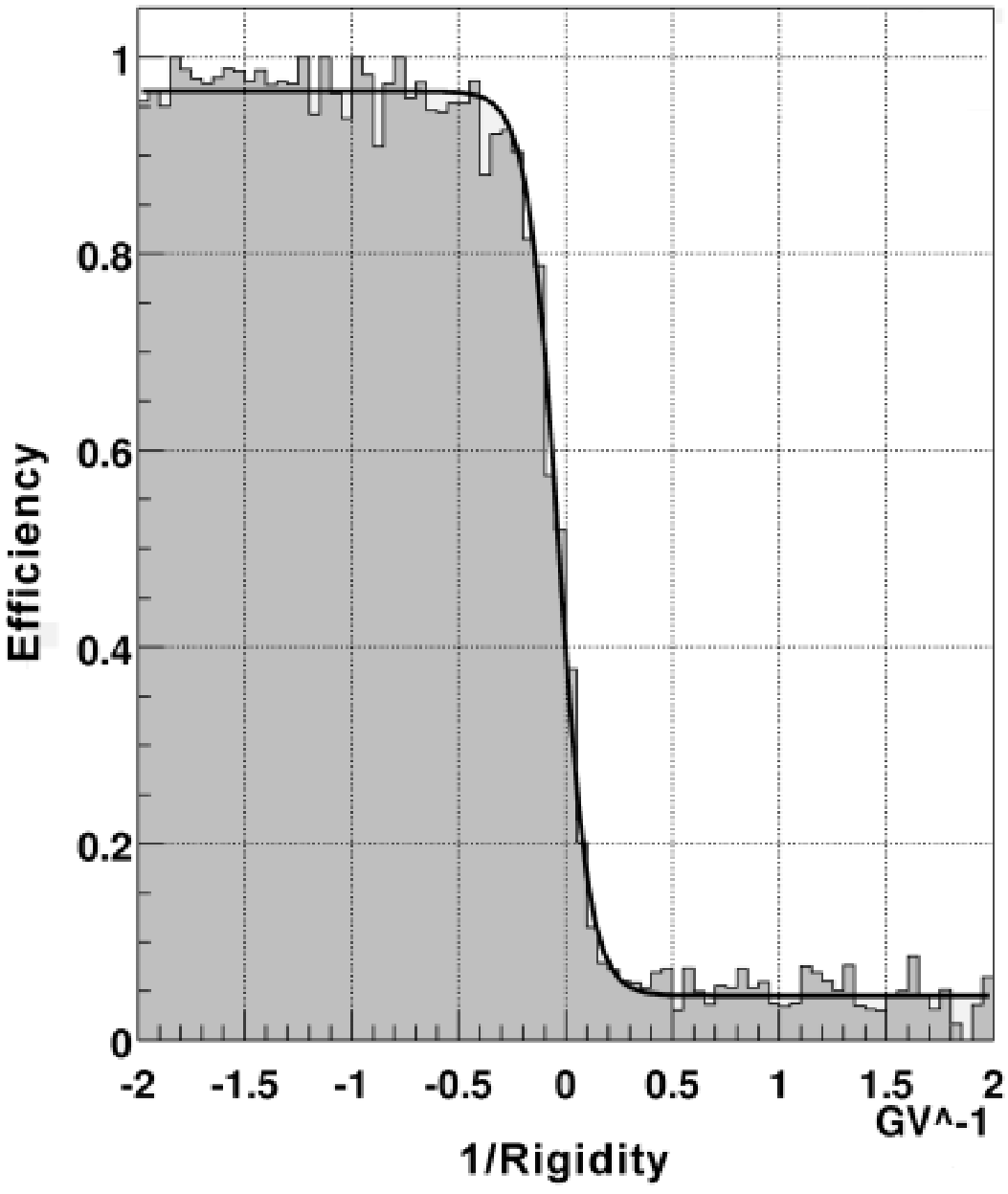


図 4.11 rigidity が正のイベントを捨てた場合の efficiency

4.4.3 データ集レートの考察

以下にストレージに書き込むイベントレートを見積もってみる。これは、BESS-Polar 実験におけるストレージデバイスとして何を選択するかの 1 つの指針になるはずである。

BESS-Polar 測定器でもアクセプタンスはほとんど変わらないのでイベントレートは 2000Hz とみてよい。そのうち 2.5%(50Hz)は efficiency 等の算出のため、バイアスをかけずに一定の割合で必ず記録すると

2000Hz(All Trigger) = 1950Hz + 50Hz(unbias)

ここで 1950Hz のうち、以下のように 10%はヒットがきわめて少ないイベントであり、80%が|Z| = 1、10%が|Z| > 2 である。

1950Hz = 960Hz(|Z|=1, Fit 済) + 640Hz(|Z|=1, Fit 未)+ 175Hz(|Z|>2) + 175Hz(Bad Track)

また、|Z|=1 でフィットを保留した 640Hz のうち 2/3 は、実際には解析等で使われることがないので捨て、960Hz は図 4.12 に示すように Z<0 の粒子を選択すれば約 1/10 にする事が出きる。さらに Bad Track のデータも破棄すると書き込むべきレートは

96Hz(select) + 175Hz(|Z|>2) + 50Hz(unbias) = 321Hz

である。1 イベントのデータを 2KB とし、2003 年に 10 日のフライトを行ったとして必要なストレージの容量を計算すると以下のように 530GB になる。これは、例えば、200GB の HDD を 3 台搭載すればデータ収集が可能である。

321Hz×3600sec×24hrs×10days×2KB = 530GB

もし、現在と同じようにテープドライブで収集を行う場合、60GB のドライブを 5 台搭載すると 300GB なので、たとえば、反陽子のみにねらいを絞って、|Z|>2 のイベントを 1/5 にすると

{96Hz(select) + 50Hz(unbias) + 17.5Hz(|Z|>2)}×3600sec×24hrs×10days×2KB = 290GB

という、データ収集の方針になろう。また、その他の物理のために中庸的なデータ収集の方針としては、例えば、|Z|>2 イベントや破棄していた保留イベント(多くはマルチトラック)も 3 回に 1 回ストレージに書き込むと

{96Hz(select)+60Hz(|Z|>2)+120Hz(keep)+50Hz(unbias)}×3600×24×10×2K = 550GB

となって、これも大容量 HDD に頼らなければデータ収集は難しい。したがって、ここ数年でますます肥大化するであろう HDD はにおける有力なストレージ候補である。

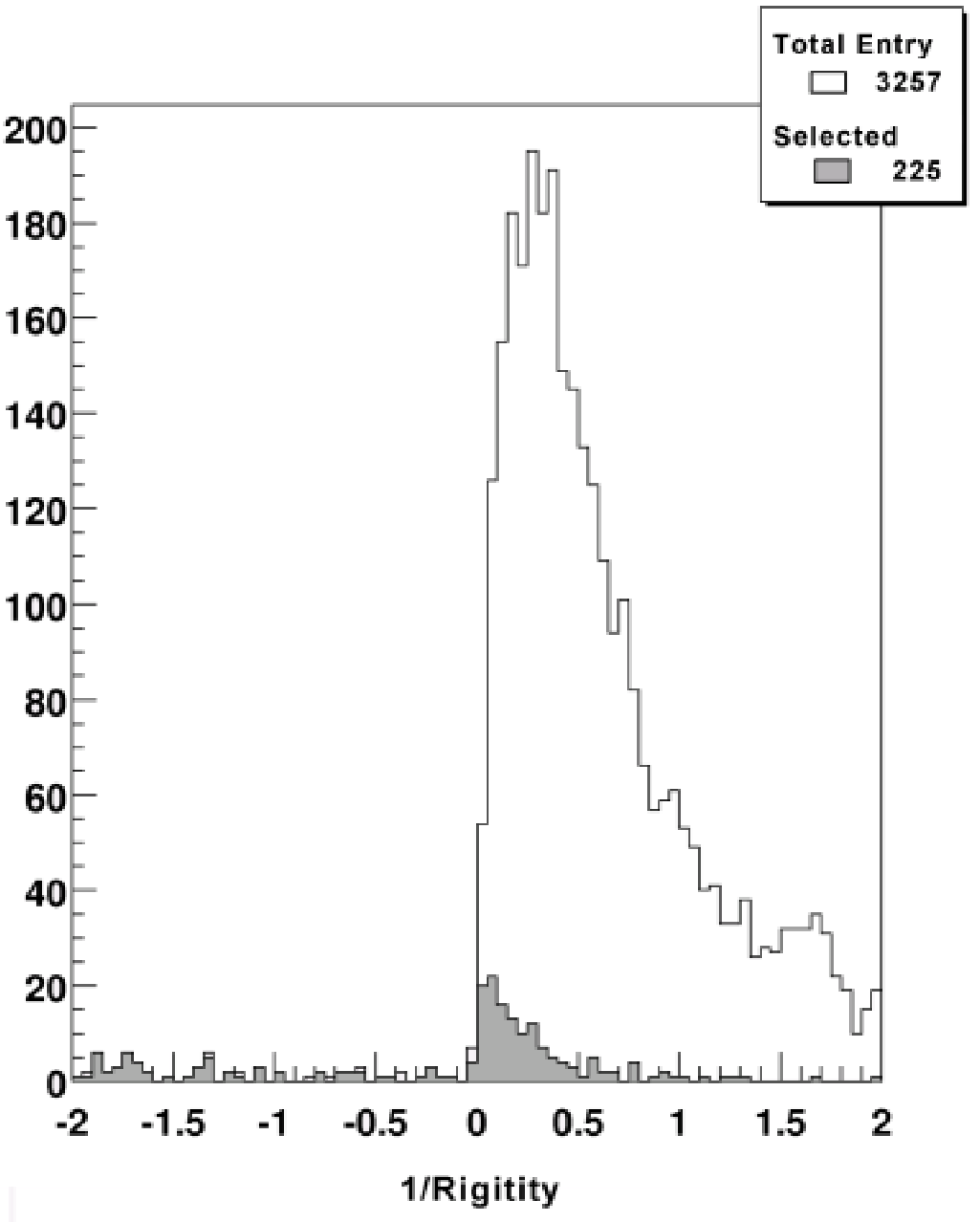


図 4.12 |Z| = 1 粒子に対して Z<0 の粒子の選択結果

第 5 章　BESS-Polar 実験へ向けての開発課題

第４章では、移植性の高い C 言語により粒子の Rigidity を求めるプログラムを作成した。その結果、汎用ルーティンは現在の BESS で用いられているコードとほぼ同じ値を、きれいなシングルトラックイベントでは２倍近いのパフォーマンスを得ることができた。この章では、そのプログラムをどのようなハードウェアの上で実行させるべきかを検討する。

5.1 New データ収集系のハードウェア構成

これまでに何度か述べたが、消費電力を減らすためには、部品点数や電力の大きい通信用トランシーバーを削減するべきである。これは、並列処理プロセッサを用いて分散的に行っていた BESS 測定器のデータ収集系を根本的に変更することを意味する。しかし、そのように収集機能の集中化は、全体としてスペースや重量を低減させることにつながるので歓迎すべきことであろう。

それでは、具体的にどのようなハードウェアをこれから開発すべきであろうか。最低限、この集中型データ収集装置において必要な機能は、(1)イベントビルダー、(2)中央制御機構(CPU)、(3)リアルタイム粒子同定システム、(4)SCSI コントローラーである。BESS-Polar 測定器でのトリガーシステムは、TOF の上下コインシデンスによる T0 トリガーで FADC や ADC/TDC がデータ収集を始め、A/D 変換が終わり次第、自身のバッファーにプッシュして、Busy をクリアする。すべてのモジュールがクリアを出すまでが、トリガーの dead-time になる。TOF や ACC のデータを収集する ADC/TDC は、そのすべてを新たに開発するので、このようなやり方に従って設計を行えばよい。FADC については、BESS-TeV 実験のために低消費電力タイプのものが開発されたのでそれを流用すればよい。しかし、現在の FADC のクレートコントローラは FIFO を搭載していないので、dead-time は 100 ～ 200[μs] となって、図 5.1 に示したように live-time は 50%程度になる。したがって、各チャンネルに対して FIFO を設けたクレートコントローラーを開発すれば、dead-time はチェンバーのドリフトタイム＋ FADC の変換時間とみていいので～ 50usec と見積もられる。このよう

にすると FIFO の段数が 4 ～ 5 程度でも 90%を超える高い live-time を望める。

一方、FIFO に詰められたデータは、A/D 変換装置とは独立してイベントビルダーへの転送を試みる。イベントビルダーは、それら計測モジュールからのデータを 1 つのイベントパケットにまとめる機構である。各モジュールから送られてくるデータのタイミングは非同期であるが、その順番はイベントの順になっているはずである。何かの不具合でその順序がずれると、イベントの再構成が不可能になるので、マスタートリガーモジュールは、T0 トリガーと同時にイベント番号を各計測モジュールに送る。各モジュールは、そのイベント番号をデータとともに送信する。イベントビルダーでは、その番号をチェックし、すべてのデータが同じイベントからであることを保証する。

(2)中央制御機構は、その 1 つになったイベントパケットを粒子同定システムに送ったり、その結果を受けて SCSI コントローラへ転送をする。したがって、この部分は、装置全体を掌握しており、ストレージへの書き込みやイベントリジェクションに対するパラメータの変更も一手におこなう。またこれらのシステムは PCI バス上に構築する予定であるが、起動時には各デバイスの各デバイスのコンフィグレーションも行う。また、(1)(2)は機構としては独立しているが、１つのボード上に両方を搭載することも可能であろう。その場合、Event　Builder で処理されたデータは、DMA コントロラーよりローカルバスのみを用いてメモリに転送されるので転送効率もよい。このブロック図を図 5.2 に示す。(3)については次節で述べる。(4)は市販の Compact　PCI 対応のボードが使用できるので特に開発の必要ない。

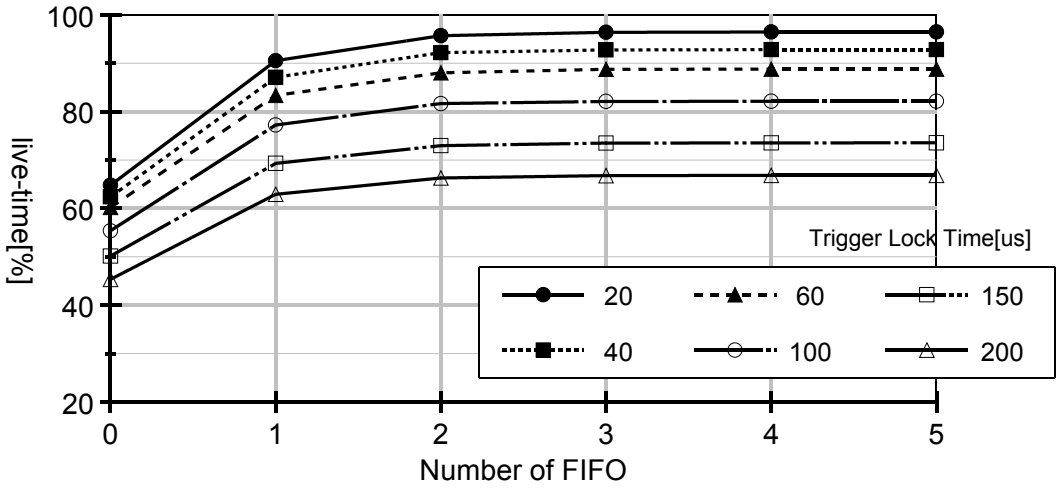


図 5.1 計測モジュールの FIFO 段数と live-time

5.2 粒子同定システムのハードウェア

第 4 章のアルゴリズムの研究では、従来と同じアルゴリズムを用いた方法では C 言語によりほぼ同じパフォーマンスを得た。全体の半分近い綺麗なシングルトラックイベントについては 2 倍近い速度で処理が行えるようになった。これにより全体として数割の効率化がねらえるが、現在 15 台の Transputer で分散して行っている処理が 1 台でできるようになったわけではない。4.4.1 節の議論で処理能力は、ほぼクロック周波数に比例してることをみた。厳密に議論する場合、能力評価の基準である MISP や GFLOP を使うべきであろうが、よほど特殊な CPU でないかぎり、これらの値はクロックに比例していると思ってよいだろう。これによるとトランスピュータは内部 20MHz であり、現在実働が 10 台であることを考えると処理能力 200(MHz・台)という数値になる。また、トラックトリガーが排除されて 4 倍のパワーが必要なので処理能力 800(MHz・台)がおおざっぱに必要である。すなわち、図 4.1(a)のような 1 台の CPU でこれをなすには 800MHz 程度のプロセッサが必要である。例えば、Pentium プロセッサ 800MHz は、2001 年現在において PC に主力製品として用いられるほど、高性能なものであり電力・熱的問題を考慮するとからならずしも最適といえない。(b)のような構成は、そのような意味で十分候補になりうるが、低消費電力かつネイティブに並列処理をサポートしている CPU がほとんど存在しないのでこれも難しい。(c)も CPU × 1 とコプロ× 1 の組み合わせでは、やはり処理能力に難がある。(d)は、多少付加回路が必要なもの低消費電力型の CPU の選択肢は十分にあるのでこれがもっとも適すると思われる。この選択をした場合、開発すべきハードウェアの概念図を図 5.3 にしめす。このような CPU の多くは 100 〜 200MHz 程度で駆動できるのでそのようなものなら 10 台以下でよい。

図 5.3 の例では、オンライン同定用の CPU が Ready 状態になると PLD 内部のレジスタにその旨を書き込む。それを受けて PLD はバス権を要求する。バス権が獲得できると、中央制御用 CPU が直接このボード上のメモリへデータを書き込めるようになるので、イベントデータを転送する。1 つのデータを転送し終わるとオンライン同定用の CPU がトラックの検出を試みる。その結果と Ready になった事を PLD に書き込み、再びバス権を外部システムに渡す。これはメインの CPU から見れば、このボードは単なる専用 LSI の集まりであり、CPU を何台用いようが単なるマスター・スレーブ方式になる。

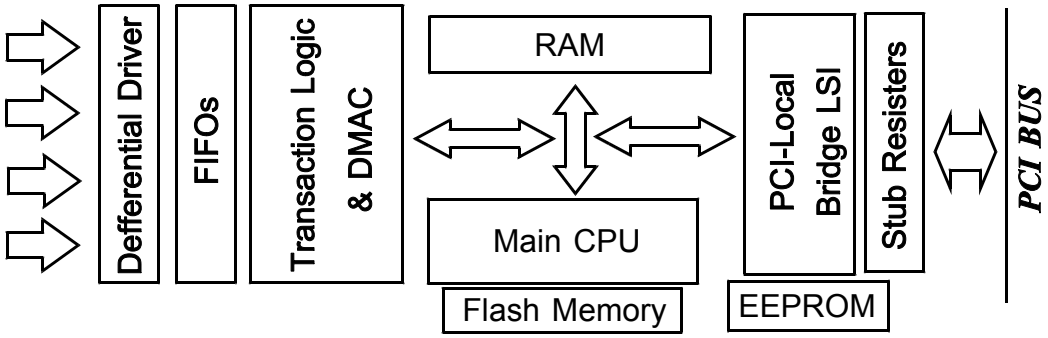


図 5.2 Event Bulding システム

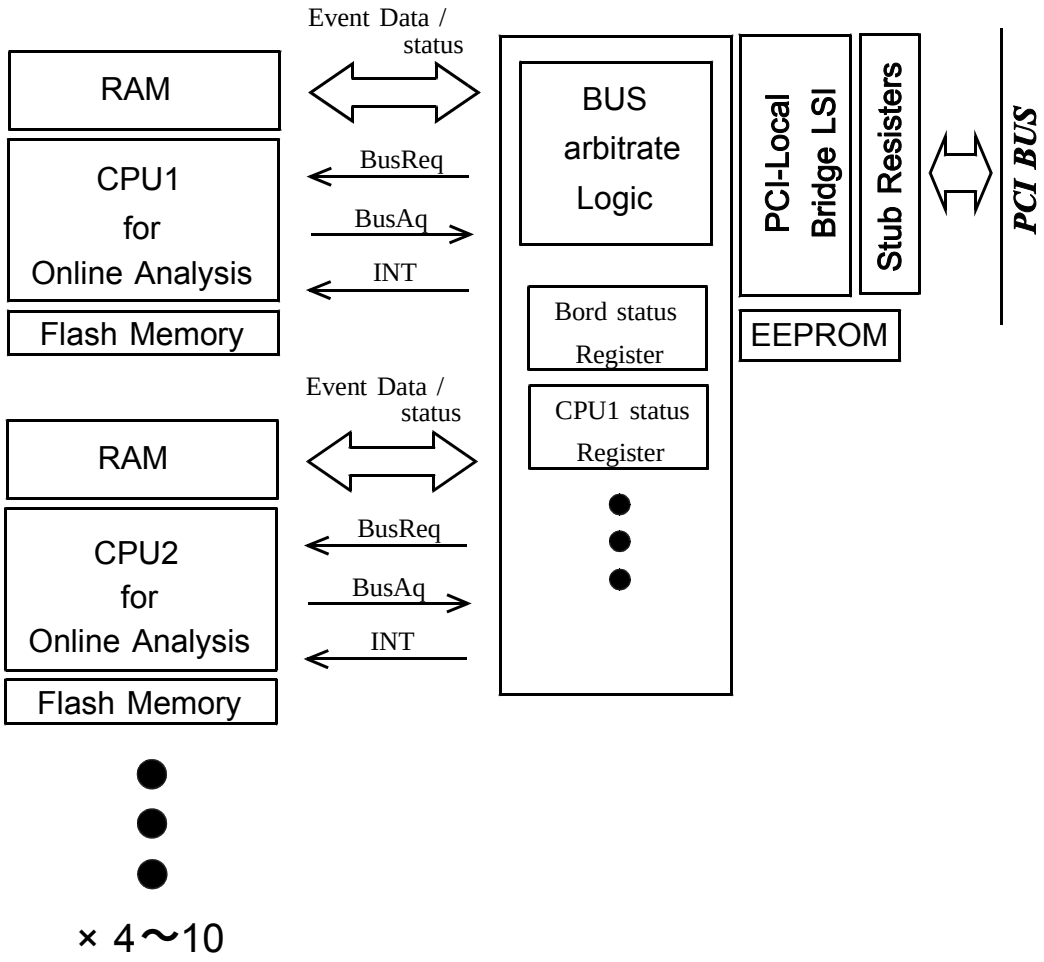


図 5.3 Online 粒子同定システム

5.3 自己診断機構および自律的復旧機能

10 〜 20 日にわたる南極フライトでは、太陽フレアの影響等により無線による地上からの制御ができなくなることもある。現在のフライトではデータ収集の開始や終了をすべて地上から操作しているが、温度センサーなどの結果をみて、非常事態には自律的にデータを収集を停止すべき場面も出てくるはずである。また、温度が正常値に戻ったときに再び測定を開始するような機能は是非とも搭載したい。どのような状況でそのような操作を自動的に行うかは、さらに注意深い別の議論が必要であるが、ここではハードウェア的にそのようなことが可能かを検討してみる。

結論から言えば、このようなことはすべて可能と言える。すべての情報は C-Bridge を経由して地上へ送信され、地上からのデータも C-Brige がまず受け取る。つまり、C-Brige を中心にしてすべての機器の間で情報の交換することができる。したがって BESS-Polar 測定器に、診断装置を付加し、モニターデータやイベントデータを定期的に受け取ることで自己診断を行うことが出きる。また異常が発生したときに、それに対処するコマンドを発効させれば、C-Brige を通じて地上からのコマンドと同様に測定器を制御することが可能となる。

また、第 4 章で開発したリアルタイム粒子同定プログラムでは、チェンバーのワイヤー位置のキャリブレーションは固定されている。しかし、20 日フライトをした場合、そのようなやり方は、かならずしも最適な結果になるとは思えない。したがって、常に別スレッドでキャリブレーションを行っておいて定期的にその結果を粒子同定システムに反映させるべきである。これも本イベントプロセッシングシステムにキャリブレーションボードを組み込めば、全体としてそれほど大きな変更をしなくてもこれは可能になる。将来に向けて、このような機構の開発も進めるべきである。

第 6 章　まとめ

これまで物理的に重要な成果を上げてきた BESS 実験で、今後の 2 つの観測が計画されている。1 つは TeV 領域までの一次宇宙線を精密に観測する BESS-TeV 実験である。これは大気ニュートリノの絶対流束を計算するために必須のデータを収集する。もう 1 つは、従来の BESS 実験で力を注いできた反陽子観測の精度を格段に向上させる BESS-Polar 実験である。南極において 10 〜 20 日間のフライトをし、これまでの 10 倍以上の反陽子を捕らえるものである。

このどちらの実験においても検出器の開発や改良もさることながら、そこで捕えられる大量の宇宙線イベントをいかにして記録するかという問題がある。記録容量の不足に対しては、記憶装置そのものの大容量化を図ることと、必要でないイベントは記録しないという 2 つの方法を採るべきである。これまでの BESS 実験でも目的とする反陽子を捕らえるために、宇宙線の大部分を占める陽子イベントをリアルタイムに排除する機能が備わっていた。BESS-Polar 実験においては設計の段階からこのようなことを考慮し、全体として電力・重量などの無駄がないものにすべきである。そこでこの 2 点について研究を行った。

まず、1 つめの記憶装置自体の大容量化であるが、我々がテープドライブそのものを開発するのは期間やコスト、技術の点でほぼ不可能である。したがって、市販されている製品を使用することになるが、最新の大容量記録装置の大半は、そのインターフェースとして SCSI-3 を採用している。現在の BESS ではこの規格に対応していないため、新たに SCSI-3 規格のテープドライブと接続・記録できるシステムを開発した。

また、粒子をオンラインで排除する機構は、粒子のトラックを再構成するために処理能力の高い CPU/DSP を使わざるを得ない。しかし、BESS-Polar 測定器では、電源としてソーラーパネルを用いるため、今まで以上に制限がきつくなり、単に高速なプロセッサを載せればよいわけではなく、必要な処理能力を見極め、電力を最低限に抑えなければならない。そこでまず、粒子のトラックを再構成するプログラムを移植性の高い C 言語で記述し、その性能を評価した。アルゴリズムの改良により、全体の半分以上のシングルトラックイベントについては約 1/2 の処理時間に短縮された。また、全体としてどの程度の能力が必要かが概算でき、新たに開発するハードウェアの大きな指針を得た。

謝　　辞

まず、BESS 実験のの物理的意味や具体的研究の指導にあたっていただいた野崎光昭教授に感謝いたします。また、神戸大学において様々な指導いただいた武田廣教授、川越清以助教授、藏重久弥助教授、原俊雄助教授、石井恒次氏、鈴木州氏にも深くお礼を申し上げます。

さらに本システムに製作するに当たって、多くの助言とエレクトロニクスの知識を教授いただいた松本浩氏、前野忠嗣氏、佐々木誠氏、また BESS 実験に関わる多くの具体的な事柄を教えていただいた山本明教授、吉田哲也助教授、吉村浩司氏、佐貫智行氏、多くの共同作業を行った松川武夫氏にも深く感謝いたします。

最後に BESS 実験の創始者であり、BESS-TeV・BESS-Polar 計画の発案者である故折戸先生には、本システムの設計時や現場の両面において叱咤激励をいただき、大いに刺激となりました。心から感謝いたします。

[参考文献]

[1] K. Yoshimura, et al.(BESS collaboration): Phys. Rev. Lett.75(1995) 3792.

[2] H. Matsunaga, et al.(BESS collaboration): Phys. Rev. Lett.81(1998) 4052.

[3] T. Maeno, et al.(BESS collaboration): astro-ph/0010381

[4] S. W.Hawking: Nature 248(1974) 30

[5] G. Jungman and K.Kamionkowski: Phys. Rev. D46(1994) 2316.

[6] T. Sanuki, et al.(BESS collaboration): Astro Phys. J, 545(2000) 1135.

[7] H. Matsunaga, et al.(BESS collaboration): Proc. 22nd Intl. Symposium Space Technology and Science(Morioka), (2000) 1720.

[8] T. Saeki, et al.(BESS collaboration): Phys. Rev. Lett. B422(1998) 319.

[9] Y. Fukuda, et al.:Phys. Rev. Lett., 81(1998) 1562.

[10] M. Honda, T. Kajita, K. Kasahara, and S. Midorikawa, Phys. Rev. D, 52(1995) 4985.

[11] M.J. Ryan, et al.:Phys. Rev. Lett., 28(1972) 985.

[12] I.P. Ivanenko, et al.:Proc. 23rd ICRC(Calgary), 2(1993) 17.

[13] E.S. Seo, et al.: Astro Phys. J, 378(1991) 763

[14] P. Papini, et al.: Proc. 23rd ICRC(Calgray) 1(1993) 579.

[15] M. Boezio, et al.: Astro Phys. J, 518(1999) 457.

[16] W. Menn, et al.: Astro Phys. J, 533(2000) 281.

[17] Ahlen, S., Balebanov, V. M., Battiston, R. et al.: Astrophys.J. 350(1994), 350..

[18] AMS Collaboration: Phys. Lett. B, 471(2000) 215.

[19] O. Adriani et al., Proc. 25th ICRC, 5(1997), 49.

[20] O. Adriani et al., Proc. 26th ICRC, 5(1999), 96.

[21] V. Karimaki, Comput. Phys. Commun. 2(1971)207