2009年度 卒業論文
FADCを用いたμ粒子の寿命測定

奈良女子大学
理学部 物理科学科
上田 怜奈
岡本 枝里香
貴志 佳代

目次

第８章 データ解析

８,１ 全データ
８,２ FADCの時間間隔
８,３ μ粒子のEvent選別

第９章 まとめ

９,１ μ粒子の寿命
９,２ 考察
９,３ 参考文献

第1章 はじめに

１．　１ 実験の目的

地上にはたくさんの原子核や素粒子が降り注いでおり、その粒子は宇宙線と呼ばれている。その宇宙線のほとんどはミュー粒子に崩壊し、その過程で粒子はさまざまな寿命を持ことが分かっている。今回の実験ではCsIシンチレーターを用いて、地表にたどり着くμ粒子を測定すること、またその結晶内で崩壊するμ粒子を使って寿命を測定、解析することを試みる。その過程において、素粒子物理学の実験的研究を行うと共に、高エネルギー物理学の基本的な実験技術を習得することを目的とする。

１．２ 実験課題

今回の実験ではシンチレーションカウンターに入射したμ粒子が崩壊したときのエネルギースペクトルを測定する。寿命の測定にあっては、プラスチックシンチレーター・CsIシンチレーター・光電子増倍管・ＦＡＤＣなどを用いて測定された電気信号を増幅し、ＡＤ変換上のデータ処理を行うことによってμ粒子の寿命を測定する。その過程において、寿命測定の解析プログラムを開発することが今回の課題である。

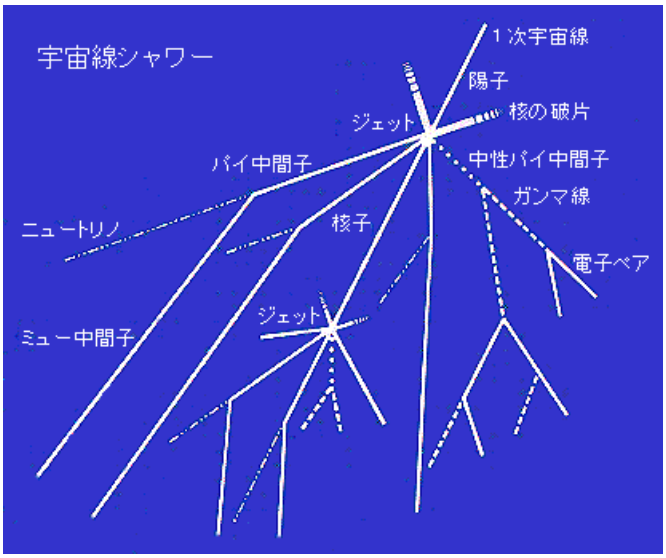
１．３ 本論文の構成

本論文ではまず宇宙線や放射線計測の原理について述べ、次にμ粒子の寿命測定の原理、装置やセットアップについて述べる。さらにデータ収集の方法、そのデータを使っての解析、プログラムの説明などを行い、最後に実験結果について報告する。

第２章 宇宙線・μ粒子

２，１ 宇宙線について

宇宙線とは、宇宙空間から地球に絶えず降り注ぐ高エネルギーの放射線（一次宇宙線）とそれが大気に入射して作る放射線（二次放射線）のことである。宇宙線は常に私たちの身の回りに降り注いでおり、高エネルギーのため建物や私たちの体を突き抜けていく。
そのため、屋内での観測が可能である。
宇宙では、超新星の爆発や、太陽の表面で起こる爆発などで発生した高エネルギーの粒子が飛び交っている。それらの粒子には、陽子（水素の原子核）が約90%、α粒子（ヘリウムの原子核）が約８％、その他の粒子が約１％含まれ、これらの宇宙線を一次宇宙線と呼んでいる。
一次宇宙線が大気に入射すると、大気中に含まれる窒素や酸素などの原子核と衝突し、核反応を起こして放射性同位元素やπ中間子などの粒子を生成する。これら二次的に生成された宇宙線を二次宇宙線と呼ぶ。二次宇宙線は原子核と相互作用し、新たな二次粒子を生成する。
ちなみに、π中間子の寿命は（2.6030±0.0024）$\times 10^{-8}$secである。二次宇宙線のうち、電子やγ線は大気中で吸収され、地上に来る大部分をμ粒子とニュートリノが占めている。



２，２　μ粒子について

μ粒子は、地上に到達する二次宇宙線の荷電粒子の大部分（約３／４）を占めており、μ粒子はπ中間子が崩壊して生成される。π中間子は宇宙から降ってきた陽子が大気中の原子核と‘強い相互作用’をすることによって生成される。π中間子には中性π中間子π^0と電荷π中間子$\pi^{\pm}$があり、は電荷π中間子は100％次のように崩壊する。

$$\pi^{+} \rightarrow \mu^{+} + \nu_{\mu}$$

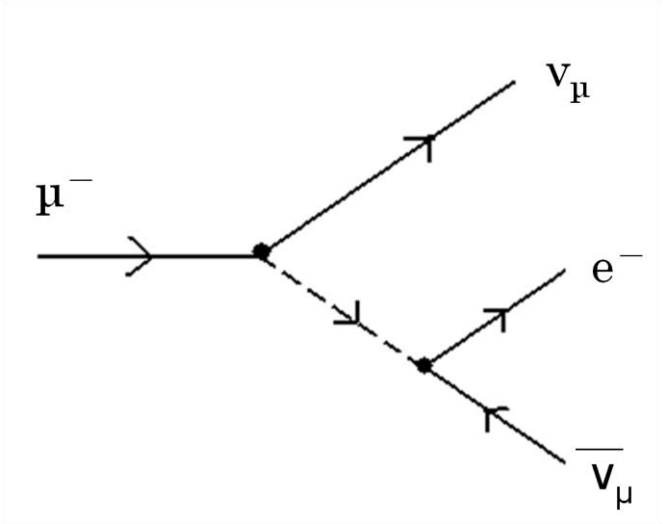
$$\pi^{-} \rightarrow \mu^{-} + \overline{\nu}_{\mu}$$

一方中性子は、電磁相互作用によって、ほぼ100％の確立で$\pi^0 \rightarrow \gamma\ \gamma$に崩壊する。

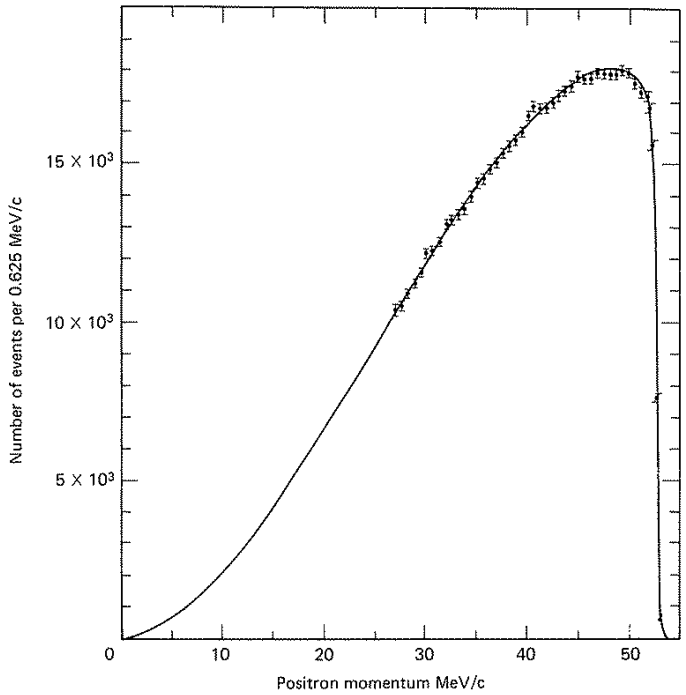
μ粒子は第2世代のレプトンで、質量105.7Mev/c²、電荷±１、スピン1/2の粒子である。そして、μ粒子は弱い相互作用によって、次のように崩壊する。

$$\mu^{-} \rightarrow e^{-} + \overline{\nu}_{e} + \nu_{\mu}$$

また、μ粒子の寿命はほぼ2.2μsecである。今回の実験では、この寿命を測定する。



電子のエネルギーが最小となるのは、μ粒子の静止系で電子が静止してつくられる場合で、最大となるのは２個のニュートリノの運動量がともに電子運動量の180°方向を向き電子に最大の反跳を与えるときである。μの静止系では、電子の質量を無視すると、以下の図を得る。



第3章 放射線計測の原理

放射線は我々の五感では感じることができない。そのため荷電粒子の電離作用と発光現象を用いて放射線を検出する。電荷粒子が通過し、電離作用を起こすと発光する物質をシンチレーターと呼ぶ。今回は、シンチレーターを用いて、その光を光電子増倍管で増幅させ、電気信号に変える方法で放射線を検出する。

３，１ 電離損失

荷電粒子が物質中を通過すると、入射荷電粒子と物質を構成する原子との相互作用によって、原子が電子と陽イオンに分離される。これを、原子の電離(Ionization)という。また、電離作用を起こさずに、原子や分子がエネルギーの高い状態(励起状態)になることもある。これを。原子、分子の励起(Excitation)と呼ぶ。

入射荷電粒子が物質を通過する時、物質中の電子と衝突し、電離や励起を繰り返しながらエネルギーの一部を失う。これを電離損失(Ionization loss)という。電離損失によって荷電粒子が失うエネルギーはBethe - Blochの式で表される。

$$\frac{dE}{dx} = 4\pi N_a r_e^2 m_e c^2 \rho \frac{Z}{A}\frac{z^2}{\beta^2}\left[\ln\left(\frac{2m\gamma^2 v^2 W_{max}}{I^2}\right) - 2\beta^2\right] (\mathrm{Mev/g/cm^2}) \qquad (3.1)$$

ここで、N_0はアボガドロ数、Ｉは電離ポテンシャル、Ｚは物質の原子番号、Ａは物質の原子量、ｖは入射粒子の速度、ｚは入射粒子の電荷、ｅは電子の電荷、ｍは電子の質量である。

電離損失 $\frac{dE}{dx}$ は入射荷電粒子の質量には依存しないが、速度$v = \beta c$には依存する。入射荷電粒子のエネルギーが低く、βが小さい時、Bethe - Blochの式は

$$\frac{dE}{dx} \propto \frac{1}{\beta^2} \qquad (3.2)$$

となる。入射粒子の運動量が大きくなると、電離損失は$1/\beta^2$に従って急激に減少し、最小値に達する。この領域での電離をMinimum ionizationという。電荷が同じ粒子ならば、粒子の質量が３倍になるちころで、最小値はほぼ同じ値をとる。

入射荷電粒子のエネルギーが高くなると、$\beta^2 \cong 1$となり、(3.1)式のlogの中の項が効くのでエネルギー損失は、$\log\gamma$でエネルギーが増加するにつれて上昇する。

$$\frac{dE}{dx} \propto \log\left[\frac{\beta^2}{Z\left(1-\beta^2\right)}\right] \tag{3.3}$$

$$\gamma = \frac{\beta^2}{\sqrt{1-\beta^2}} = \frac{E}{m}$$

電離・励起状態にある原子・分子が基底状態に遷移する時、二つの状態のエネルギーの差を光として放出する。この発光をシンチレーション(Sintillation)という。今回の実験では、シンチレーション光を用いて、入射粒子の電離損失を測定する。

３，２ 制動放射

電子は質量が小さいので、他の荷電粒子と衝突した際に、原子の励起やイオン化を行わない弾性衝突であっても、原子核の作る電場によって、速度の時間変化、方向の変化をかなり受け、加速度運動する。古典電磁気によると、加速度の二乗に比例するエネルギーを電磁波として放出する。加速度の大きさは物質内の荷電粒子の電荷Zに比例し、入射電子の質量に反比例するので、放出するエネルギーは$(Z/me)^2$に比例する。したがって、物質内でZの大きい原子核の存在が最も制動放射に効く。

また、制動放射によるエネルギー損失は入射電子のエネルギーに比例して増していく。そのため、この損失は電子が高速になってから有効となる。この理論式は次式のようになる。

$$-\frac{dE}{dx} = \frac{NEZ^2 r_e^2}{137}\left(4\ln\frac{183}{z^{\frac{1}{3}}} + \frac{2}{9}\right) \tag{3.4}$$

ここで、Nは物質の単位体積中の原子数（$N=N_0\rho/A$)、Eは入射電子のエネルギー、r_eは電子の古典半径（$r_e=e_0^2/m_ec^2=2.81\times10^{-13}$)、Aは物質の原子量、$N_0$はアボガドロ数（$6.02\times10^{23}$)、$\rho$は物質の密度である。

制動放射でエネルギーを失って、電子の始めのエネルギーの*1/e*になるまでに走る物質層の長さを放射長(radiation length)という。(3.4)式より、

$$-\frac{dE}{E} = \frac{dx}{x_0}$$

となるので、放射長X_0を求めることができる。

３，３　　CsIシンチレーターのエネルギー損失

CsIシンチレーターのエネルギーロスは

$$\Delta E_{(\mathrm{MeV})} = \frac{dE}{dx}_{\left(\frac{\mathrm{MeV}}{\mathrm{g/cm^2}}\right)} \times \rho_{(\mathrm{g/cm^2})} \times t_{(\mathrm{cm})}$$

で求めることができる。
ここで

$$\frac{dE}{dx} = 1.243$$

$$\rho = 4.53$$

であるのでCsIシンチレーターの１ｃｍあたりのエネルギー損失は

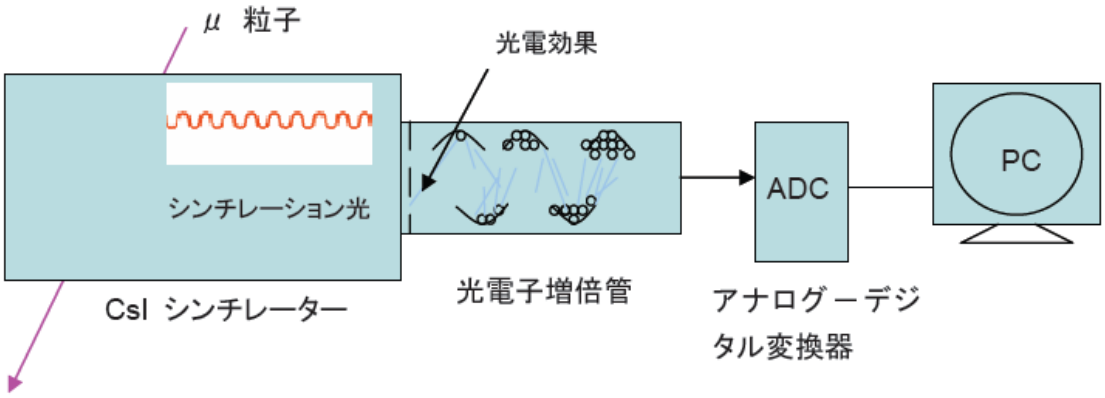
$$\Delta E = 1.243 \times 4.53 \times 1 = 5.63\mathrm{MeV}$$

第４章 測定原理

４，１ 計測原理

放射線を計測する際、放射線は五感では感じることができないため、粒子の動きを信号として計測できる形にしなければならない。そこで、荷電粒子が物質の中を通るとエネルギーを光として放出するという現象を利用して、その光を電気的信号に変換し計測を行った。

μ粒子がシンチレーター内を通ることにより光を放出する。その光は光電子増倍管の光電面に当たると、光電効果により電子を発生する。光電子増倍管でその電子を増幅させ、ADC(Analog Degital Converter)を用いてデジタル信号に変換し解析をおこなう。



４，２ μ粒子の寿命算出

μ粒子の寿命測定を行うにあたって、放射性崩壊の指数関数法則を用いる。今回の実験で用いた指数関数法則は以下のように導かれる。

今、それぞれの粒子が単位時間に崩壊する確率をλとする、独立な粒子の集合を考える。時間dtの間に崩壊する数dNは次式で与えられる。

$$dN = -\lambda N(t)dt \qquad (4.1)$$

ここで、N(t)はある時間tに存在する粒子の数である。（４，１）を積分する。N_0を時間t_0に存在する粒子数とすると、

$$\int_{N_0}^{N} \frac{dN}{N} = -\lambda \int_{t_0}^{t} dt$$

$$[\ln N]_{N_0}^{N} = -\lambda\,[t]_{t_0}^{t}$$

$$\ln N - \ln N_0 = -\lambda\,(t - t_0)$$

$$\ln \frac{N}{N_0} = -\lambda\,(t - t_0) \qquad (4.2)$$

t_0 =0の場合、(４，2) 式は通常の放射性崩壊の指数関数法則である次式に書き換えられる。

$$N = N_0 e^{-\lambda t} \qquad (4.3)$$

時間 t と t＋dt との間の無限に小さい時間間隔 dt の間に崩壊する粒子の数は、平均として(4.1) である。N_0個すべての粒子の生存時間の和Lは tN λ dt の t＝0から t＝∞までの積分であり、（４，4）式で与えられる。

$$L = \int_0^{\infty} tN\,\lambda\,dt$$

$$= \int_0^{\infty} tN_0\,\lambda\,e^{-\lambda t}\,dt$$

$$= N_0\,\lambda \left[\frac{1}{\lambda^2}\right]_0^{\infty}$$

$$= \frac{N_0}{\lambda} \qquad (4.4)$$

以上より、平均生存時間L/N_0（平均寿命時間 τ ）は(4.5)式で与えられる。

$$\frac{N_0}{L} = \tau = \frac{1}{\lambda} \Leftrightarrow \lambda = \frac{1}{\tau} \qquad (4.5)$$

これを（４，3）式に代入すると、

$$N = N_0 e^{-\frac{t}{\tau}} \qquad (4.6)$$

が得られる。
この式を用いてμ粒子の寿命を測定する。

μ粒子がシンチレーター内を通過すると光を出し、その信号は一つ山になる。一方、μ粒子がシンチレーター内で静止し、しばらくして崩壊すると、μ粒子のエネルギー損失と電子のエネルギー損失でその信号は二つ山になる。

この実験では一つ山のピークが来てからと二つ山のピークが来るまでの時間差が(4.6)式のｔであり、このｔを測定することにより（4.6）式からμ粒子の寿命τを算出できる。

4,3　プラスチックシンチレーター

ポリエスチレン等プラスチック中に有機発光物質が溶かし込まれている。そのため、形状の加工が簡単である。また、応答が数nsecのオーダーであり、シグナルの立ち上がりと立ち下がりが非常に速い(時間分解能が高い)一方、光量が無機シンチレーターに比べると少なく、エネルギーの等しい粒子に対する発光量にばらつきがでる(エネルギー分解能が低い)。時間分解能が高いことを利用し、今回の実験では、トリガーカウンターとして使用する。

Ｖモードは、信号の時間が ～１μsec程度である時に使うモードであることに対し、プラスチックシンチレーターの信号は ～10nsec程度で、非常に速い。
なので、ＶモードよりもＱモードが適している。

４，４ ADCについて

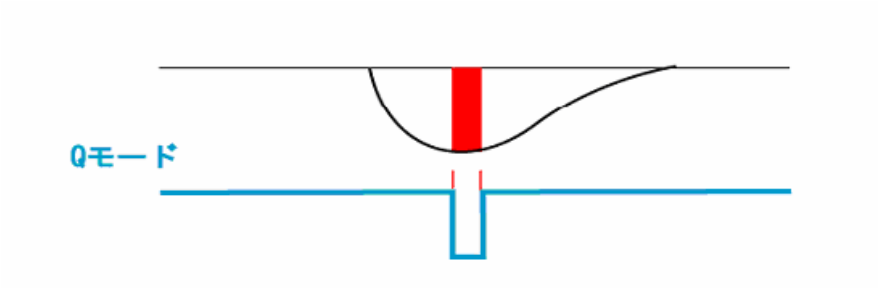
光電子増倍管で増幅したシンチレーターの信号をデジタル信号に変換するのがADC(Analog Degital変換回路)である。

今回の実験では、ADC(Qモード、Vモード、FADC)を用い、ＣｓＩシンチレーター内で崩壊したμ粒子から来る信号を数値化して解析を行った。

・Qモード(荷電積分型ADC)

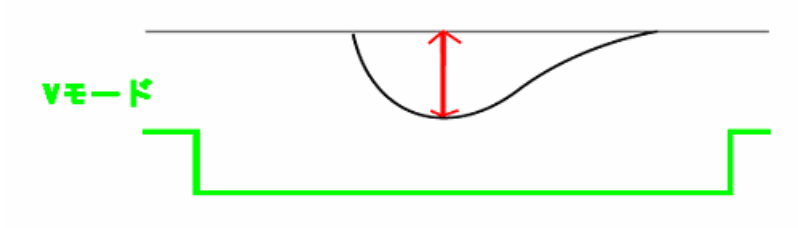
GATEパルスが持続している間の信号を積分するモードのこと。

測定量は$Q=\int_{\Delta t} i\,dt$ となる。



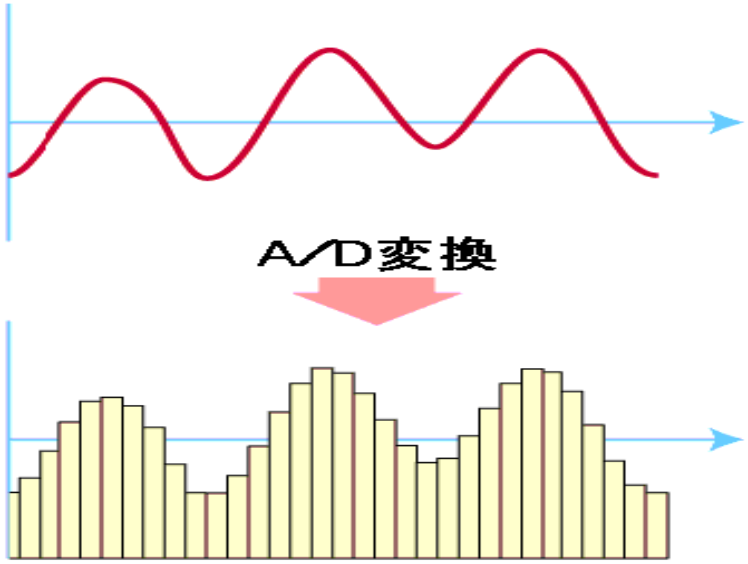
・Vモード(ピークホール型ADC)

GATEパルスが持続している間のピークの電圧を測定するモード

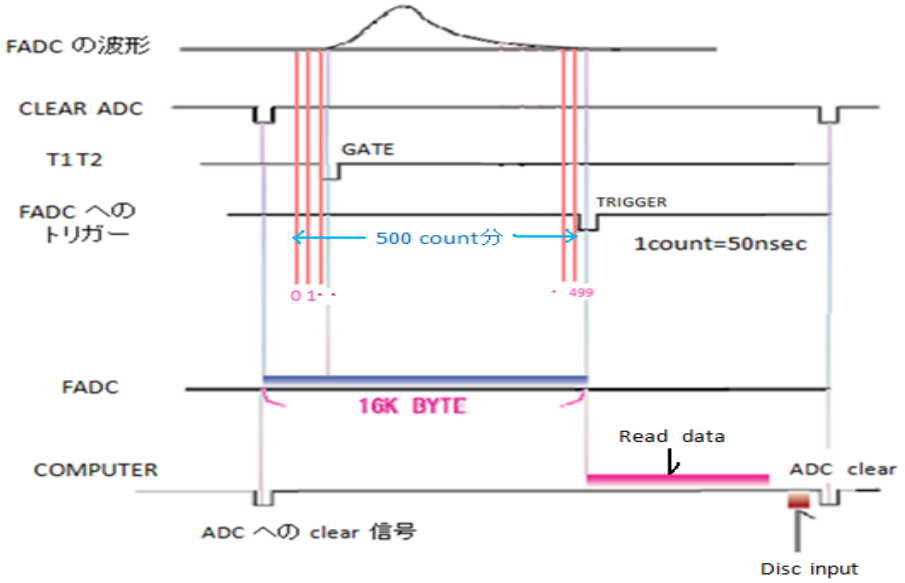


・FADC(フラッシュADC)

時間ごとに波高を測定するモードである。FADCは通常のADCに比べて、A-D変換速度が極めて速いので、時系列データを読み込むことが可能となり、 信号の波形を細部まで観測することができる。



FADCにはstartモードとstopモードがあり、FADCのモジュール内で切り換えることができる。今回の実験ではstopモードを使用した。
FADCの波形をコンピューターに取り込むまでの過程を以下で説明する。
FADCの波形が来る前にまず、ADCのLAMClearをする。波形が来たらシンチレーションカウンターによりGATEを開く。次にFADCのstopモードにより波形の終端でトリガーをかける。その終端からさかのぼり前から設定した時間ごとにシグナルの高さを500countプロットする。ここで、1 countを50nsecと設定した。最後にプロットしたFADCの波形をコンピューターに読み込み、読み込み終了後、再びADCのLAMClearをして、次の波形のプロットに備える。

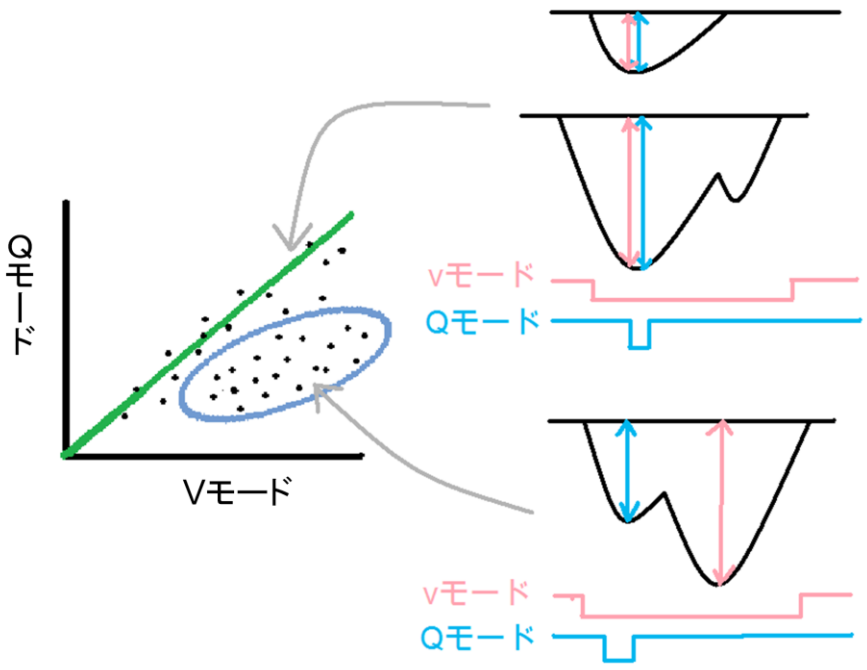


4,5　Qモード、Vモードで見るμ粒子の崩壊

μ粒子が崩壊せず通過すると波形が一つ山になるので、QモードとVモードで同じ波形を測定することになる。なので、μ粒子が崩壊することなくシンチレーターを通過する時のエネルギー損失は、Qモード、Vモードで二次元プロットすると比例関係になる。

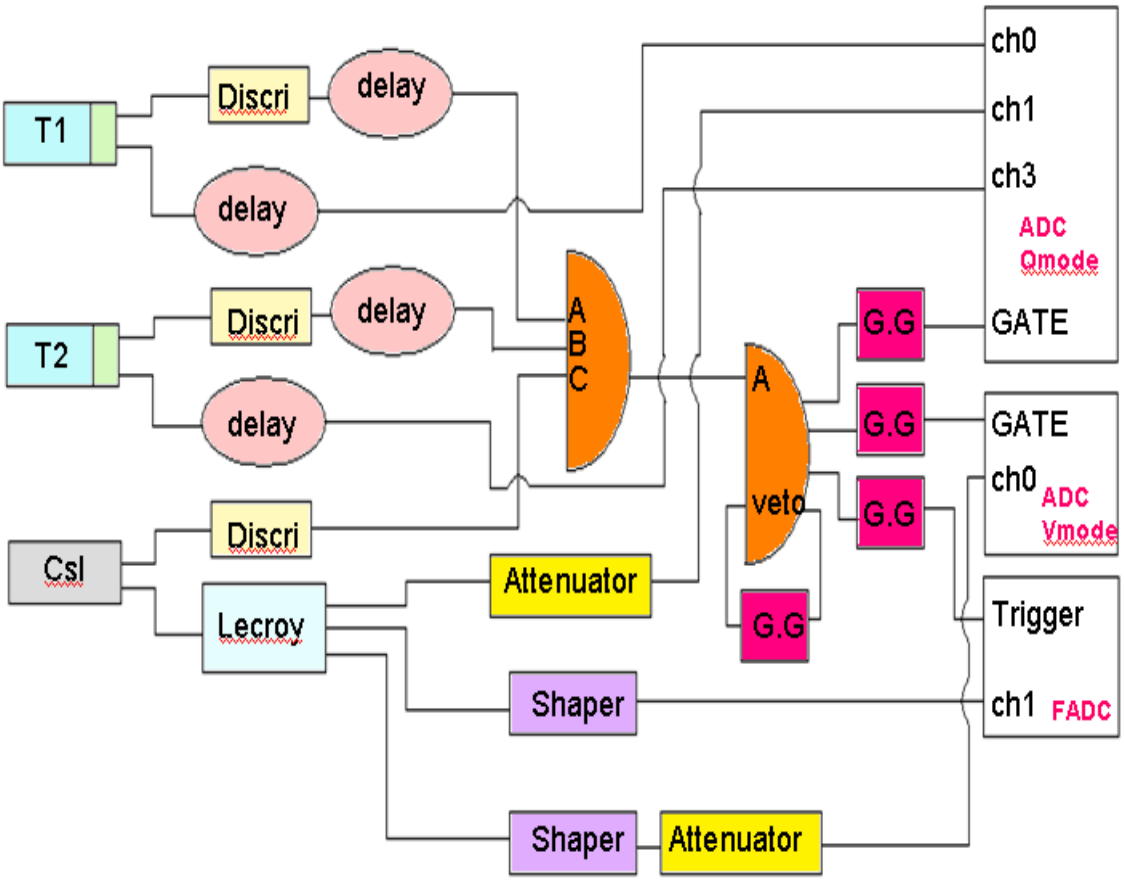
μ粒子が崩壊すると波形が二つ山になる。このとき、前の山が後ろの山よりも高い場合も比例関係となる。

一方、前の山が後ろの山よりも低い場合、QモードとVモードで異なった波形を測定することになる。したがって、μ粒子が崩壊すると、QモードとVモードの比例関係からは外れるものもあり、この比例関係から外れたイベントがμ粒子の崩壊である可能性がある。



第5章 セットアップ

5,1エレクトロニクスのセットアップ



今回の実験では上図のような配線を組み、イベントを収集した。それぞれ用いた装置について説明する。

(1)シンチレーションカウンター(scintilation counter)

シンチレーションカウンターは放射線検出器の一種である。シンチレーターは荷電粒子が通過する時にその粒子が失うエネルギーを光エネルギーに変換する物質である。シンチレーターに粒子が入射すると、ある波長の光が放出される。この光を光電子増倍管で電気信号に変換すると、放射線検出器として働く。

シンチレーターには、有機シンチレーターと無機シンチレーターがあり、それぞれ発光機構に違いがある。今回の実験では、前者のプラスチックシンチレーターと後者のCsIシンチレーターの二種類を用いる。

・プラスチックシンチレーター（T1,T2,）

ポリエスチレン等プラスチック中に有機発光物質が溶かし込まれている。そのため、形状の加工が簡単である。また、応答が数nsecのオーダーであり、シグナルの立ち上がりと立ち下がりが非常に速い(時間分解能が高い)一方、光量が無機シンチレーターに比べると少なく、エネルギーの等しい粒子に対する発光量にばらつきがｓる(エネルギー分解能が低い)。時間分解能が高いことを利用し、今回の実験では、トリガーカウンターとして、使用する。

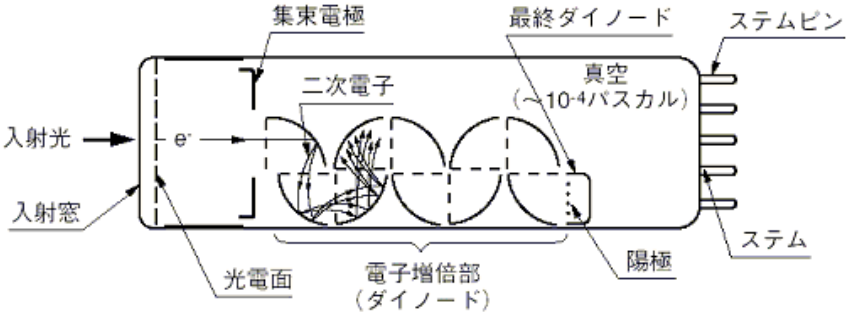
・CsI(Tl)シンチレーター

入射粒子のエネルギーに応じて光り、そのエネルギー損失が１つの数値として光電子増倍管で増幅され、コンピューターで読み込まれる。プラスチックシンチレーターに比べてエネルギー分解能が高ので、トリガーカウンターの近くに置き、粒子のエネルギーを復元する。

(2)光電子増倍管(Photomultiplier)

光電効果を利用して光エネルギーを電気エネルギーに変換する光電管を基本とし、電流増幅、つまりは電子増幅機能を付けた高感度光検出器である。

光電面に光が入射すると光電効果によって電子が飛び出す。真空管の中で電極に高電圧をかけて電子を加速すると。飛び出した電子がダイノード(Dynode)に衝突し、ダイノードがいくつかの電子を放出する。この放出された電子は、次のダイノードとの間にかけられた電場によって加速され、次のダイノードに衝突し、より多くの電子を放出される。これを繰り返すことで電子を増幅する。この電子がアノード(Anode)に集まり、電気信号として出力される。



（3)ディスクリミネーター(discriminator)

あらかじめ設定しておいたthreshold(しきい値)よりも大きなシグナルが入力された時、方形(パルス)波を出力する装置。放射線検出器からのシグナルの波高分析やタイミングパルスの発生などに使用されるが、今回の実験ではthreshold以下のノイズを除去するために用いた。パルスは - 0.7V、140nsecとNIM規格で決められている。

（4)コインシデンス(coincidence)

コインシデンスは複数のパルスが時間的に重複して入力された時にパルスを出力する。今回の実験では、T1,T2同時に信号がきた時に信号がでるようにした。

（5)ゲート・ジェネレーター(Gate and delay Generators)

入力信号に対して、最小で110nsecから最大で4.0secまでの範囲でdelayさせたり、パルス幅を調整するためのものである。

（6)アテニュエーター（Attenuator)

ADCが読み取れる範囲にするため、入力信号を減衰させる装置。単位はdB。

$$dB=-20\log_{10}\frac{V_{in}}{V_{out}}$$

V_{in}；入力信号の波高
V_{in}：出力信号の波高

（7)シェイパー（Shaper)

波形整形をするためのもので、FADCが読み込みやすいように立ち上がり、立ち下がりを長くする。

（8)ADC（アナログ・デジタル変換回路)

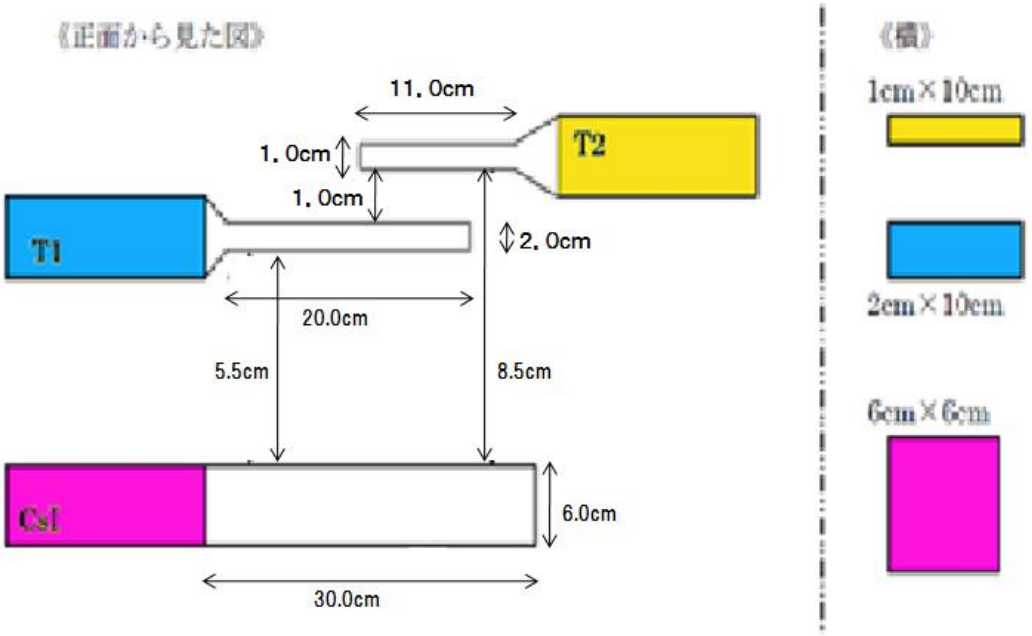
アナログ電気信号をデジタル電気回路に変換する電子回路のこと。今回の実験ではQモード、Vモード、FADCを用いる。

（9）エレクトロニクスの設定

今回の実験では、それぞれの設定を以下のように決定し、測定を行った。

	HV（V）	Threshold（mV）	Width（nsec）
T1	－１８００	１５０	１６０
T2	—２０１０	１５０	１６０
CsI	－２２００	１５０	１６０

５，２ シンチレーションカウンターのセットアップ

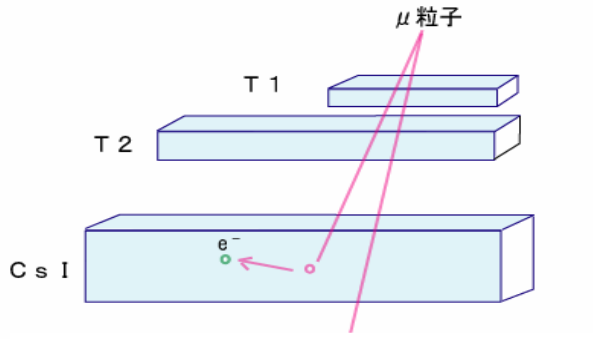


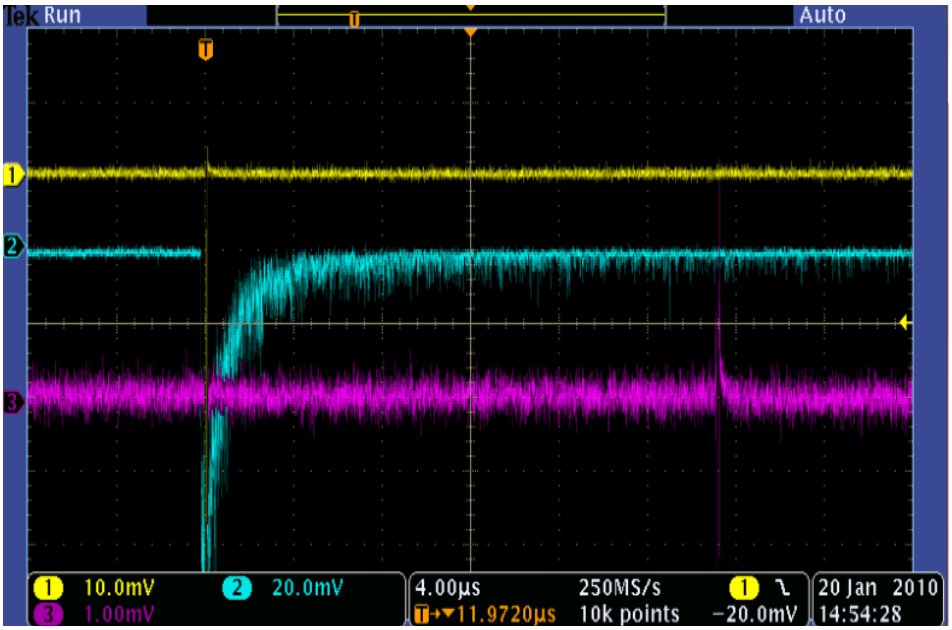
今回は上図のように、プラスチックシンチレーター(T1,T2)とCsIシンチレーターを配置し、実験を行った。

（1）トリガーカウンター

宇宙線であるμ粒子は宇宙から絶え間なく降り注いでいる。CsI(Tl)にはあらゆる方向からμ粒子が入射してくる。そのため、CsI(Tl)中でμ粒子が崩壊したかどうかが難しい。そこで、T1,T2をＣｓI(Tl)の上に置き、CsI(Tl)にHitの条件を与える。そうすることによって、余分なデータが削られ、必要とするデータのみを取り出すことが可能となる。この作業をするのがトリガーカウンターである。

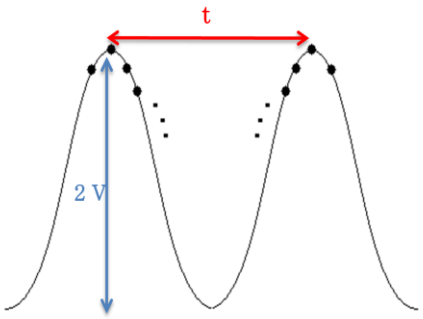
今回の実験では、Ｔ１,T2をトリガーカウンターとして使用し、T1,T2,CsIのコインシデンスをONにして測定する。

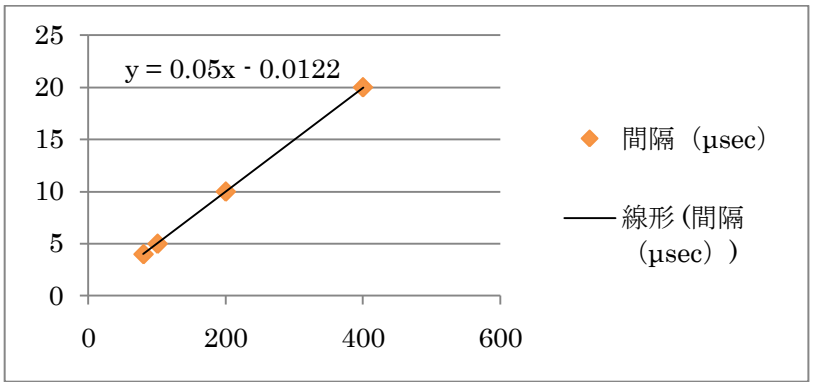




５，３ FADCのサンプリング周波数

μ粒子の寿命を測定するために、FADCの時間間隔を測定する。まず発信機からの信号をｔ＝４、５、10、20［μsec］と順に変化させてＦＡＤＣに入力し、10eventづつ測定する。こうしてＦＡＤＣがどのような時間間隔でデータを収集しているのか(サンプリング周波数)がわかる。本実験では傾きが0.05±0.0122μsecなので、サンプリング周波数は50nsecと得られた。




25
20
15
10
5
0
y = 0.05x - 0.0122
0
200
400
600
間隔（μsec）
線形 (間隔（μsec）)

第６章 CAMACと
データ収集プログラム

６，１ CAMAC (computer aided measurements and control)

CAMACはモジュール化されたデータを処理するシステムで、世界中のほとんどの素粒子・原子核研究室やたくさんの工業現場で使用されている。これはU.SのNIMとヨーロッパのESONE委員会のジョイントで提供された。

CAMACは計算機周辺でのデジタル化された情報の処理を各機能ごとにモジュール化して行えるようにできている。すなわち、実験装置など、外からの情報はプラングイン・ユニットまたはモジュールの画面パネルからコネクターを通して入り込む。この情報はプラングインの中で処理されると、裏面のプリント基板エッジを利用したコネクターでクレートと呼ばれるプラングインを収容する箱の裏側の配線(データウェイ)とつながれる。このデータウェイはクレートコントローラーが制御するが、たいていはクレートコントローラー自身が計算機の指示に従って制御するようになっている。クレートコントローラーはたいてい小型計算機とCAMACのインターフェースを兼ねる。プラングイン・ユニットは回路配線に使われたプリント基板自身のエッジが86ピンのコネクターとなって飛び出しており、クレートに挿し込むと自動的にクレート裏側のコネクターを通してCAMACデータウェイと接続され、電源やデータの受け渡し、制御信号の受け渡しがされるようになっている。

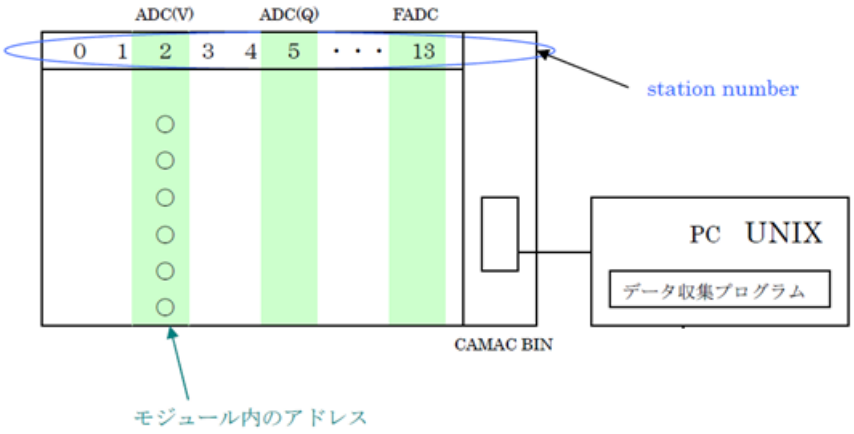
CAMACの規則は、アドレスの指定であり、CNAFで指定する。

ここで、Cはcrate controlerの数。今、crateは1つだけ使っているので、C=1。Nはstation number、Aはsubadress（モジュール内のアドレス）ADCにおいては12個の信号を読める。Fはfunction（コンピュータからの命令）である。

ここでＬＡＭとはLook At Meの略で、モジュールからデータを読出し可となったことをコンピューターに知らせる信号のことである。

代表的な命令の例を以下に示す。

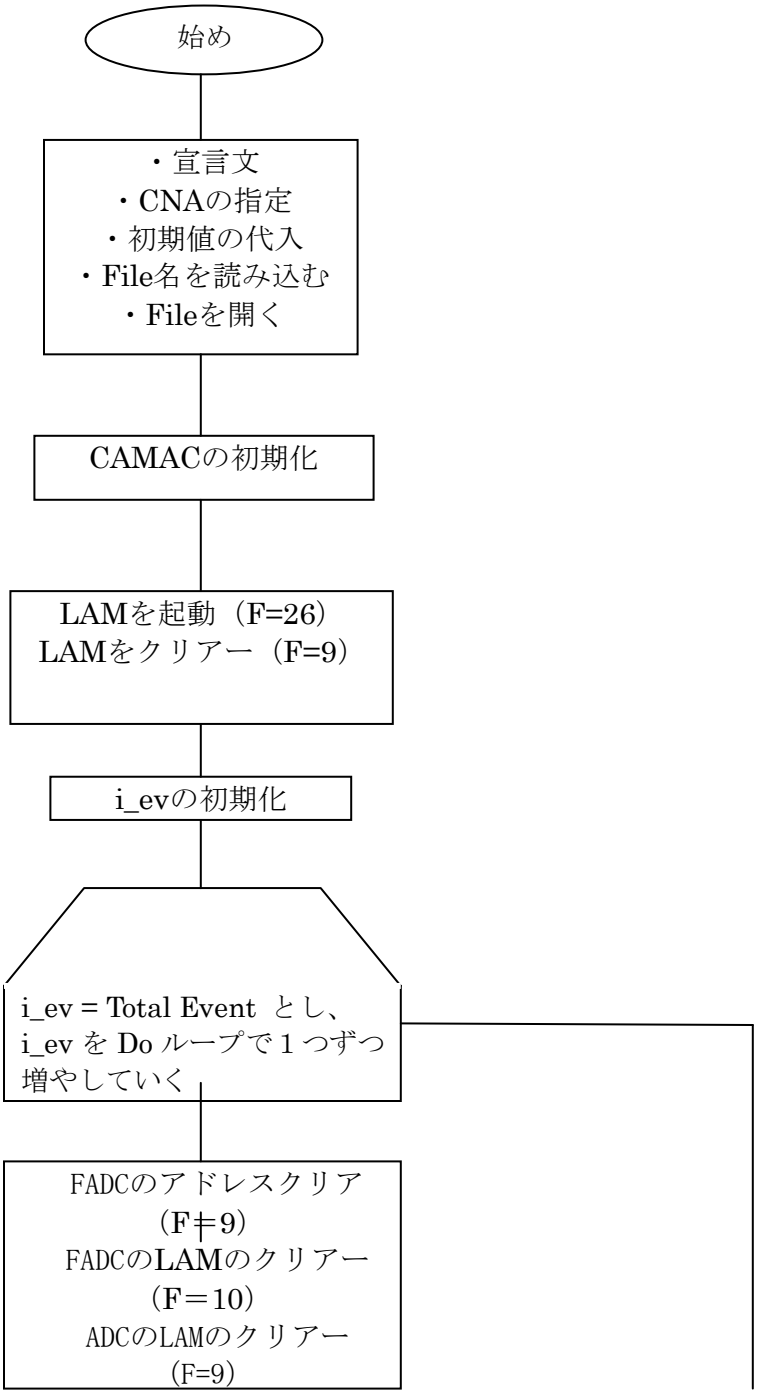
F(0),F(2)；Read Data
F(8)；Test LAM
F(9)；Clear LAM
F(24)；Disable LAM
F(26)；Enable LAM

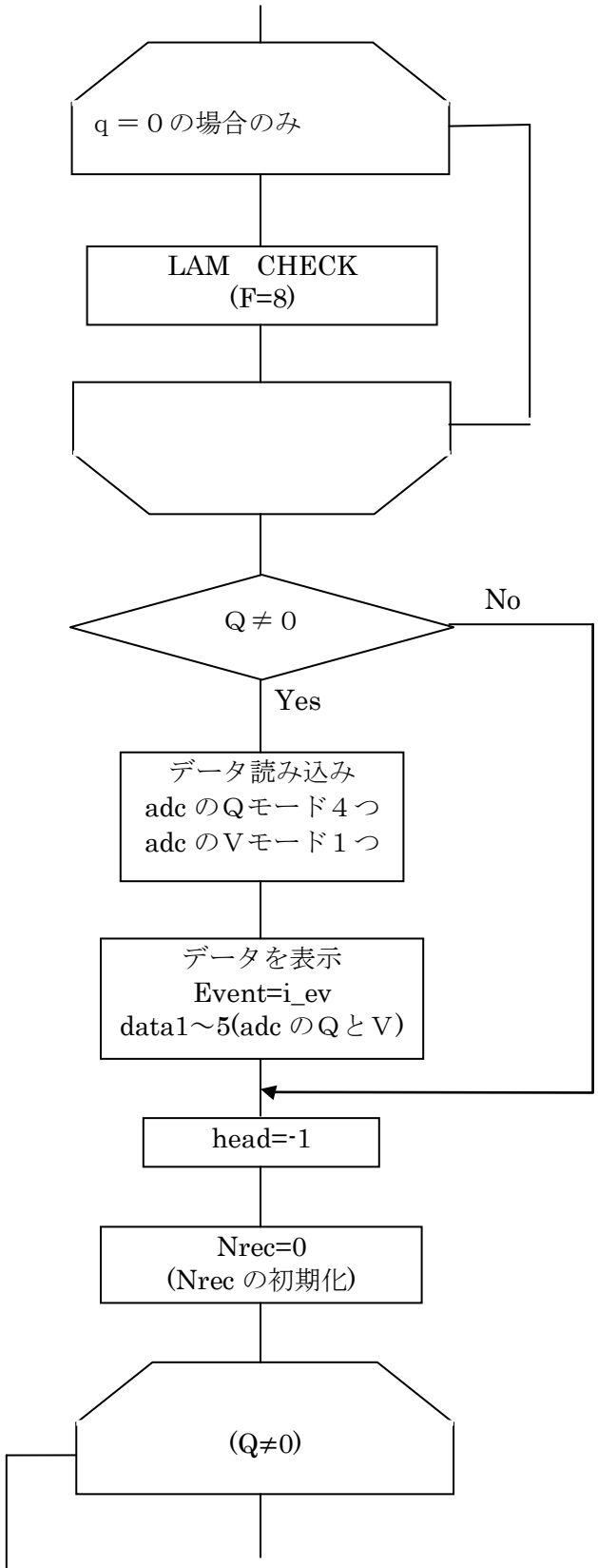

ADC(V)
ADC(Q)
FADC
0 1 2 3 4 5 ・・・ 13
station number
PC UNIX
データ収集プログラム
CAMAC BIN
モジュール内のアドレス

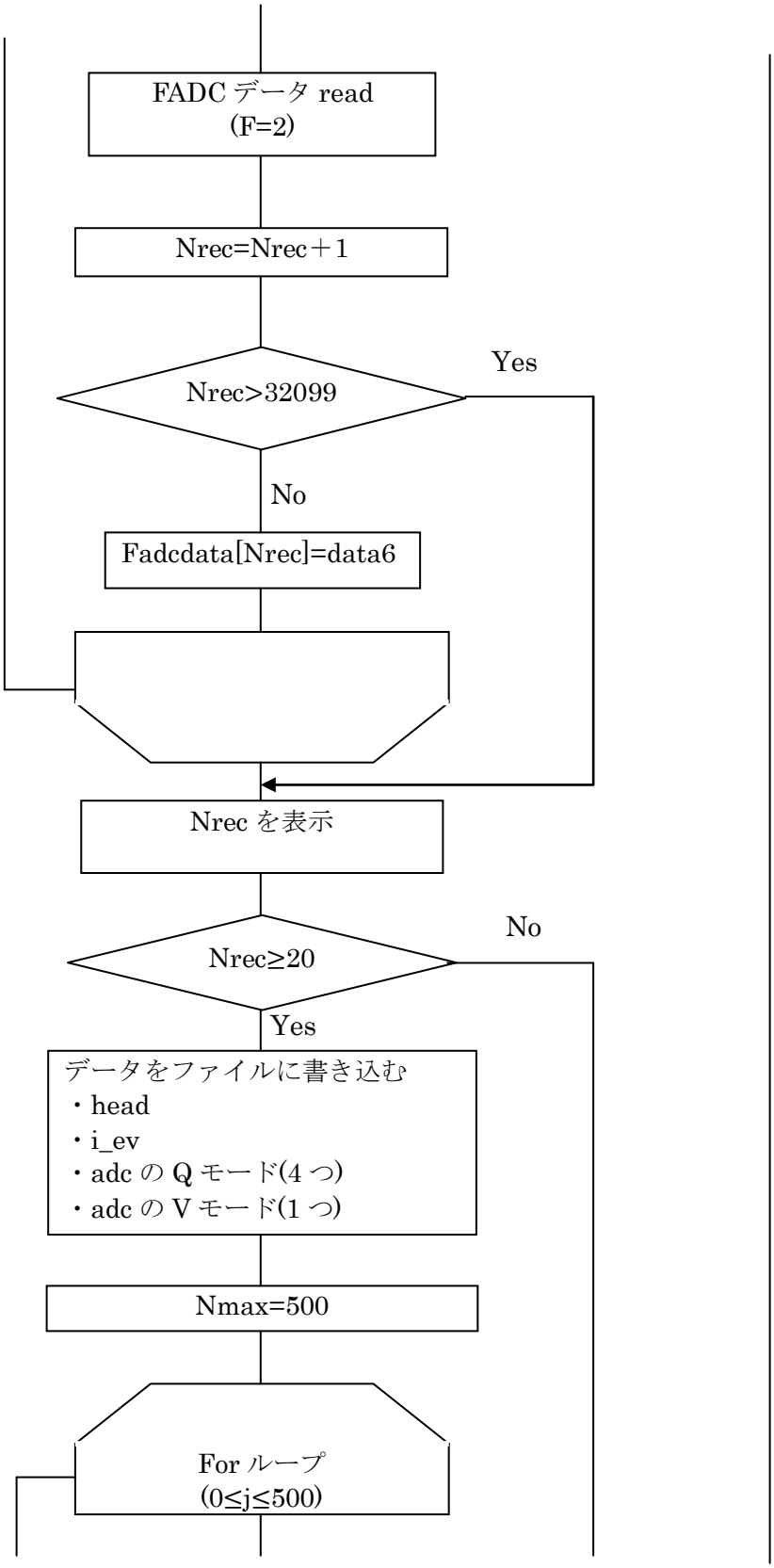
６，２ データ収集プログラム

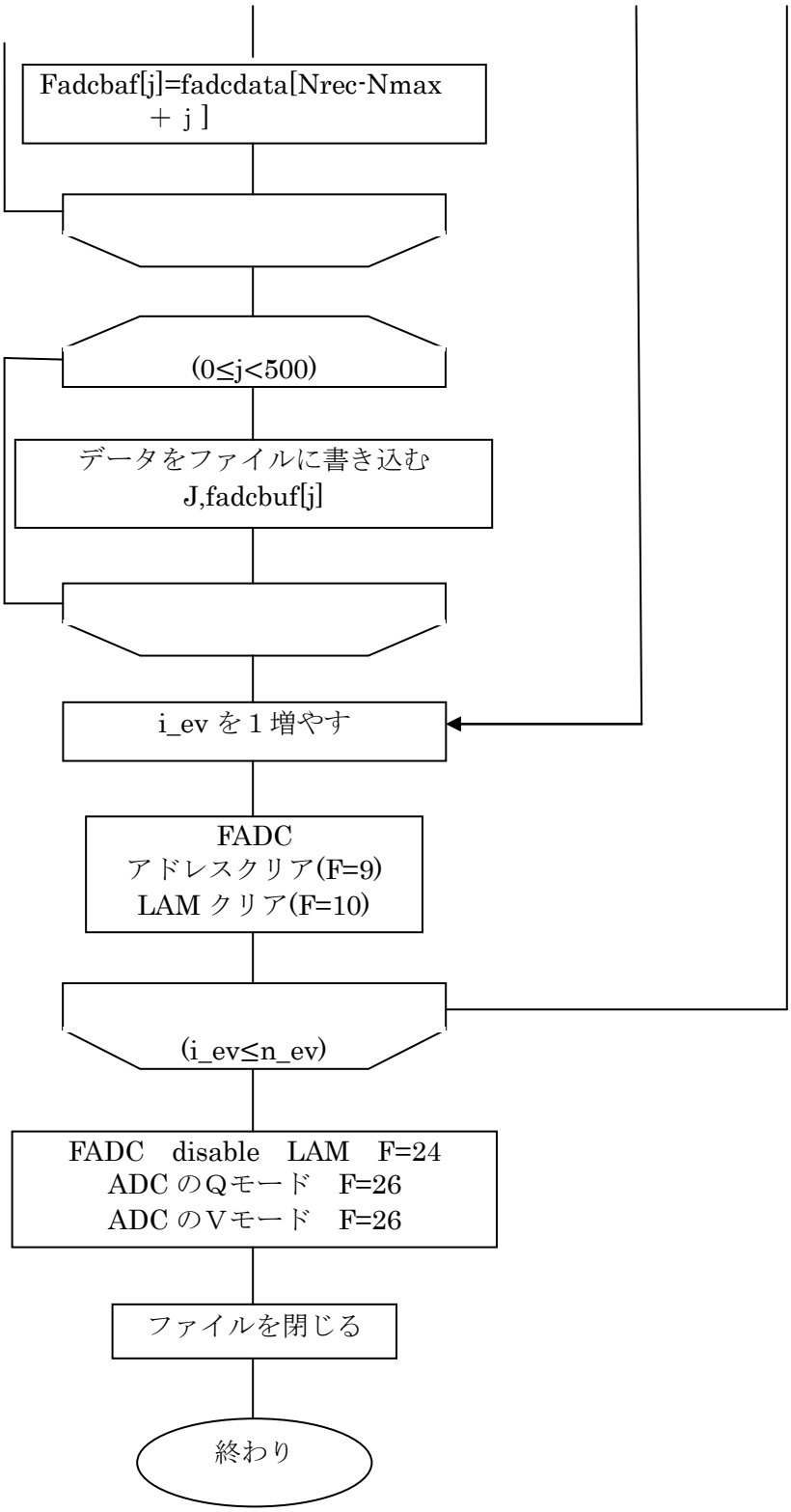
今回の実験では、CAMACからのデータ収集プログラムの言語にC言語を用いる。C言語を用いて、収集するイベントの個数を指定し、データを収集した。プログラムのフローチャートを以下に示す。

＜フローチャート＞

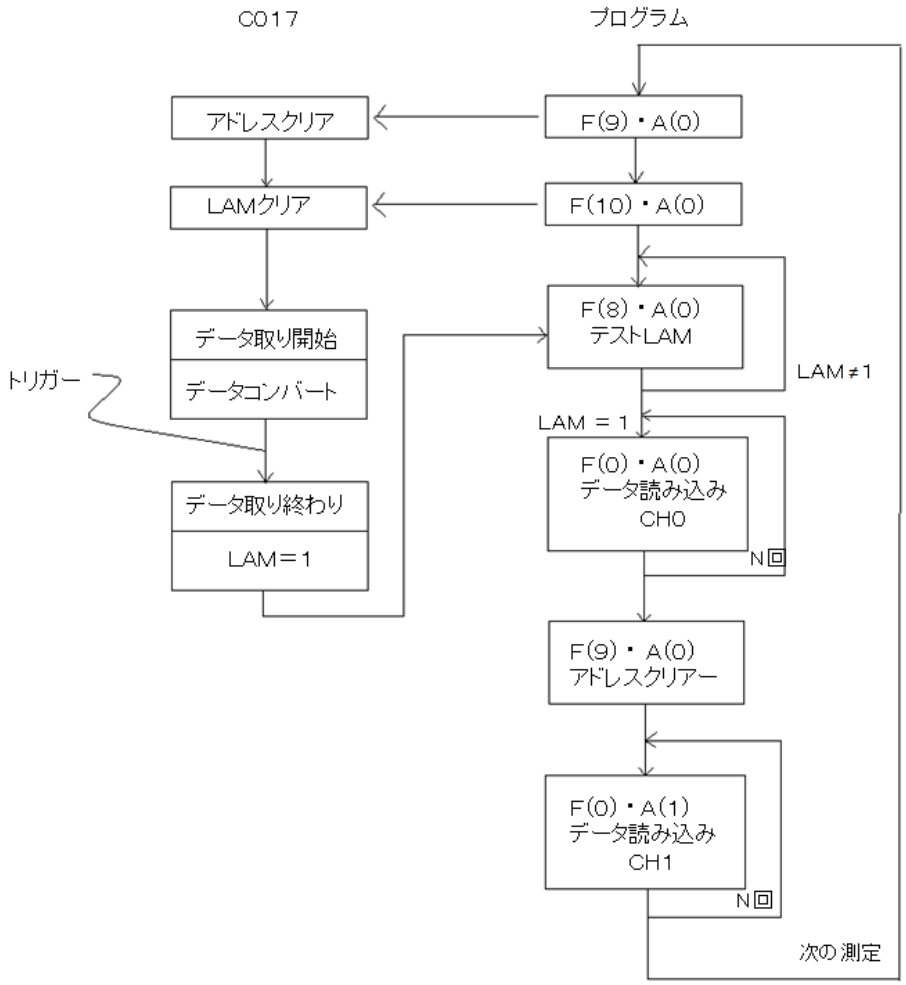



q ＝ 0 の場合のみ
LAM　CHECK
(F=8)
Q ≠ 0
No
Yes
データ読み込み
adc のＱモード４つ
adc のＶモード１つ
データを表示
Event=i_ev
data1～5(adc のＱとＶ)
head=-1
Nrec=0
(Nrec の初期化)
(Q≠0)


FADC データ read
(F=2)
Nrec=Nrec＋1
Nrec>32099
Yes
No
Fadcdata[Nrec]=data6
Nrec を表示
Nrec≥20
No
Yes
データをファイルに書き込む
・head
・i_ev
・adc の Q モード(4 つ)
・adc の V モード(1 つ)
Nmax=500
For ループ
(0≤j≤500)


Fadcbaf[j]=fadcdata[Nrec-Nmax
＋ j]
(0≤j<500)
データをファイルに書き込む
J,fadcbuf[j]
i_ev を 1 増やす
FADC
アドレスクリア(F=9)
LAM クリア(F=10)
(i_ev≤n_ev)
FADC　disable　LAM　F=24
ADC のQモード　F=26
ADC のVモード　F=26
ファイルを閉じる
終わり

＜FADCのstopモードのフローチャート＞



Ｆ（０）・Ａ（０）　　：ＣＨ０　データリード
　　データを読み込むと内部アドレスはインクリミメントする。
Ｆ（０）・Ａ（１）　　：ＣＨ１　データリード
　　データを読み込むと内部アドレスはインクリミメントする。
Ｆ（８）・Ａ（０又は１）：テストＬＡＭ
　　データの取り込みが終わるとＱ＝０となる。
Ｆ（９）・Ａ（０又は１）：アドレスのクリアー
　　内部のメモリーのアドレスをクリアーする。
Ｆ(10)・Ａ（０又は１）：ＬＡＭクリアー
　　ＬＡＭをクリアーし次のデータ取りに備える。

第７章　二つ山を探すプログラム

UNIXコンピュータでデータ収集し、データ解析を行った。ミュー粒子崩壊の場合FADCの波形が二つ山になるので、この二つ山を探すプログラムを作成した。

静止したμ粒子が、CsIシンチレーター中で崩壊したことによる代表的な二つ山の信号を図１に示す。

以下、多くの一つ山の信号の中から自動的に二つ山の信号を探し出すプログラムの中味について説明する。また、プログラムコードを論文の付録２につけておく。

イベントナンバー、ADCのＱモード４つ、Ｖモード１つの値を読み込む。

１イベントごとに５００word読み込む。

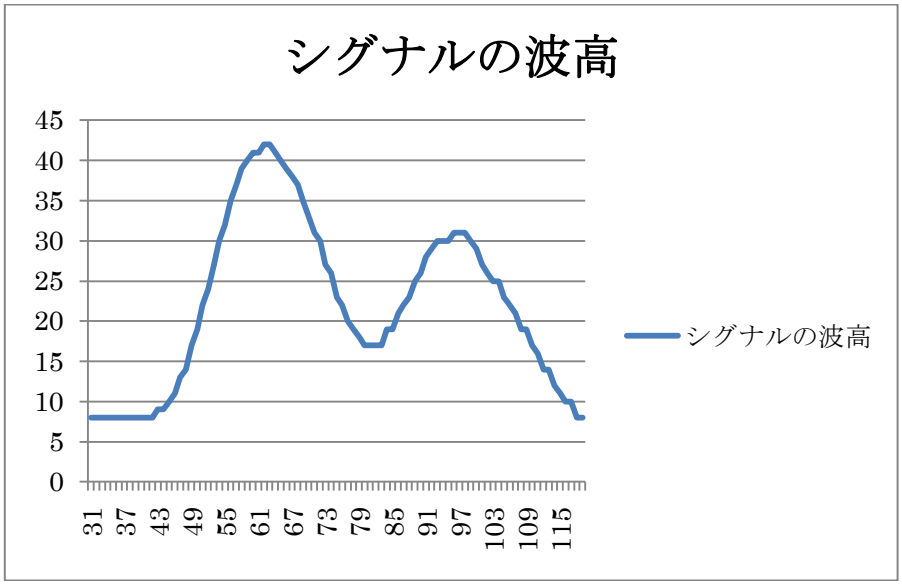


図１

シグナルの傾きを求めるために、微分する。
プログラムでは、j 番目のbinを degi_diff[j]、j －４番目のbinを degi_diff[j-4] とし、その差を dif2[j] とする。

例
dif2[４] = degi_diff[４] – degi_diff[０]

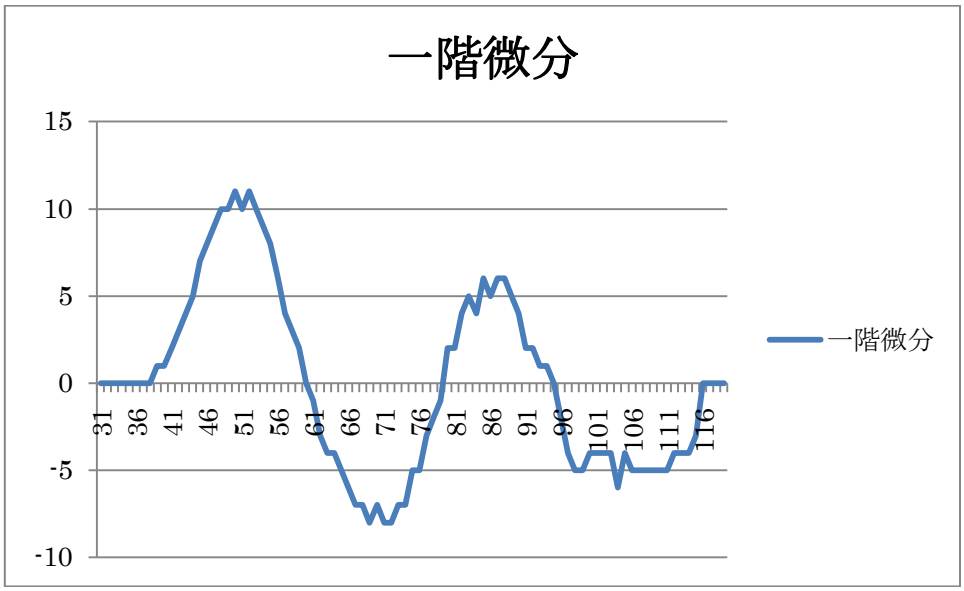


図２

シグナルの傾きをデジタル化するために
微分した値が３以上の場合１とする。
微分した値が-4以下の場合-1とする。
微分した値が-4より大きく3より小さい場合０とする。

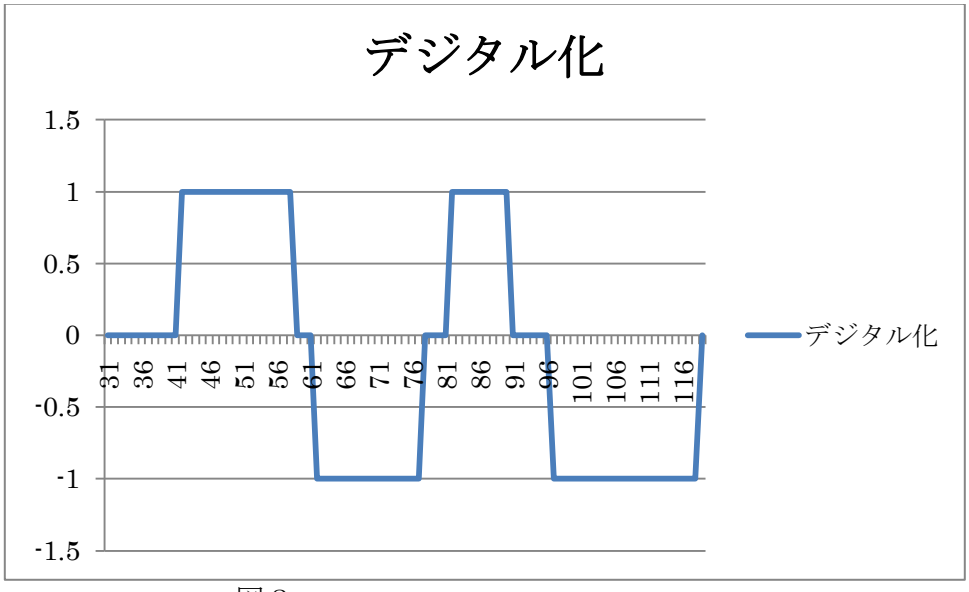


図３

デジタル化した値をさらに微分する。

例
dif2[１]=degi_diff[１]-degi_diff[０]

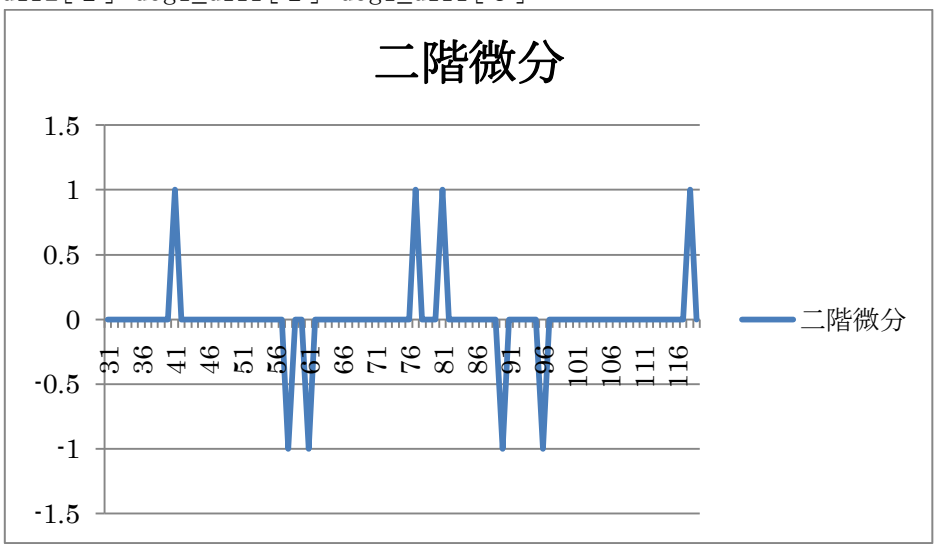


図４

-1が２回続いたときにのみ山であるとみなせるので、値を１ずつ加える。

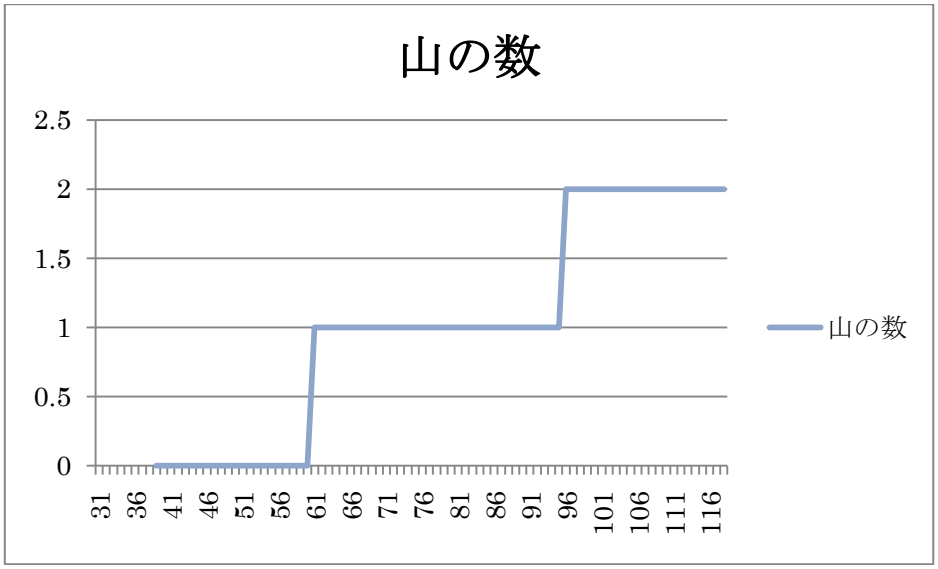


図５

ここでこの値が１となったときが一つ目の山のピーク位置となり、この値が２となったときが二つ目の山のピーク位置となる。
よって、二つの山の距離を求めるために、二つ目のピーク位置から一つ目のピーク位置をひく。

① ２つ山の距離が１０以上
② ２つ山である、かつ山の高さが２５５より小さい

① ・②の条件がそろったときのみ『イベント番号』を表示する。

第８章 データ解析

８，１ 全データ

今回の実験で得られたイベントは

Event数	
データ	843929　イベント
二つ山崩壊と考えられるEvent	1043 イベント

μ粒子の崩壊と考えられる　　Eventを用いて寿命を測定する。

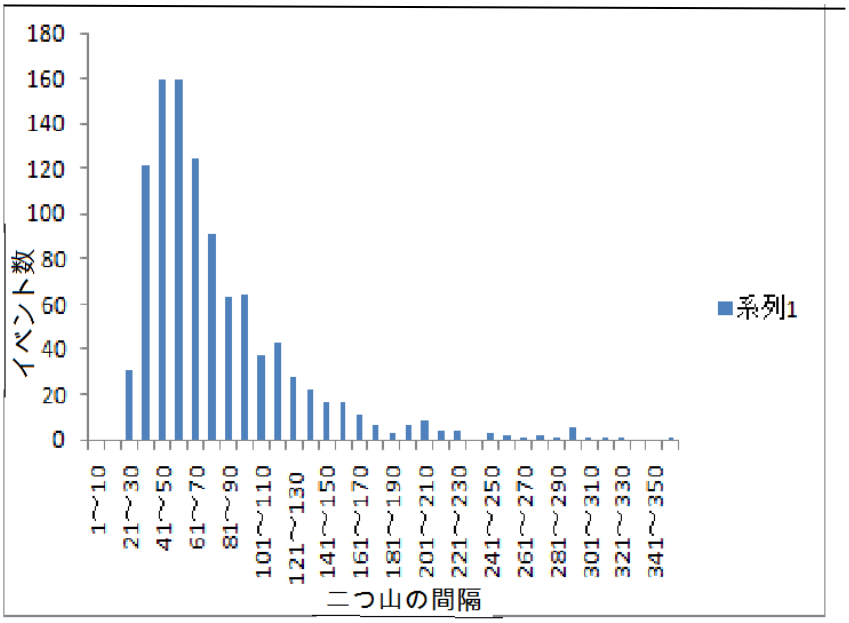
８，２ μ粒子のEvent選別

（１） 寿命は、　　$y = P_1 \exp\left(\frac{P_2 - x}{P_3}\right)$　　（８，１）

という式で表される。j2の値をEvent数ごとに選別するとこの形になることが予想される。ここで(８,1)式のP3の値が寿命に対応しているので、P3にサンプリング周波数の50nsecをかけると寿命を算出できる。

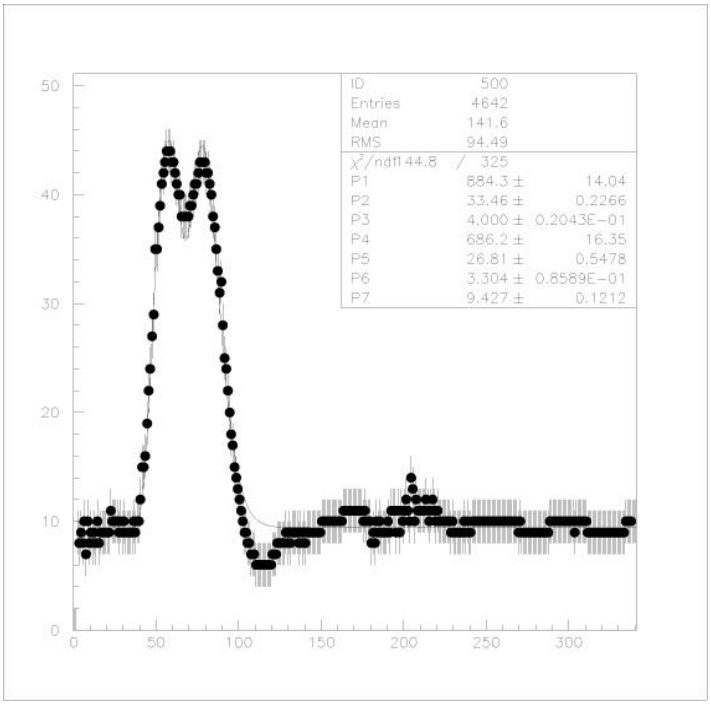
$\tau = P3 \times 50\text{nsec}$

今回の実験のデータを用いると次のようなグラフになる。

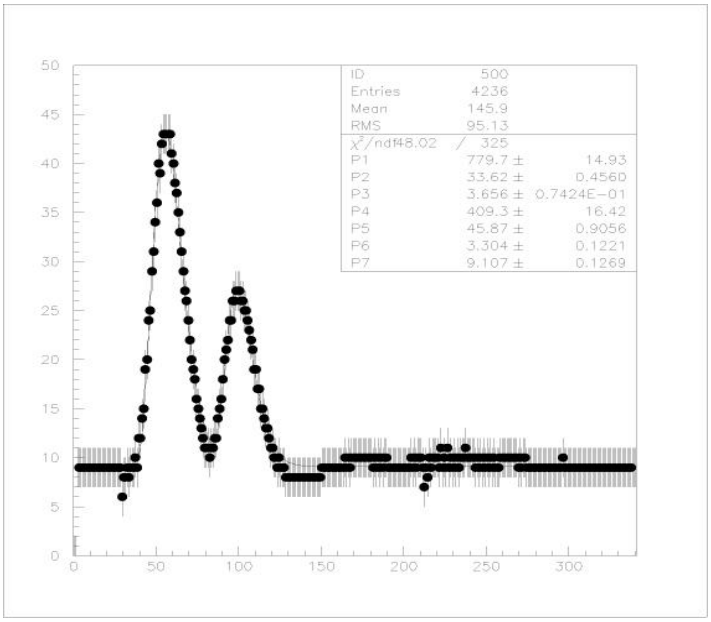


実際のシグナルの波高例を以下に示す。

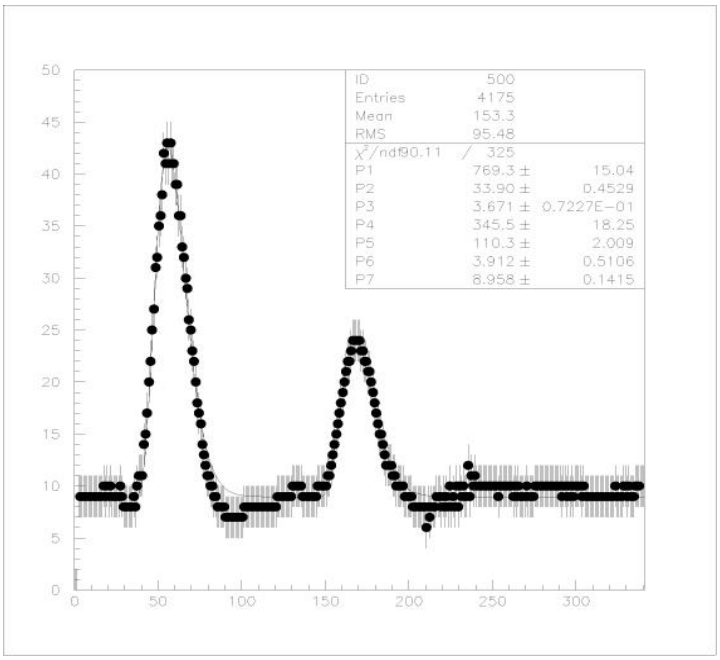
例１：間隔が２６



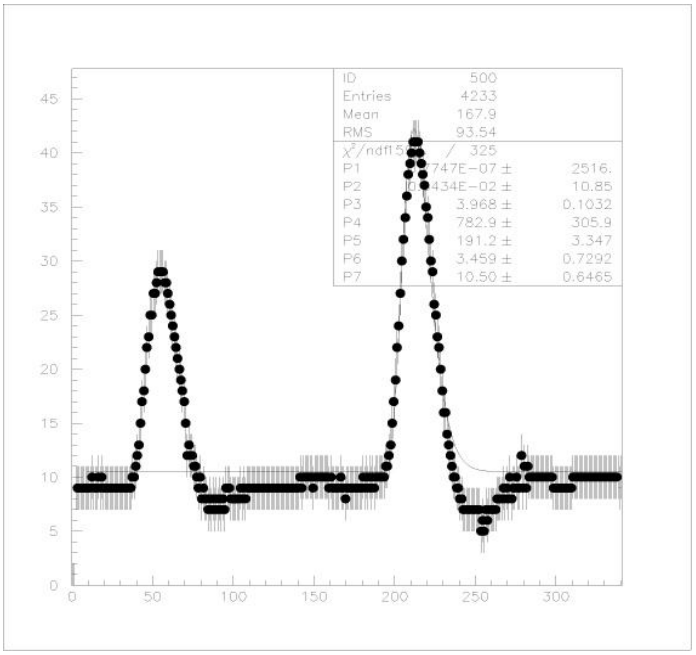
例２：間隔が４５



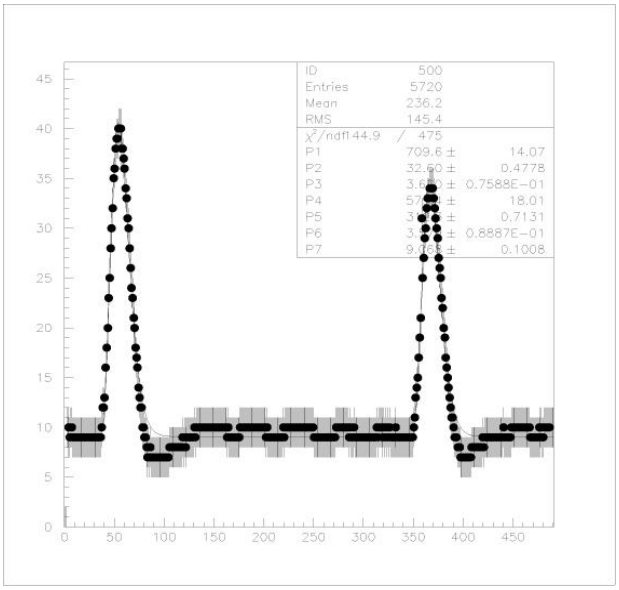
例３：間隔が１１０



例４：間隔が１９１

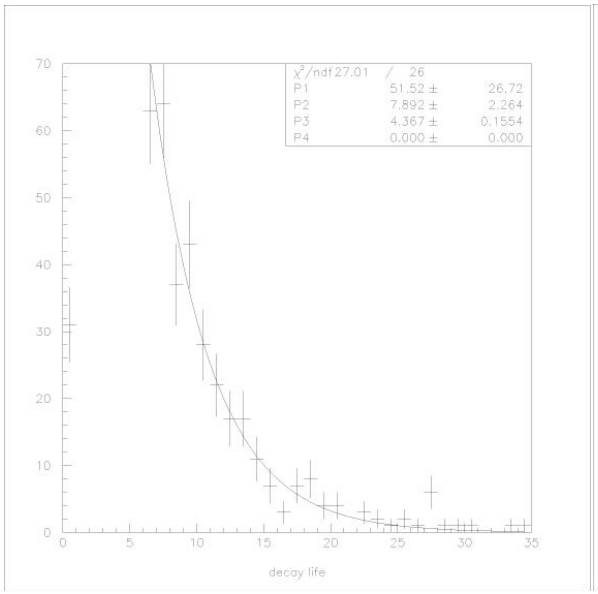


例５：間隔が３１２

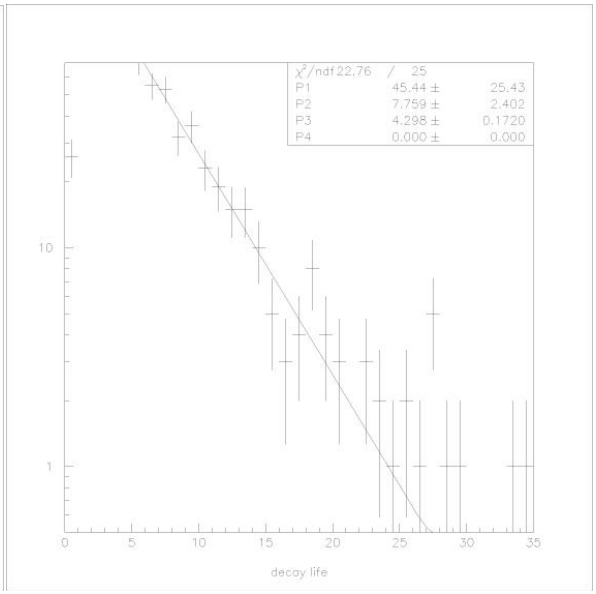


（２） このグラフを(８， １)式でFitすると、下のようになる。

Fitの範囲を３から３５とする。



ヒストグラム



logスケール

ここではP_3の値が 4.367±0.1554 なので、これにサンプリング周波数の50nsecをかけると、

$\tau = P_3 \times 50\text{nsec}$

$\fallingdotseq 2.18 \pm 0.08 \ \mu\text{sec}$

（3）FITの範囲

FITの範囲を変えることによってP_3の値も変わってくる。
いまこの範囲を変えて、それぞれの寿命を求めてみた。
ここでは１に近いほど精度が良いといえる。

範囲	χ^2検定	寿命[μ sec]
３～１８	１．２６	２．１０±０．０５
３～２３	１．１４	２．１５±０．０８
３～２８	１．１５	２．１７±０．０８
３～３３	１．０６	２．１８±０．０８
３～３５	１．０４	２．１８±０．０８
４～１８	０．９２	１．９８±０．０９
４～２３	０．９５	２．０５±０．０９
４～２８	１．０３	２．０７±０．０９
４～３３	０．９７	２．０９±０．０９
４～３５	０．９５	２．０９±０．０９

第９章 まとめ

９，１ μ粒子の寿命

今回算出したμ粒子の寿命は2.18±0.08μsecである。
ここで±0.08は寿命の誤差を表している。

９，２ 考察

Particle Data Bookに記載されているμ粒子の寿命は 2.197019±0. 000021μsec
であるので、今回の実験結果は良く一致しているといえる。

今回の実験ではデータをイベントごとに読み込んでいたため、データ解析をするのに時間を要したが、今後の課題としていくつかのイベントをまとめてデータ解析を行えるようにプログラムを改善することが挙げられる。

９,３ 付録

付録１・・・データ収集プログラムコード
付録２・・・二つ山を探すプログラムコード

〈データ収集プログラム〉　付録１

```
/***** muon-daq.c ***** created 2002/Dec./25th Kenkichi Miya. *****
 * Original version was written by S.Ono 2002/Jan./27th
 * This is simplified version having only CAMAC control/IO part.
 * LAM clear was moved to outside of "if(q!=0)". 2003/Jul./9th
 ************************************************************/
/**** original title comment *****
  Data taking test program
  2000/ 1/27   S.Ono & A.Tango
  *******************************/
#include <fcntl.h>
#include <sys/time.h>
#include <unistd.h>
#include <sys/types.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <math.h>
#include <sys/errno.h>
#include "camlib.h"
#include <time.h>
#include <netinet/in.h>

FILE *fp; /* The file to save the taken data.*/

int main(){
  int i_ev, n_ev;
  int q,x,data1,data2,data3,data4,data5;
  int cadcqn,cadcvn,cfadcn;
  int ch1,ch2,ch3,ch4;
  int Nrec,fadcdata[32100],j,data6;



  cadcqn=1; /* ADC-Qmode module number */
  cadcvn=3; /* ADC-Vmode module number */
  cfadcn=5; /* FADC module addless */
  ch1=0;
  ch2=1;
  ch3=3;
  ch4=4;
 int  chfadc=1;

   /*=====
    * Ask the file name to save the taken data.
    * Also open the data file.
    *=====*/
     char fname[36];
    printf("File name to save data?¥n");
         scanf("%s",fname);
    fp=fopen(fname, "w");


   /*=====
    * How many events do you take?
```

```
 *====*/
 printf("Number of events?¥n");
 scanf("%d",&n_ev);

 /*=====
  * Open CCP interface device file.
  * If it fails, exit.
  *=====*/
 if(COPEN()){
   printf("ccp open error¥n");
   exit(-1);
 }

 /*=====
  * Initialize CAMAC.
  *=====*/
 CSETCR(0);
 CGENZ();
 CGENC();
 CREMI();

  int lamsrc=cfadcn;
  int lamch =chfadc;

printf("LAM source: %d, %d¥n",lamsrc, lamch);
/*=====
  * Enable LAM and Clear it.
  *=====*/
 CAMAC(NAF(lamsrc,lamch,26),&data1,&q,&x); /* F=26 is enable. */
 CAMAC(NAF(lamsrc,lamch, 9),&data1,&q,&x); /* F=9 is clear. */

 CAMAC(NAF(cadcqn,ch1,26),&data1,&q,&x); /* F=26 is enable. */
 CAMAC(NAF(cadcvn,ch1,26),&data1,&q,&x); /* F=26 is enable. */
 CAMAC(NAF(cadcqn,ch1,9),&data1,&q,&x); /* clear LAM. */
 CAMAC(NAF(cadcvn,ch1,9),&data1,&q,&x); /* clear LAM */
 /*=====
  * Again send enable command to prepare the first event.
  *=====*/
 CAMAC(NAF(lamsrc,lamch,26),&data1,&q,&x);
 CAMAC(NAF(cadcqn,ch1,26),&data1,&q,&x); /* F=26 is enable. */
 CAMAC(NAF(cadcvn,ch1,26),&data1,&q,&x); /* F=26 is enable. */

 /*=====
  * send message to the user.
  *=====*/
 printf("CAMAC initilize done.¥n");
 printf("Number of event = %d¥n",n_ev);

 /*====
  * Event loop.
  *====*/
 i_ev = 1;
 while( i_ev <= n_ev )
 {
   /*---
    *  Address clear for FADC
    ---*/
   CAMAC(NAF(cfadcn,chfadc,9),&data6,&q,&x);
```

```
 CAMAC(NAF(cfadcn,chfadc,10),&data6,&q,&x); /* F=10 FADC LAM clear. */
 CAMAC(NAF(cadcqn,ch1,9),&data1,&q,&x); /* F=9 LAM clear. */
 CAMAC(NAF(cadcvn,ch1,9),&data1,&q,&x); /* F=9 LAM clear. */


   /*---------
   * Test LAM.
   *---------*/

  do {
  CAMAC(NAF(lamsrc,lamch,8),&data5,&q,&x);  /* F=8 is test LAM.*/
  } while ( q==0);


  /*------
   * If no event comes yet, q is set to be 0,
   * otherwise, the digitized event is there!
   *-----*/

     /*-----
      * Read the digitized data from the register.
      *----*/
    printf("Event= %d¥n",i_ev);
    int qq;
     CAMAC(NAF(cadcqn,ch1,2),&data1,&qq,&x);
    printf("i_ev, data1, q, x = %d %d %d %d¥n", i_ev, data1, qq, x );
     CAMAC(NAF(cadcqn,ch2,2),&data2,&qq,&x);
     printf("i_ev, data2, q, x = %d %d %d %d¥n", i_ev, data2, qq, x );
     CAMAC(NAF(cadcqn,ch3,2),&data3,&qq,&x);
    printf("i_ev, data3, q, x = %d %d %d %d¥n", i_ev, data3, qq, x );
     CAMAC(NAF(cadcqn,ch4,2),&data4,&qq,&x);
    printf("i_ev, data4, q, x = %d %d %d %d¥n", i_ev, data4, qq, x );
     CAMAC(NAF(cadcvn,ch1,2),&data5,&qq,&x);

    printf("i_ev, data5, q, x = %d %d %d %d¥n", i_ev, data5, qq, x );
     /*-----
      * Update the event counter(i_ev), and send mesage for
      * every 100 events.
      *-----*/

     if( i_ev%100 == 0 )
      {
       printf("Event= %d¥t",i_ev);
       printf("data1= %d¥n",data1);
       printf("data2= %d¥n",data2);
       printf("data3= %d¥n",data3);
       printf("data4= %d¥n",data4);
       printf("data5= %d¥n",data5);
      }

    int head=-1;

   Nrec=0;

do
  {
     CAMAC(NAF(cfadcn,chfadc,2),&data6,&q,&x);
   Nrec=Nrec+1;
      if (Nrec>32099)
```

```
        {
          break;
        }
              fadcdata[Nrec]=data6;

  }
  while(q != 0);

printf("Nrec = %d¥n", Nrec);

//--
// output only when Nrec>20
//  If Nrec is small data has strange values
//   2009/12/26  HIH
//--
if(Nrec>=20) {
//--
//  output data
//--

     /*-----
      * The read data is written into the file.
      *-----*/

    fprintf(fp," %d",head);
    fprintf(fp," %d",i_ev);
     fprintf(fp," %d",data1);
     fprintf(fp," %d",data2);
     fprintf(fp," %d",data3);
     fprintf(fp," %d",data4);
     fprintf(fp," %d¥n",data5);

 const int Nmax=500;
 int fadcbuf[Nmax];
 for (j=0; j !=Nmax; j++){
   fadcbuf[j] = fadcdata[Nrec-Nmax+j];
   }

  for (  j = 0 ; j != Nmax ; j++ )
     {
    fprintf(fp," %d %d ¥n",j, fadcbuf[j]);

   }


   /*-----
    * Clear LAM to wait for the next event.
    *   fuc=9 ; LAM clear for usual module
    *   FADC
    *    fuc=9 ; address clear
    *      =10 ; LAM clear
    *-----*/
   i_ev++;
}// end if Nrec IF statement.

   CAMAC(NAF(lamsrc,lamch,9),&data1,&q,&x);
   CAMAC(NAF(lamsrc,lamch,10),&data1,&q,&x);
```

```
    }// end of event loop


  /*=======
   Instructions for termination.
  *======*/
    CAMAC(NAF(lamsrc,lamch,24),&data1,&q,&x); /* F=24 is desable lam.*/
    CAMAC(NAF(cadcqn,ch1,24),&data1,&q,&x); /* F=26 is enable. */
    CAMAC(NAF(cadcvn,ch1,24),&data1,&q,&x); /* F=26 is enable. */

    CCLOSE();                      /* CAMAC close. */
    fclose(fp);                    /* Close data file.*/

    return 0;
}
```

〈２つ山を探すプログラム〉　付録２

```
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <iostream>
#include <fstream>

/* analysis routine  */
int  peak_search(int );

/* data format    */
int iflag;
int iev;
int idat[4];
static const int nfmax=500;
int itime[nfmax];
int fadc[nfmax];
int diff[nfmax];
int degi_diff[nfmax];
int dif2[nfmax];
/* results  */

int ipeak2;

//==========main function==========


int main( int argc, char* argv[])
{



  int  i;
  int  j;
  int ifind;

  //----open file-txt file  -----//
  std::ifstream fin(argv[1]);

    if(!fin)
      {
         std::cout << "txt-file can not open !" <<std::endl ;
        return 1;}
      std::cout <<" file is opend"<<std::endl;


       i=0;

         while(!fin.eof())
           {
             if(fin.eof()) {return 1;}

             i++;
```

```
           fin >>iflag>> iev >> idat[0] >> idat[1]
                >>idat[2]>> idat[3] >> idat[4] ;

      /*  event data*/
          fadc[0]=18;
         int itim;
         int adc;
      if(iflag == -1) {

          for (int j=0; j <500; j++){

                  fin >>itim>> adc ;
              itime[j]=itim;
              fadc[j]=adc;

  } // fadc for loop

            ipeak2=0;
              int ifind=peak_search(i);


                 if(ifind==2) std::cout<<" event_no="<<i
                        <<" peak_dif= "<<ipeak2

                         <<"  CSI-Q =  " <<idat[1]

                         <<"  CSI-V =  " <<idat[4]

                              <<std::endl;
      }  //iflag=-1

        }      // while
       };  //end main

int peak_search(int ievent) {

        int ifind=0;
        fadc[0]=8;
        diff[0]=0;
        dif2[0]=0;
        degi_diff[0]=0;
        int vthdifp=3;
       int vthdifm=-4;
       int pedestal=8;
       int vthsig=10;


       for (int j=4; j<500; j++){

        diff[j]=fadc[j]-fadc[j-4];
        if( diff[j]>= vthdifp) {
                 degi_diff[j]=1;}
         else if( diff[j]<= vthdifm){
                degi_diff[j]=-1;
               }
         else {
               degi_diff[j]=0;}
        }
```

```
 for (int j=4; j<500; j++){

    dif2[j] = degi_diff[j] - degi_diff[j-1];

 }

int jpeak[500];    //jpeak : peak position
int adcpeak[500];  //adcpeak : peak pilse hight
for (int k=0; k<500; k++) {
    jpeak[k]=0;
    adcpeak[k]=0;

   }
 int nsame_peak=0;
 int prev_peak_sign=0;


 for (int j=4; j<500; j++){
   if( fadc[j]-pedestal>= vthsig) {
        if (dif2[j] == +1) { prev_peak_sign= +1;}
        if (dif2[j]==-1) {

              if (prev_peak_sign==-1) {nsame_peak +=1;
                 jpeak[nsame_peak]=j;
           // take a peak position 2-point latter.
                 adcpeak[nsame_peak]=fadc[j-2];
               }
           prev_peak_sign= -1;

        }
    }


 }// for loop

        ifind=0;

        ipeak2 = jpeak[2]-jpeak[1];

       double adcmax = adcpeak[1];
       if (adcpeak[2]> adcmax) adcmax=adcpeak[2];
       if(ipeak2>=10){
          if(nsame_peak>= 2 && adcmax< 255 )
                {ifind=2; }
       }

    return ifind;

    };
```

９,４参考文献

・奈良女子大学理学部物理科学科 2007年度卒業生 高坂玲加 馳川香菜実
『2007年度卒業論文 FADCを用いたμ粒子の寿命測定』
・奈良女子大学理学部物理科学科 2008年度卒業生 中牧理絵 松下絵理
『2008年度卒業論文 FADCを用いたμ粒子の寿命測定』

謝辞

私たちの卒業研究のために、お忙しい中多くの時間を割いてくださった林井先生をはじめ、ゼミや日々の実験で熱心にご指導いただいた宮林先生、諸先輩方に心より感謝しています。おかげさまで、この一年を充実して過ごすことができました。この一年で学んだことを今後も生かしていきたいと思います。
本当にありがとうございました。